Source Code for EMapKGS2: the WHEAT Electronic Mapper

by

Greg Pouch

Kansas Geological Survey
Open-file Report 95-65

1995

VB CROSS REFERENCE - TABLE OF CONTENTS

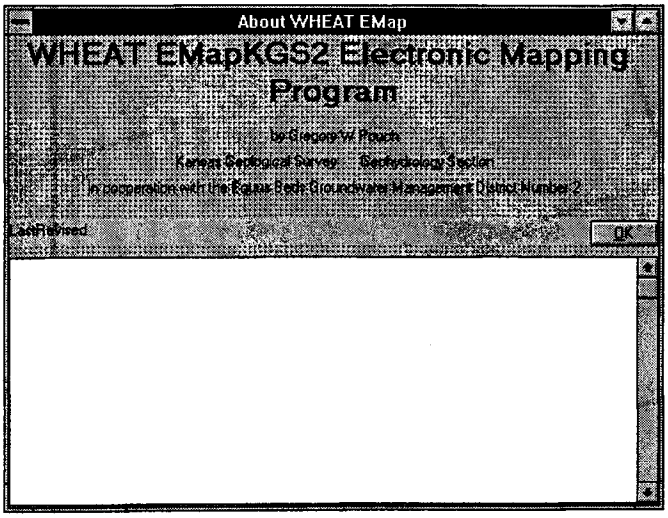
For file: EMAPKGS2.MAK
12/13/95 1:40:48 **PM**

Page #

Source Listing

ABOUT.FRM (illustration) 1
ABOUT.FRM: (declarations) 2
ASKOZ.FRM (illustration) 7
ASKOZ.FRM: (declarations) 8
FNTTBL01.FRM (illustration) 12
FNTTBL01.FRM: (declarations) 13
GRIDEDIT.FRM (illustration) 16
GRIDEDIT.FRM: (declarations) 17
LBLOPT22.FRM (illustration) 38
LBLOPT22.FRM: (declarations) 39
LINOPT02.FRM (illustration) 69
LINOPT02.FRM: (declarations) 70
LUTINDEX.FRM (illustration) 89
LUTINDEX.FRM: (declarations) 90
LUTLGND2.FRM (illustration) 95
LUTLGND2.FRM: (declarations) 96
MAP005.FRM (illustration) 97
MAP005.FRM: (declarations) 98
MDIWHT01.FRM (illustration) 184
MDIWHT01.FRM: (declarations) 185
PICKCHAR.FRM (illustration) 187
PICKCHAR.FRM: (declarations) 188
PNTOPT10.FRM (illustration) 192
PNTOPT10.FRM: (declarations) 193
POLYOPT4.FRM (illustration) 216
POLYOPT4.FRM: (declarations) 217
PRNTOPT3.FRM (illustration) 239
PRNTOPT3.FRM: (declarations) 240
RSTROPT1.FRM (illustration) 259
RSTROPT1.FRM: (declarations) 260
SPATQRY.FRM (illustration) 262
SPATQRY.FRM: (declarations) 263
SPTANLY1.FRM (illustration) 269
SPTANLY1.FRM: (declarations) 270
STATUS.FRM (illustration) 283
STATUS.FRM: (declarations) 284
THMORDR1.FRM (illustration) 292
THMORDR1.FRM: (declarations) 293
THMVISBL.FRM (illustration) 298
THMVISBL.FRM: (declarations) 299
TILEOPT7.FRM (illustration) 301
TILEOPT7.FRM: (declarations) 302
USERLINS.FRM (illustration) 328
USERLINS.FRM: (declarations) 329
USERPOLY.FRM (illustration) 351
USERPOLY.FRM: (declarations) 352
USERPTS1.FRM (illustration) 374
USERPTS1.FRM: (declarations) 375
VIEWOPT1.FRM (illustration) 397
VIEWOPT1.FRM: (declarations) 398

WHTCURS2.FRM (illustration) 412
WHTCURS2.FRM: (declarations) 413
ASKOZ.BAS: (declarations) 422
CURSOR01.BAS: (declarations) 427
DBUTIL01.BAS: (declarations) 431
DEBUGPRN.BAS: (declarations) 458
DFVRTPN2.BAS: (declarations) 459
EQVLNCE1.BAS: (declarations) 479
FNTTBL01.BAS: (declarations) 484
GEOMLIB2.BAS: (declarations) 489
GRATICUL.BAS: (declarations) 510
LINPLY01.BAS: (declarations) 522
LOADGEOM.BAS: (declarations) 531
LUTCOLOR.BAS: (declarations) 554
MAPHLPR1.BAS: (declarations) 561
MAPPRINT.BAS: (declarations) 573
MAPWMF2.BAS: (declarations) 590
MEMALLOC.BAS: (declarations) 611
RASTER01.BAS: (declarations) 615
SPATQRY1.BAS: (declarations) 641
THMEDTPP.BAS: (declarations) 654


About WHEAT EMap
WHEAT EMapKGS2 Electronic Mapping Program
OK

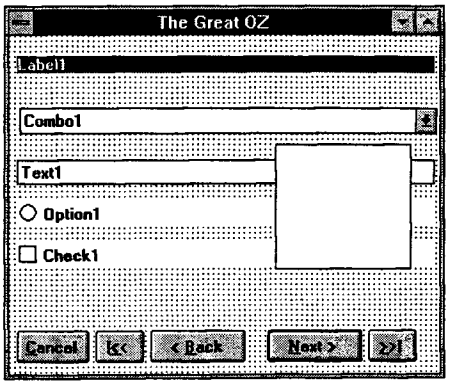
```
VERSION 2.00
Begin Form frmAboutBox
  BackColor      = &H00C0C0C0&
  BorderStyle    = 3 'Fixed Double
  Caption        = "About WHEAT EMap"
  ClientHeight   = 5490
  ClientLeft     = 1080
  ClientTop      = 1110
  ClientWidth    = 7515
  Height         = 5925
  Left           = 1020
  LinkTopic      = "Form1"
  MaxButton      = 0 'False
  MinButton      = 0 'False
  ScaleHeight    = 5490
  ScaleWidth     = 7515
  Top            = 735
  Width          = 7635
  Begin Timer Timer1
    Left         = 4320
    Top          = 5460
  End
  Begin CommandButton cmdOK
    Caption      = "&OK"
    FontBold     = 0 'False
    FontItalic   = 0 'False
    FontName     = "MS Sans Serif"
    FontSize     = 8.25
    FontStrikethru = 0 'False
    FontUnderline  = 0 'False
    Height       = 300
    Left         = 6720
    TabIndex     = 5
    Top          = 2160
    Width        = 800
  End
  Begin TextBox txtDisclaimer
    FontBold     = 0 'False
    FontItalic   = 0 'False
    FontName     = "MS Sans Serif"
    FontSize     = 8.25
    FontStrikethru = 0 'False
    FontUnderline  = 0 'False
    Height       = 2895
    Left         = 0
    MultiLine    = -1 'True
    ScrollBars   = 2 'Vertical
    TabIndex     = 4
    Top          = 2580
    Width        = 7515
  End
  Begin Label lblRevisionDate
    BackColor    = &H00C0C0C0&
    Caption      = "LastRevised"
```

```
    FontBold      = 0  'False
    FontItalic    = 0  'False
    FontName      = "MS Sans Serif"
    FontSize      = 8.25
    FontStrikethru = 0  'False
    FontUnderline = 0  'False
    Height        = 255
    Left          = 0
    TabIndex      = 6
    Top           = 2160
    Width         = 6735
  End
  Begin Label lblSponsor
    Alignment     = 2  'Center
    BackColor     = &H00C0C0C0&
    BackStyle     = 0  'Transparent
    Caption       = "in cooperation with the Equus Beds Groundwater Management
District Number 2"
    FontBold      = 0  'False
    FontItalic    = 0  'False
    FontName      = "MS Sans Serif"
    FontSize      = 8.25
    FontStrikethru = 0  'False
    FontUnderline = 0  'False
    Height        = 435
    Left          = 0
    TabIndex      = 3
    Top           = 1680
    Width         = 7500
  End
  Begin Label lblOrganization
    Alignment     = 2  'Center
    BackColor     = &H00C0C0C0&
    BackStyle     = 0  'Transparent
    Caption       = "Kansas Geological Survey     Geohydrology Section"
    FontBold      = 0  'False
    FontItalic    = 0  'False
    FontName      = "MS Sans Serif"
    FontSize      = 8.25
    FontStrikethru = 0  'False
    FontUnderline = 0  'False
    Height        = 255
    Left          = 0
    TabIndex      = 2
    Top           = 1380
    Width         = 7500
  End
  Begin Label lblAuthor
    Alignment     = 2  'Center
    BackColor     = &H00C0C0C0&
    BackStyle     = 0  'Transparent
    Caption       = "by Gregory W Pouch"
    FontBold      = 0  'False
    FontItalic    = 0  'False
```

```
108|      FontName      = "MS Sans Serif"
109|      FontSize      = 8.25
110|      FontStrikethru = 0  'False
111|      FontUnderline  = 0  'False
112|      Height        = 255
113|      Left          = 0
114|      TabIndex      = 1
115|      Top           = 1080
116|      Width         = 7500
117|    End
118|    Begin Label lblTitle
119|      Alignment     = 2  'Center
120|      AutoSize      = -1 'True
121|      BackColor     = &H00C0C0C0&
122|      BackStyle     = 0  'Transparent
123|      Caption       = "WHEAT EMapKGS2 Electronic Mapping Program"
124|      FontBold      = -1 'True
125|      FontItalic    = 0  'False
126|      FontName      = "MS Sans Serif"
127|      FontSize      = 18
128|      FontStrikethru = 0  'False
129|      FontUnderline  = 0  'False
130|      Height        = 870
131|      Left          = 0
132|      TabIndex      = 0
133|      Top           = 0
134|      Width         = 7500
135|      WordWrap      = -1 'True
136|    End
137| End
138| Option Explicit
139| Option Compare Text
140| Dim strDisclaimer As String
141|
142| Sub cmdOK_Click ()
143|  Me.Hide
144| End Sub
145|
146| Sub Form_Load ()
147|    Call GetDisclaimer
148|    On Error Resume Next
149|    Me!lblRevisionDate.Caption = App.Path & "\" & App.EXEName & ".EXE last revised " &
     Format$(FileDateTime(App.Path & "\" & App.EXEName & ".EXE"))
150| End Sub
151|
152| Sub Form_Paint ()
153|    If Me!txtDisclaimer <> strDisclaimer Then Me!txtDisclaimer = strDisclaimer
154| End Sub
155|
156| Sub GetDisclaimer ()
157|
158|      strDisclaimer = ""
159|      strDisclaimer = strProgName & strVersion & " is the Electronic MAP part of the
     WHEAT, the " & CRLF
```

```
160|     strDisclaimer = strDisclaimer & "Windows-based, Hydrogeologic Exploration and
       Appraisal Toolkit." & CRLF
161|     strDisclaimer = strDisclaimer & "This set of programs was developed by Greg Pouch "
162|     strDisclaimer = strDisclaimer & "at the Kanas Geological Survey, Geohydrology
       Section,"
163|     strDisclaimer = strDisclaimer & "under contract to the "
164|     strDisclaimer = strDisclaimer & "Equus Beds Groundwater Management District,
       Halstead, Kansas.  (KGS OFR 95-xx)" & CRLF & CRLF
165|
166|
167|     strDisclaimer = strDisclaimer & "E-MAP is provided free of charge and without any
       warranty of any kind.  "
168|     strDisclaimer = strDisclaimer & "Free technical support for using or adding features to
       this program or for designing databases is not available." & CRLF & CRLF
169|
170|     strDisclaimer = strDisclaimer & "DISCLAIMER" & CRLF & "Neither the Equus Beds
       Groundwater Management District nor "
171|     strDisclaimer = strDisclaimer & "the Kansas Geological Survey guarantees this
       computer program "
172|     strDisclaimer = strDisclaimer & "to be free from errors or inaccuracies and disclaims
       any "
173|     strDisclaimer = strDisclaimer & "responsibility or liability for interpretations based on
       data "
174|     strDisclaimer = strDisclaimer & "processed using this software or decisions based
       thereon." & CRLF & CRLF
175|                                  '& CRLF
176|
177|    Dim SectionName As String, Buffer As String, KeyString As String, DefaultValue As
       String
178|     SectionName = "Disclaimer"
179|     Buffer = Space$(1024)
180|
181|     Dim n As Long, nBytesBack As Long
182|     Do
183|       n = n + 1
184|       KeyString = "Disclaimer" & n
185|       DefaultValue = ""
186|       nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
       1024, INIFile)
187|       strDisclaimer = strDisclaimer & CRLF & Left$(Buffer, nBytesBack)
188|     Loop While nBytesBack <> 0
189|
190|     Dim sngHowLong As Single
191|       KeyString = "HowLongToShowAboutBox"
192|       DefaultValue = "0"   'This will disable the timer
193|       nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
       1024, INIFile)
194|       sngHowLong = Val(Left$(Buffer, nBytesBack)) * 1000
195|       If sngHowLong > 65535 Then sngHowLong = 60000
196|       If sngHowLong < 0 Then sngHowLong = 60000
197|       Me!Timer1.Interval = sngHowLong
198|
199| End Sub
200|
```

```
201| Sub Timer1_Timer ()
202|     Me.Hide
203|     Me!Timer1.Interval = 0
204| End Sub
205|
```

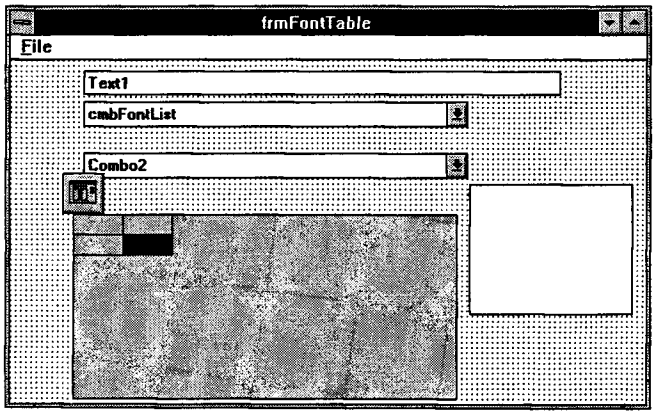

The Great OZ
Label1
Combo1
Text1
Option1
Check1
Cancel
|<<
< Back
Next >
>>|

```
VERSION 2.00
Begin Form frmOz
  Caption       = "The Great OZ"
  ClientHeight  = 3930
  ClientLeft    = 2805
  ClientTop     = 1230
  ClientWidth   = 4965
  ControlBox    = 0  'False
  Height        = 4335
  Left          = 2745
  LinkTopic     = "Form1"
  ScaleHeight   = 69.321
  ScaleMode     = 6  'Millimeter
  ScaleWidth    = 87.577
  Top           = 885
  Width         = 5085
  Begin CommonDialog CMDialog1
    Index       = 0
    Left        = 6210
    Top         = 2880
  End
  Begin CommandButton cmdCancel
    Cancel      = -1  'True
    Caption     = "&Cancel"
    Height      = 375
    Index       = 0
    Left        = 90
    TabIndex    = 11
    Top         = 3420
    Width       = 825
  End
  Begin CommandButton cmdNext
    Caption     = "&Next >"
    Default     = -1  'True
    Height      = 375
    Index       = 0
    Left        = 2970
    TabIndex    = 10
    Top         = 3420
    Width       = 1095
  End
  Begin CommandButton cmdBack
    Caption     = "< &Back"
    Height      = 375
    Index       = 0
    Left        = 1620
    TabIndex    = 9
    Top         = 3420
    Width       = 1095
  End
  Begin CommandButton cmdStartOver
    Caption     = "|&<<"
    Height      = 375
    Index       = 0
```

```
   55|    Left        = 990
   56|    TabIndex     = 8
   57|    Top         = 3420
   58|    Width       = 555
   59|  End
   60|  Begin CommandButton cmdAllDefaults
   61|    Caption     = "&>>|"
   62|    Height      = 375
   63|    Index       = 0
   64|    Left        = 4140
   65|    TabIndex     = 7
   66|    Top         = 3420
   67|    Width       = 555
   68|  End
   69|  Begin CommandButton Command1
   70|    Caption     = "Command3"
   71|    Height      = 375
   72|    Index       = 0
   73|    Left        = 5130
   74|    TabIndex     = 6
   75|    Top         = 3420
   76|    Width       = 1695
   77|  End
   78|  Begin PictureBox Picture1
   79|    Height      = 1455
   80|    Index       = 0
   81|    Left        = 3060
   82|    ScaleHeight  = 1425
   83|    ScaleWidth   = 1545
   84|    TabIndex     = 5
   85|    Top         = 1260
   86|    Width       = 1575
   87|  End
   88|  Begin CheckBox Check1
   89|    Caption     = "Check1"
   90|    Height      = 255
   91|    Index       = 0
   92|    Left        = 120
   93|    TabIndex     = 4
   94|    Top         = 2400
   95|    Width       = 4755
   96|  End
   97|  Begin OptionButton Option1
   98|    Caption     = "Option1"
   99|    Height      = 255
  100|    Index       = 0
  101|    Left        = 120
  102|    TabIndex     = 3
  103|    Top         = 1920
  104|    Width       = 4755
  105|  End
  106|  Begin TextBox Text1
  107|    Height      = 285
  108|    Index       = 0
```

```
109|      Left          =  90
110|      TabIndex      =  2
111|      Text          =  "Text1"
112|      Top           =  1440
113|      Width         =  4815
114|    End
115|    Begin ComboBox Combo1
116|      Height        =  300
117|      Index         =  0
118|      Left          =  120
119|      Style         =  2  'Dropdown List
120|      TabIndex      =  1
121|      Top           =  840
122|      Width         =  4815
123|    End
124|    Begin Label lblUserChoice
125|      Caption       =  "0"
126|      Height        =  375
127|      Index         =  0
128|      Left          =  4950
129|      TabIndex      =  12
130|      Top           =  3060
131|      Visible       =  0  'False
132|      Width         =  195
133|    End
134|    Begin Label Label1
135|      AutoSize      =  -1  'True
136|      BackColor     =  &H00000080&
137|      Caption       =  "Label1"
138|      ForeColor     =  &H0000FFFF&
139|      Height        =  195
140|      Index         =  0
141|      Left          =  90
142|      TabIndex      =  0
143|      Top           =  240
144|      Width         =  4815
145|      WordWrap      =  -1  'True
146|    End
147| End
148| Option Explicit
149|
150| Sub cmdAllDefaults_Click (Index As Integer)
151| cmdAllDefaults(Index).SetFocus
152| lblUserChoice(0).Caption = 2
153| Me.Hide
154| End Sub
155|
156| Sub cmdBack_Click (Index As Integer)
157| cmdBack(Index).SetFocus
158| lblUserChoice(0) = -1
159| Me.Hide
160| End Sub
161|
162| Sub cmdCancel_Click (Index As Integer)
```

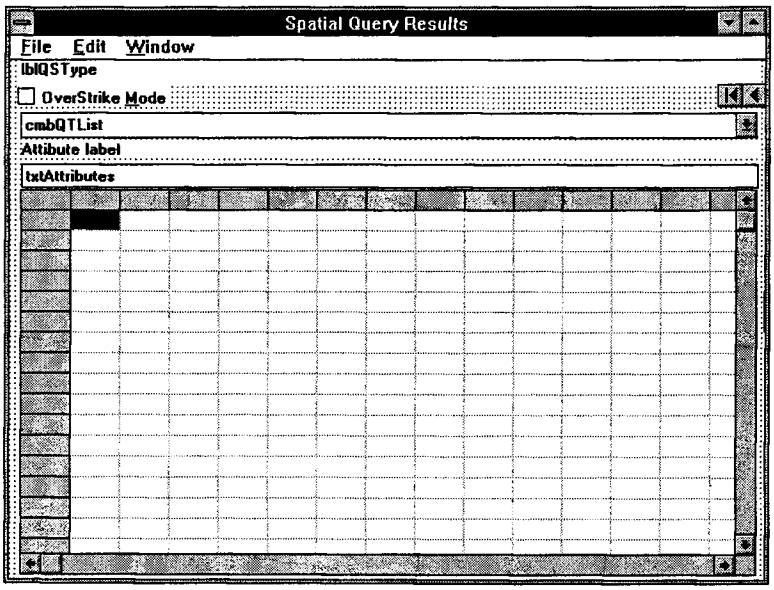
```
163| Me.cmdCancel(Index).SetFocus
164| lblUserChoice(0) = 0
165| Me.Hide
166| End Sub
167|
168| Sub cmdNext_Click (Index As Integer)
169| cmdNext(Index).SetFocus
170| lblUserChoice(0) = 1
171| Me.Hide
172| End Sub
173|
174| Sub cmdStartOver_Click (Index As Integer)
175| cmdStartOver(0).SetFocus
176| lblUserChoice(0) = -2
177| Me.Hide
178| End Sub
179|
180| Sub Command1_Click (Index As Integer)
181|   If Me.Command1(Index).FontItalic = True Then
182|     Me.Command1(Index).FontItalic = False
183|   Else
184|     Me.Command1(Index).FontItalic = True
185|   End If
186| End Sub
187|
```


frmFontTable
File
Text1
cmbFontList
Combo2

```
VERSION 2.00
Begin Form frmFontTable
  Caption        = "frmFontTable"
  ClientHeight   = 4020
  ClientLeft     = 1500
  ClientTop      = 1725
  ClientWidth    = 7365
  Height         = 4710
  Left           = 1440
  LinkTopic      = "Form1"
  ScaleHeight    = 4020
  ScaleWidth     = 7365
  Top            = 1095
  Width          = 7485
  Begin PictureBox Picture1
    Height       = 1545
    Left         = 5310
    ScaleHeight  = 1515
    ScaleWidth   = 1875
    TabIndex     = 4
    Top          = 1440
    Width        = 1905
  End
  Begin CommonDialog CMDialog1
    Left         = 600
    Top          = 1320
  End
  Begin TextBox Text1
    Height       = 285
    Left         = 840
    TabIndex     = 3
    Text         = "Text1"
    Top          = 120
    Width        = 5535
  End
  Begin Grid Grid1
    Height       = 2175
    Left         = 720
    TabIndex     = 2
    Top          = 1800
    Width        = 4455
  End
  Begin ComboBox Combo2
    Height       = 300
    Left         = 840
    Style        = 2 'Dropdown List
    TabIndex     = 1
    Top          = 1080
    Width        = 4455
  End
  Begin ComboBox cmbFontList
    Height       = 300
    Left         = 840
    Sorted       = -1 'True
```

```
55|      Style         =  2  'Dropdown List
56|      TabIndex      =  0
57|      Top           =  480
58|      Width         =  4455
59|    End
60|    Begin Menu zmnuFile
61|      Caption       =  "&File"
62|      Begin Menu mfOpen
63|        Caption     =  "&Open"
64|      End
65|      Begin Menu mfClose
66|        Caption     =  "&Close"
67|      End
68|      Begin Menu zz1
69|        Caption     =  "-"
70|      End
71|      Begin Menu mfExit
72|        Caption     =  "E&xit"
73|      End
74|    End
75| End
76| Option Explicit
77| Dim DB As database, tblIn As Table
78|
79| Sub Combo2_Click ()
80| text1 = Format(Combo2.ListIndex)
81| End Sub
82|
83| Sub mfExit_Click ()
84|   End
85| End Sub
86|
87| Sub mfOpen_Click ()
88| On Error GoTo ERRmfOpen
89| Dim DBName As String' May want to change scope of this later to global.
90| CMDialog1.DialogTitle = "Database to Open"
91| CMDialog1.Filter = "MS Access Databases|*.mdb"
92| CMDialog1.Flags = &H1000& Or &H4000&   'OFN_FILEMUSTEXIST, allow sharing
93| CMDialog1.Action = 1
94| Dim Temp As String
95| Temp = CMDialog1.Filename
96|
97| If Len(Temp) = 0 Then Exit Sub 'the user did not enter a file name
98|
99| Set DB = OpenDatabase(Temp, False, False, "")
100| Set tblIn = DB.OpenTable("tblFontList")
101|
102| frmFontTable.Show
103| '  Call CreateFontTable(tblIn, FontTable())
104| Call LoadFontTable(tblIn, FontTable())
105| Call LoadComboBoxWithFontNames(frmFontTable.Combo2)
106|
107|  text1 = Format(Combo2.ListIndex)
108|
```

```
109|     DBName = Temp
110|     ' Open an MSAccess database for non-exclusive, read-write access.
111|     'Set DB = OpenDatabase(DBName, False, False, "") 'Open the desired database
112|  Exit Sub
113|
114|  ERRmfOpen:
115|
116|  If Err = 3011 Then ' no tblFontList in the current database, so create one after prompting
     the user.
117|     Dim lAns As Long
118|     lAns = MsgBox("There is no Font List table that this program recognizes in the current
     database.  Do you wish to create one before proceeding?", 35, "Table Not Found")
119|     Select Case lAns
120|        Case 2 'User chose cancel
121|           Err = 0
122|           Exit Sub
123|           Resume Next
124|        Case 6 ' Yes
125|           Call DefineFontTable(DB)'  Dim tdfFonts As New tabledef, ndxFonts As New
     Index, fldName As New field, fldNumber As New field, fldExtra As New field
126|           Resume
127|        Case 7
128|           Err = 0
129|           Exit Sub
130|           Resume
131|        Case Else
132|           MsgBox "Anwer " & lAns & " was returned, and is not handled here."
133|     End Select
134|  End If
135|  MsgBox "Error #" & Err & " while opening database.  " & Error
136|  Err = 0
137|  Exit Sub
138|  Resume
139|  End Sub
140|
```


Spatial Query Results
File Edit Window
lblQSType
OverStrike Mode
cmbQTList
Attibute label
txtAttributes

```
VERSION 2.00
Begin Form frmGridDataEdit
  Caption        = "Spatial Query Results"
  ClientHeight   = 6060
  ClientLeft     = 495
  ClientTop      = 795
  ClientWidth    = 8745
  Height         = 6750
  Icon           = GRIDEDIT.FRX:0000
  Left           = 435
  LinkTopic      = "Form1"
  MDIChild       = -1 'True
  ScaleHeight    = 106.892
  ScaleMode      = 6 'Millimeter
  ScaleWidth     = 154.252
  Top            = 165
  Width          = 8865
  Begin ComboBox cmbQTList
    Height       = 300
    Left         = 120
    Sorted       = -1 'True
    Style        = 2 'Dropdown List
    TabIndex     = 2
    Top          = 600
    Visible      = 0 'False
    Width        = 8535
  End
  Begin CommonDialog CMDialog1
    Left         = 8730
    Top          = 2700
  End
  Begin Data Data1
    Caption      = "Move record "
    Connect      = ""
    DatabaseName = ""
    Exclusive    = 0 'False
    Height       = 270
    Left         = 8190
    Options      = 0
    ReadOnly     = 0 'False
    RecordSource = ""
    Top          = 270
    Visible      = 0 'False
    Width        = 2445
  End
  Begin CheckBox chkOverstrike
    Caption      = "OverStrike &Mode"
    Height       = 255
    Left         = 90
    TabIndex     = 1
    Top          = 285
    Width        = 1757
  End
  Begin TextBox txtAttributes
```

```
55|     Height      = 285
56|     Left        = 135
57|     TabIndex    = 4
58|     Text        = "txtAttributes"
59|     Top         = 1200
60|     Width       = 8535
61|   End
62|   Begin Grid grdAttributes
63|     Cols        = 15
64|     Height      = 4515
65|     Left        = 120
66|     Rows        = 30
67|     TabIndex    = 5
68|     Top         = 1500
69|     Width       = 8535
70|   End
71|   Begin Label lblQSSource
72|     Caption     = "lblQSSource"
73|     Height      = 255
74|     Left        = 120
75|     TabIndex    = 6
76|     Top         = 600
77|     Width       = 8415
78|   End
79|   Begin Label lblQSType
80|     Caption     = "lblQSType"
81|     Height      = 255
82|     Left        = 120
83|     TabIndex    = 0
84|     Top         = 0
85|     Width       = 8535
86|   End
87|   Begin Label lblAttributes
88|     Caption     = "Attibute label"
89|     Height      = 210
90|     Left        = 135
91|     TabIndex    = 3
92|     Top         = 930
93|     Width       = 8535
94|   End
95|   Begin Menu zzzzFile
96|     Caption     = "&File"
97|     Begin Menu mfOpenDB
98|       Caption   = "&Open Database..."
99|       Visible   = 0  'False
100|    End
101|    Begin Menu mfCloseDB
102|      Caption   = "&Close Database"
103|      Visible   = 0  'False
104|    End
105|    Begin Menu zz10
106|      Caption   = "-"
107|    End
108|    Begin Menu mfNewTable
```

```
109|       Caption      =  "New &Table..."
110|     End
111|     Begin Menu zzz1
112|       Caption      =  "-"
113|     End
114|     Begin Menu mfNewFile
115|       Caption      =  "New &File..."
116|     End
117|     Begin Menu zz3
118|       Caption      =  "-"
119|       Visible      =  0  'False
120|     End
121|     Begin Menu mfSaveTableChanges
122|       Caption      =  "&Save Changes"
123|       Visible      =  0  'False
124|     End
125|     Begin Menu mfAbandonTableChanges
126|       Caption      =  "&Abandon Changes"
127|       Visible      =  0  'False
128|     End
129|     Begin Menu zzzzz1
130|       Caption      =  "-"
131|     End
132|     Begin Menu mfExit
133|       Caption      =  "E&xit"
134|     End
135|   End
136|   Begin Menu zzEdit
137|     Caption        =  "&Edit"
138|     Begin Menu meUndoThisRow
139|       Caption      =  "&Undo changes to this row"
140|       Shortcut     =  %{BKSP}
141|     End
142|     Begin Menu zz6
143|       Caption      =  "-"
144|     End
145|     Begin Menu meCopy
146|       Caption      =  "&Copy"
147|       Shortcut     =  ^{INSERT}
148|     End
149|     Begin Menu mePaste
150|       Caption      =  "&Paste"
151|       Shortcut     =  +{INSERT}
152|     End
153|     Begin Menu meCut
154|       Caption      =  "Cu&t"
155|       Shortcut     =  +{DEL}
156|     End
157|     Begin Menu meClearGrid
158|       Caption      =  "C&lear"
159|       Shortcut     =  ^X
160|     End
161|     Begin Menu zzzz1
162|       Caption      =  "-"
```

```
163|     End
164|     Begin Menu meInsertRow
165|       Caption      = "&Append Records"
166|       Shortcut     = ^A
167|     End
168|     Begin Menu meDeleteRow
169|       Caption      = "&Delete Records"
170|       Shortcut     = ^D
171|     End
172|     Begin Menu zzz5
173|       Caption      = "-"
174|     End
175|     Begin Menu meAddColumn
176|       Caption      = "Add Co&lumn"
177|     End
178|     Begin Menu meColDefinition
179|       Caption      = "Column De&finition"
180|     End
181|   End
182|   Begin Menu mnuWindow
183|     Caption        = "&Window"
184|     WindowList     = -1  'True
185|   End
186| End
187| Option Explicit
188|
189| Dim flgChangeRow As Integer, flgIgnoreMovement As Integer
190| Dim flgChangesWereMade As Integer, flgBookmarkable As Integer
191|
192| Dim strBookMark() As String
193| Dim flgSkipMe() As Long
194| Dim nFields As Long, n As Long, nRecs As Long, nR As Long
195| Dim nF As Long, FieldType() As Long
196| Dim lngCurrentRow As Long, lngCurrentCol As Long
197|
198| Dim DB As Database, tblLineData As Table
199| 'Dim snpQTList As Snapshot, dnsCurrentQT As dynaset
200|
201| Dim flgAddingRecord As Integer   'Might want to eliminate this variable.
202| Dim TabChar As String * 1, CR  As String * 1, CRLF  As String * 2
203| Dim ListOfFieldTypes As String
204|
205| Sub cmbQTList_Click ()
206| Dim Temp As String
207| mousepointer = 11
208|
209| Temp = cmbQTList
210| Call SetQuery(Temp)
211| flgIgnoreMovement = True
212| grdAttributes.FixedCols = 0: grdAttributes.Cols = 1
213| grdAttributes.Cols = 2: grdAttributes.FixedCols = 1
214| grdAttributes.Rows = 2
215| 'Function GetClipBoardTextFromDynaset (dynIn As Dynaset, flgOptions As Long) As
     String
```

```
Set dynViewInGrid = DB.CreateDynaset(Temp)
Call InitializeGridForDynaset(Me!grdAttributes, dynViewInGrid)

Temp = GetClipBoardTextFromDynaset(dynViewInGrid, 0)
Call SetGridFocusColumn(Me!grdAttributes, 1)
Call FillGridFromClipBoardText(Temp, Me!grdAttributes, 12&)

grdAttributes.RemoveItem 1


'Call OverwriteGridFromDynaset(Me!grdAttributes, dynViewInGrid)
flgIgnoreMovement = False
mousepointer = 0

End Sub

Sub cmdClearGrid_Click ()
 flgIgnoreMovement = True
  grdAttributes.FixedRows = 0
  grdAttributes.Rows = 1
  grdAttributes.Rows = 2
  grdAttributes.FixedRows = 1
 flgIgnoreMovement = False
End Sub

Sub Data1_Reposition ()
'On Error Resume Next
'DisplayCurrentRecord
End Sub

Sub DefineColumn ()
    flgIgnoreMovement = True
    Dim Temp As String, tmpFieldName As String, tmpFieldType As Integer,
strFieldTypeList As String
    grdAttributes.Row = 0
    n = grdAttributes.Col
    Temp = grdAttributes
    If Temp = "" Then
   tmpFieldName = "Field" & Format$(n, "00")
   tmpFieldType = 10'Default to text
      Else
   tmpFieldType = Val(Right(Temp, 3))
   If tmpFieldType <> 0 Then
      tmpFieldName = Left(Temp, Len(Temp) - 3)
      Else
       tmpFieldName = Temp
       tmpFieldType = 10
   End If
      End If

   ReDim OZQuestion(1 To 4)  As typOZQuestion
   Dim Accepted As Integer
   'Define the name.
   OZQuestion(1).KindOfControl = CONTROL_LABEL
```

```
269|    OZQuestion(1).StringData = "This field is currently named: " & Temp & " ." & CRLF &
       "If you wish, you may enter a new name below." '& CRLF & "and droning on and on and
       on about absolutely nothing at all in the world of any possible interest to anyone."
270|    OZQuestion(2).KindOfControl = CONTROL_TEXTBOX
271|    OZQuestion(2).StringData = tmpFieldName
272|
273|    'Define the datafield type.
274|    OZQuestion(3).KindOfControl = CONTROL_LABEL
275|    OZQuestion(3).StringData = "It currently has a datatype of " & tmpFieldType
276|    OZQuestion(4).KindOfControl = CONTROL_COMBOBOX
277|    strFieldTypeList = ListOfFieldTypes
278|    OZQuestion(4).StringData = strFieldTypeList
279|    OZQuestion(4).Extra = tmpFieldType
280|  'Consult the Wizard.
281|    AskOZ OZQuestion(), Accepted
282|
283|    If Accepted Then
284|       tmpFieldType = OZQuestion(4).Extra
285|       tmpFieldName = OZQuestion(2).StringData
286|       grdAttributes = tmpFieldName & " " & Format$(tmpFieldType, "00")
287|    End If
288|    flgIgnoreMovement = False
289|
290|  End Sub
291|
292|  Sub DisplayCurrentRecord ()
293|  On Error GoTo ERRDisplayCurrentRecord
294|  flgIgnoreMovement = True
295|  grdAttributes.Row = 1
296|  Dim Temp As String
297|  For n = 1 To nFields
298|     grdAttributes.Col = n
299|     grdAttributes = Format(Data1.Recordset.Fields(n - 1).Value)
300|  Next n
301|
302|
303|  flgIgnoreMovement = False
304|  Exit Sub
305|
306|  ERRDisplayCurrentRecord:
307|  If Err = 30010 Then 'Invalid number of columns or rows
308|     Err = 0
309|     Exit Sub
310|     Resume Next
311|  End If
312|  MsgBox "Error number " & Err & " encountered while displaying a record." & CRLF &
       Error$
313|  Err = 0
314|  Exit Sub
315|  Resume
316|
317|
318|
319|  End Sub
```

```
320|
321| Sub Form_Load ()
322|
323| CR = Chr$(13)
324| TabChar = Chr$(9)
325| CRLF = Chr$(13) & Chr$(10)
326| ListOfFieldTypes = ""
327|
328| 'cmbTypeList.Clear
329| 'cmbTypeList.AddItem "Do not save this field", 0
330| 'cmbTypeList.AddItem "BOOLEAN", 1
331| 'cmbTypeList.AddItem "BYTE", 2
332| 'cmbTypeList.AddItem "Short Integer", 3
333| 'cmbTypeList.AddItem "Long Integer 4-bytes", 4
334| 'cmbTypeList.AddItem "CURRENCY", 5
335| 'cmbTypeList.AddItem "Single Precision REAL*4", 6
336| 'cmbTypeList.AddItem "Double Precision REAL*8", 7
337| 'cmbTypeList.AddItem "DATE", 8
338| 'cmbTypeList.AddItem "Do not Save this field", 9
339| 'cmbTypeList.AddItem "TEXT", 10
340| 'cmbTypeList.AddItem "LONGBINARY", 11
341| 'cmbTypeList.AddItem "MEMO (Long String)", 12
342|
343|
344| ListOfFieldTypes = "Do not save this field" & CRLF & "BOOLEAN" & CRLF & "BYTE" &
    CRLF
345| ListOfFieldTypes = ListOfFieldTypes & "Short Integer" & CRLF & "Long Integer 4-bytes"
    & CRLF
346| ListOfFieldTypes = ListOfFieldTypes & "CURRENCY" & CRLF & "Single Precision
    REAL*4" & CRLF & "Double Precision REAL*8" & CRLF
347| ListOfFieldTypes = ListOfFieldTypes & "DATE" & CRLF & "Do not Save this field" &
    CRLF & "TEXT" & CRLF & "LONGBINARY" & CRLF & "MEMO (Long String)" & CRLF
348|
349|
350|
351|
352|
353| End Sub
354|
355| Sub Form_QueryUnload (Cancel As Integer, UnloadMode As Integer)
356| 'Check whether changes need to be saved.
357|
358| If flgChangesWereMade Then
359|     'Present a yes/no/cancel box to the user.
360|   nF = MsgBox("Changes were made to this data set." & CRLF & "Do you wish to save
    them now?", 35)
361|   If nF = 2 Then
362|     Cancel = True
363|     Exit Sub'user chose cancel, so clear out.
364|     End If
365|   If nF = 6 Then 'User chose to save changes
366|     Call mfSaveTableChanges_Click
367|     If flgChangesWereMade Then Cancel = True 'Still need to cancel
368|   End If
```

```
369|    'flgChangesWereMade = False  'If the user chose yes or no, reset the flag.
370|  End If
371|
372|  End Sub
373|
374|  Sub Form_Resize ()
375|  If frmGridDataEdit.WindowState = 1 Then Exit Sub'iconized
376|  On Error GoTo ERRResize
377|     lblAttributes.Width = frmGridDataEdit.ScaleWidth - 5
378|     txtAttributes.Width = frmGridDataEdit.ScaleWidth - 5
379|     grdAttributes.Width = frmGridDataEdit.ScaleWidth - 5
380|     grdAttributes.Height = frmGridDataEdit.ScaleHeight - 27.5
381|
382|  cmbQTList.Width = Me.ScaleWidth - 10
383|
384|  Exit Sub
385|
386|  ERRResize:
387|  If Err = 380 Then 'invalid width
388|    frmGridDataEdit.Width = frmGridDataEdit.Width * 1.2
389|    Resume
390|  End If
391|
392|
393|  End Sub
394|
395|  Sub GetIgnoreMovement (FromValue As Long)
396|   FromValue = flgIgnoreMovement
397|  End Sub
398|
399|  Sub grdAttributes_KeyDown (Keycode As Integer, Shift As Integer)
400|  If Keycode > 48 Then
401|     txtAttributes.SetFocus
402|  End If
403|  End Sub
404|
405|  Sub grdAttributes_KeyPress (KeyAscii As Integer)
406|     If KeyAscii = 13 Then SendKeys "{DOWN}"
407|     If KeyAscii > 31 Then SendKeys Chr$(KeyAscii)
408|  End Sub
409|
410|  Sub grdAttributes_RowColChange ()
411|    'every time a new cell is selected, the text appears
412|    ' in textbox text1 and text1 receives focus.
413|    'Updates of text1 automatically appear in the grid.
414|  If flgIgnoreMovement Then Exit Sub
415|
416|     flgIgnoreMovement = True
417|     lngCurrentRow = grdAttributes.Row
418|     lngCurrentCol = grdAttributes.Col
419|     txtAttributes = grdAttributes.Text
420|     grdAttributes.Row = 0
421|     lblAttributes.Caption = grdAttributes.Text
422|     grdAttributes.Row = lngCurrentRow
```

```
423|     flgIgnoreMovement = False
424|
425|  End Sub
426|
427|  Sub GridClipNRowsNCols (ClipText As String, nRows As Long, nCols As Long)
428|     nCols = 1
429|     nRows = 1
430|     Dim TabChar As String * 1, CR  As String * 1
431|     CR = Chr$(13)
432|     TabChar = Chr$(9)
433|     Dim n As Long
434|  ' Find number of rows (delimeted by asc(13))
435|     For n = 1 To Len(ClipText)
436|    If Mid$(ClipText, n, 1) = CR Then
437|       nRows = nRows + 1
438|       nCols = nCols + 1
439|    End If
440|    If Mid$(ClipText, n, 1) = TabChar Then
441|       nCols = nCols + 1
442|    End If
443|     Next n
444|     nCols = nCols \ nRows
445|  End Sub
446|
447|  Sub lblQSSource_Change ()
448|  If lblQSSource = "CLEAR" Then
449|     grdAttributes.FixedRows = 1
450|     flgIgnoreMovement = True
451|     ClearGrid Me.grdAttributes
452|     flgIgnoreMovement = False
453|    Else
454|     SetQuery CStr(lblQSSource)
455|  End If
456|  End Sub
457|
458|  Sub lblQSType_Change ()
459|  If flgChangesWereMade Then
460|       'Present a yes/no box to the user.
461|    Me.Show
462|    nF = MsgBox("Changes were made to this data set." & CRLF & "Do you wish to save
      them now?", 36)
463|    If nF = 6 Then 'User chose to save changes
464|      Call mfSaveTableChanges_Click
465|     Else
466|      flgChangesWereMade = False
467|    End If
468|  End If
469|
470|  End Sub
471|
472|  Sub ListQrysTblsInComboBox (DBIn As Database, ComboOut As ComboBox)
473|  ' this subroutine lists all queries and tables in a database
474|  '   in a combobox, for later user selection.
475|     ComboOut.Clear
```

```
476|    Dim DB_SYSTEMOBJECT As Long
477|    DB_SYSTEMOBJECT = &H80000002
478|
479|    Dim snpQTList As Snapshot 'declared locally so it goes away
480|    Set snpQTList = DBIn.ListTables()
481|
482|    Dim QTName As String
483|    snpQTList.MoveFirst
484|    Do Until snpQTList.EOF 'For each Table and query
485|   If Not ((snpQTList("Attributes") And DB_SYSTEMOBJECT) <> 0) And
     (snpQTList("Attributes") < 6) Then
486|     'Exclude system objects from list of tables and queries presented to user
487|     QTName = snpQTList("Name")
488|     ComboOut.AddItem QTName
489|   End If
490|   snpQTList.MoveNext
491|    Loop
492|    snpQTList.Close
493|    Set snpQTList = Nothing
494| End Sub
495|
496| Sub meAddColumn_Click ()
497|  grdAttributes.Cols = grdAttributes.Cols + 1
498| End Sub
499|
500| Sub meClearGrid_Click ()
501|  flgIgnoreMovement = True
502|   grdAttributes.FixedRows = 0
503|   grdAttributes.Rows = 1
504|   grdAttributes.Rows = 2
505|   grdAttributes.FixedRows = 1
506|  flgIgnoreMovement = False
507|
508| End Sub
509|
510| Sub meColDefinition_Click ()
511|   DefineColumn
512| End Sub
513|
514| Sub meCopy_Click ()
515| Dim Current As Control
516| Set Current = ActiveControl
517| If TypeOf Current Is TextBox Then
518|    Clipboard.Clear
519|    Clipboard.SetText Current
520| End If
521| If TypeOf Current Is Grid Then
522|  flgIgnoreMovement = True
523|    Dim ClipText As String, OldRow As Long, OldCol As Long, n As Long
524|    If (Current.SelStartCol <> Current.SelEndCol) Then
525|   OldCol = Current.Col: OldRow = Current.Row
526|   Current.Row = 0
527|   For n = Current.SelStartCol To Current.SelEndCol
528|      Current.Col = n
```

```
529|       ClipText = ClipText & Current & TabChar
530|    Next n
531|     Current.Row = OldRow: Current.Col = OldCol
532|     ClipText = Left$(ClipText, Len(ClipText) - 1) & CRLF
533|    End If
534|    ClipText = ClipText & Current.Clip
535|    Clipboard.Clear
536|    Clipboard.SetText ClipText
537|    grdAttributes.FixedRows = 1
538|  flgIgnoreMovement = False
539| End If
540| End Sub
541|
542| Sub meDeleteRow_Click ()
543|
544|    Dim n As Integer, nRows As Integer
545|
546|    flgIgnoreMovement = True
547|   grdAttributes.Col = 0
548|    For n = grdAttributes.SelStartRow To grdAttributes.SelEndRow
549|   grdAttributes.Row = n
550|   If grdAttributes <> "XXX" Then grdAttributes = "D"
551|    Next n
552|    flgChangesWereMade = True
553|    flgIgnoreMovement = False
554| End Sub
555|
556| Sub meInsertRow_Click ()
557|
558|
559|   Dim n As Long
560|   flgIgnoreMovement = True
561|   For n = grdAttributes.SelStartRow To grdAttributes.SelEndRow
562|     grdAttributes.AddItem "A"   ', (ActiveControl.Row) 'Chr$(9) & Chr$(9) & CR
563|   Next n
564|   flgChangesWereMade = True
565|   flgIgnoreMovement = False
566| End Sub
567|
568| Sub mePaste_Click ()
569| Dim Current As Control
570| Set Current = ActiveControl
571| Dim ClipText As String
572| ClipText = Clipboard.GetText()
573|
574| If TypeOf Current Is TextBox Then
575|    Dim Temp As String
576|    Temp = Current.Text
577|    'Check these XXXX use frmTileOptions as example
578|    Temp = Mid$(Temp, 1, Current.SelStart) & ClipText & Mid$(Temp, Current.SelStart +
    Current.SelLength)
579|    Current.Text = Temp
580| End If
581|
```

```
582  If TypeOf Current Is Grid Then
583     Dim nRows As Long, nCols As Long
584     Dim n As Long
585  ' Find number of rows (delimeted by asc(13))
586     For n = 1 To Len(ClipText)
587     If Mid$(ClipText, n, 1) = CR Then
588        nRows = nRows + 1
589     End If
590     If Mid$(ClipText, n, 1) = TabChar Then
591        nCols = nCols + 1
592     End If
593     Next n
594
595     If nRows = 0 Then
596        nRows = 1
597     End If
598     If nCols = 0 Then
599        nCols = 1
600     End If
601     nCols = nCols \ nRows
602     If Current.SelStartCol + nCols > Current.Cols Then Current.Cols =
      Current.SelStartCol + nCols + 1
603     If Current.SelStartRow + nRows > Current.Rows Then Current.Rows =
      Current.SelStartRow + nRows + 1
604
605     Current.SelEndCol = Current.SelStartCol + nCols - 1
606     Current.SelEndRow = Current.SelStartRow + nRows - 1
607     Current.Clip = ClipText
608  End If
609     'FillGridFromClipBoardText (TextIn As String, grdOut As Grid, flgHowTo As Long)
610  End Sub
611
612  Sub meUndoThisRow_Click ()
613   If grdAttributes.Row <= nRecs Then
614     flgIgnoreMovement = True
615     Data1.Recordset.Bookmark = strBookMark(grdAttributes.Row)
616     For nF = 1 To nFields
617    grdAttributes.Col = nF
618    If FieldType(nF) < 11 Then grdAttributes = Format(grdAttributes)
619     Next nF
620     grdAttributes.Col = 0
621     grdAttributes = grdAttributes.Row
622     flgIgnoreMovement = False
623    Else
624     grdAttributes.RemoveItem grdAttributes.Row
625   End If
626  End Sub
627
628  Sub mfAbandonTableChanges_Click ()
629   flgIgnoreMovement = True
630
631   grdAttributes.Col = 0
632   For nR = 1 To grdAttributes.Rows - 1
633     grdAttributes = nR
```

```
634|   Next nR
635|   grdAttributes.Col = lngCurrentCol
636|   flgIgnoreMovement = False
637|   If lngCurrentRow < grdAttributes.Rows Then grdAttributes.Row = lngCurrentRow
638| End Sub
639|
640| Sub mfCloseDB_Click ()
641|  DB.Close
642|  Set DB = Nothing
643|   Data1.Recordset.Close
644|   cmbQTList.Clear
645|
646| End Sub
647|
648| Sub mfExit_Click ()
649|  ShutDownAndFreeMem
650| End Sub
651|
652| Sub mfNewFile_Click ()
653| Static OutFile As String, OutExt As String, DelimiterChosen As String
654|
655|   'Decide whether it should be Space, tab, or comma delimited.
656|    ReDim OZQuestion(1 To 5)  As typOZQuestion
657|    Dim Accepted As Integer
658|    OZQuestion(1).KindOfControl = CONTROL_LABEL
659|    OZQuestion(1).StringData = "What character do you want separating columns of
     data?"
660|    OZQuestion(2).KindOfControl = CONTROL_OPTIONBUTTON
661|    OZQuestion(2).StringData = "&Tab"
662|    OZQuestion(3).KindOfControl = CONTROL_OPTIONBUTTON
663|    OZQuestion(3).StringData = "&, Comma"
664|    OZQuestion(4).KindOfControl = CONTROL_OPTIONBUTTON
665|    OZQuestion(4).StringData = "&5 spaces"
666|    OZQuestion(5).KindOfControl = CONTROL_OPTIONBUTTON
667|    OZQuestion(5).StringData = "&; Semi-colon"
668| 'Consult the Wizard.
669|    AskOZ OZQuestion(), Accepted
670|    If Accepted Then
671|        If OZQuestion(2).Extra Then DelimiterChosen = Chr$(9): Me.CMDialog1.DefaultExt
     = "TDF"
672|        If OZQuestion(3).Extra Then DelimiterChosen = ",": : Me.CMDialog1.DefaultExt =
     "CSV"
673|        If OZQuestion(4).Extra Then DelimiterChosen = "     ": Me.CMDialog1.DefaultExt =
     "SDF"
674|        If OZQuestion(5).Extra Then DelimiterChosen = ";": Me.CMDialog1.DefaultExt =
     "TXT"
675|      Else
676|        Exit Sub
677|    End If
678|
679|   'Open a file
680|
681| Me.CMDialog1.DialogTitle = "Text file to receive output of current grid"
682| Me.CMDialog1.Filter = "Text|*.txt;*.tdf;*.sdf|Comma Separated Values
```

```
      (*.csv)|*.csv|Tab-delimted file(*.tdf)|*.tdf|All files (*.*)|(*.*)"
 683| Me.CMDialog1.Filename = OutFile
 684| Me.CMDialog1.Flags = OFN_OVERWRITEPROMPT Or OFN_PATHMUSTEXIST
 685|
 686| Me.CMDialog1.Action = 1
 687| Dim Temp As String
 688|
 689| OutFile = UCase$(Me.CMDialog1.Filename)
 690| If OutFile = "" Then Exit Sub
 691| Dim FileNum As Integer
 692| FileNum = FreeFile
 693| Open OutFile For Output As FileNum
 694|
 695| 'Write it out.
 696| Dim LineOut As String, Qt As String * 1, nR As Long, nC As Long, nRows As Long,
      nCols As Long
 697| nCols = grdAttributes.Cols
 698| nRows = grdAttributes.Rows
 699|
 700| Qt = Chr$(34)
 701| flgIgnoreMovement = True
 702|  grdAttributes.Row = 0
 703| 'For first line, add a SerialNumber label and put quotes around everything
 704|  LineOut = Qt & "SerialNumber" & Qt & DelimiterChosen
 705|     For nC = 1 To nCols - 1
 706|    grdAttributes.Col = nC
 707|    LineOut = LineOut & Qt & grdAttributes & Qt & DelimiterChosen
 708|     Next nC
 709|  LineOut = Left(LineOut, Len(LineOut) - Len(DelimiterChosen))'truncate extra delimiter at
      end  of line
 710|  Print #FileNum, LineOut
 711| For nR = 1 To grdAttributes.Rows - 1
 712|    grdAttributes.Row = nR
 713| 'No quotes around values here.
 714|    LineOut = nR & DelimiterChosen
 715|    For nC = 1 To nCols - 1
 716|       grdAttributes.Col = nC
 717|       LineOut = LineOut & grdAttributes & DelimiterChosen
 718|    Next nC
 719| LineOut = Left(LineOut, Len(LineOut) - Len(DelimiterChosen))'truncate extra delimiter at
      end  of line
 720| Print #FileNum, LineOut
 721| Next nR
 722| flgIgnoreMovement = False
 723|
 724| 'Close the file
 725|
 726| Close FileNum
 727|
 728| End Sub
 729|
 730| Sub mfNewTable_Click ()
 731|
 732|  Dim NewTable As New TableDef
```

```
733|    Dim TempVar As Variant
734|   Dim n As Long, nCols As Long, m As Long, nRows As Long, TableName As String
735|   Dim nC As Long, nR As Long
736|   nCols = grdAttributes.Cols
737| On Error GoTo ERRmfNewTable
738| TableName = "User Table " & Format$(MapDB.TableDefs.Count, "0000")
739| TableName = InputBox$("Enter a name for the table to be saved as.", "Saving Grid to
     Table", TableName)
740|
741| If TableName = "" Then
742|    ' Erase NewField
743|     Exit Sub
744| End If
745|   NewTable.Name = TableName
746| 'Add the serial number field.
747|   Dim LatestField As New field
748|   LatestField.Type = DB_LONG
749|   LatestField.Name = "Serial Number"
750|   NewTable.Fields.Append LatestField
751|   MapDB.TableDefs.Append NewTable
752|   Set LatestField = Nothing
753|   Set NewTable = Nothing
754|
755| 'Get the newly created table
756|  Dim tblOut As Table
757|
758|
759| 'This will take the first row of the grdAttributes grid,
760| ' and use it to define the names and types for an output table.
761| 'No provision is made for being able to append to a table
762| '   or to overwrite a table.
763| 'Command decision by Greg Pouch, 22 June 1994
764|
765|   'ReDim NewField(0 To nCols) As New Field
766|   ReDim FieldNames(0 To nCols) As String
767|
768| Dim Temp As String, tmpFieldName As String, tmpFieldType As Integer
769| Dim strFieldTypeList As String
770| flgIgnoreMovement = True
771| grdAttributes.Row = 0
772|
773|
774| For n = 1 To grdAttributes.Cols - 1
775|     grdAttributes.Col = n
776|     Temp = grdAttributes
777|     If Temp = "" Then
778|    Temp = InputBox$("Please enter a name for this field, in column " & n, "Saving Table",
     "Field" & Format$(n, "00")) & "   "
779|    grdAttributes = Left(Temp, Len(Temp) - 3)
780|     End If
781|     tmpFieldName = Left(Temp, Len(Temp) - 3)
782|     tmpFieldType = Val(Right(Temp, 3))
783|     If tmpFieldType = 0 Then
784|    'We have an untagged field, so ask the user whether
```

```
785|    'to skip it or to define the data type.
786|    ReDim OZQuestion(1 To 2)  As typOZQuestion
787|    Dim Accepted As Integer
788|    OZQuestion(1).KindOfControl = CONTROL_LABEL
789|    OZQuestion(1).StringData = "This field, " & Temp & ", has no data type associated it
       with it." & CRLF & "Choose one from the list box."
790|    OZQuestion(2).KindOfControl = CONTROL_COMBOBOX
791|    strFieldTypeList = ListOfFieldTypes
792|    OZQuestion(2).StringData = strFieldTypeList
793|    AskOZ OZQuestion(), Accepted
794|    If Accepted Then
795|      tmpFieldType = OZQuestion(2).Extra
796|      grdAttributes = grdAttributes & Format(tmpFieldType, " 00")
797|    End If
798|     End If
799|     grdAttributes = tmpFieldName & Format(tmpFieldType, " 00")
800|   If tmpFieldType <> 0 And tmpFieldName <> "" Then
801|     'if the above information exists, add the field
802|     Dim VeryNewField As New field
803|    VeryNewField.Type = tmpFieldType
804|    If tmpFieldType = 10 Then VeryNewField.Size = 255
805|    VeryNewField.Name = Trim$(tmpFieldName)
806|    MapDB.TableDefs(TableName).Fields.Append VeryNewField
807|    Set VeryNewField = Nothing
808|    FieldNames(n) = tmpFieldName
809|      Else 'Otherwise, mark it to be skipped.
810|       FieldNames(n) = ""
811|   End If
812|  Next n
813|  'At this point, all the columns have been assigned a field name,
814|  ' or not if the user doesn't want to keep it, and we should save the table
815|  ' and start writing to it.
816|   Set tblOut = MapDB.OpenTable(TableName)
817|   On Error Resume Next
818|  For nR = 1 To grdAttributes.Rows
819|     grdAttributes.Row = nR
820|     tblOut.AddNew
821|     tblOut("Serial Number") = nR
822|     For nC = 1 To nCols
823|       grdAttributes.Col = nC
824|       If (FieldNames(nC) <> "") Then
825|     TempVar = grdAttributes
826|     tblOut(FieldNames(nC)) = TempVar
827|       End If
828|     Next nC
829|     tblOut.Update
830|  Next nR
831|       Set tblOut = Nothing
832|
833|
834|  Exit Sub
835|  ERRmfNewTable:
836|  Select Case Err
837|   Case 3010 'Table of that name already exists
```

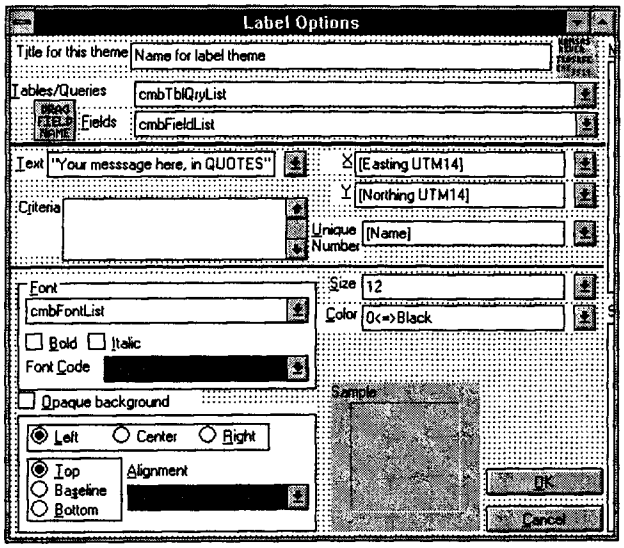
```
838|     TableName = InputBox$("Table " & TableName & " already exists.  Please choose a
       new name.", "Saving Grid to Table", "User Table" & Format$(Now))
839|    If TableName = "" Then
840|       Err = 0
841|       Exit Sub
842|       Resume Next
843|    End If
844|     NewTable.Name = TableName
845|     Resume
846|  Case Else
847|     MsgBox "Error #" & Err & " occured in frmGridEdit.mfNewTable." & CRLF & Error
848|     Err = 0
849|     Exit Sub
850|     Resume
851|  End Select
852|
853|
854|  End Sub
855|
856|  Sub mfOpenDB_Click ()
857|  Dim DBName As String' May want to change scope of this later to global.
858|  CMDialog1.DialogTitle = "Database to Open"
859|  CMDialog1.Filter = "MS Access Databases|*.mdb"
860|  CMDialog1.Flags = &H1000&    'OFN_FILEMUSTEXIST
861|  CMDialog1.Action = 1
862|  Dim Temp As String
863|
864|  Temp = CMDialog1.Filename
865|  If Len(Temp) = 0 Then 'the user did not enter a file name
866|     Else
867|      DBName = Temp
868|      ' Open an MSAccess database for non-exclusive, read-write access.
869|      Set DB = OpenDatabase(DBName, False, False, "") 'Open the desired database
870|      'Risky proposition, setting MapDB=DB, but it lets me avoid a lot of re-coding
871|      Set MapDB = DB
872|
873|      Data1.DatabaseName = DBName
874|      Me.Caption = "Grid-style data editor " & DBName
875|      Call ListQrysTblsInComboBox(DB, cmbQTList)
876|
877|      cmbQTList.SetFocus
878|      SendKeys "%{DOWN}"
879|  End If
880|
881|  End Sub
882|
883|  Sub mfSaveTableChanges_Click ()
884|     SaveGridToDynaset
885|  End Sub
886|
887|  Sub SaveGridToDynaset ()
888|  On Error GoTo ERRSaveAttributes
889|  If Not flgChangesWereMade Then Exit Sub
890|  flgIgnoreMovement = True
```

```
891|
892| 'Might want to change this so that it doesn't reset row number
893| 'This way, multiple records could be displayed and edited.
894| Dim nRows As Long, nRec As Long, Col0 As String
895| nRows = grdAttributes.Rows - 1
896| For nRec = 1 To nRows
897|    grdAttributes.Row = nRec
898| 'Examine the zeroeth column, which could contain a flag for NEW, EDITED, DELETED,
    etc.
899|    grdAttributes.Col = 0
900|    Col0 = grdAttributes
901|   If Col0 = "E" Then
902|    Data1.Recordset.Bookmark = strBookMark(nRec)
903|    Data1.Recordset.Edit
904|    For nF = 1 To nFields
905|   If FieldType(nF) < 11 Then 'don't update memo and BLOB objects, because the grid
    control can lose them due to chr(0)s
906|      grdAttributes.Col = nF
907|      Data1.Recordset.Fields(nF - 1) = grdAttributes
908|   End If
909|    Next nF
910|    Data1.Recordset.Update
911|   End If
912|
913|   If Col0 = "D" Then
914|     grdAttributes = "XXX"
915|     Data1.Recordset.Bookmark = strBookMark(nRec)
916|     Data1.Recordset.Delete
917|   End If
918|
919|   If Col0 = "A" Then
920|    grdAttributes = ""
921|    Data1.Recordset.AddNew
922|    For nF = 1 To nFields
923|   If FieldType(nF) < 11 Then 'don't update memo and BLOB objects, because the grid
    control can lose them due to chr(0)s
924|      grdAttributes.Col = nF
925|      Data1.Recordset.Fields(nF - 1) = grdAttributes
926|   End If
927|    Next nF
928|    Data1.Recordset.Update
929|   End If
930|
931| Next nRec
932|
933|
934| flgIgnoreMovement = False
935| flgChangesWereMade = False
936|
937| Exit Sub
938| ERRSaveAttributes:
939|
940| If Err = 3021 Then 'No current record
941|    Data1.Recordset.AddNew
```

```
942|     Resume Next
943| End If
944| If Err = 91 Then 'Object variable not set.  ie, no XYBLOB to update
945|   Resume Next
946| End If
947| If Err = 3058 Then
948|    grdAttributes.SelStartRow = grdAttributes.Row
949|    grdAttributes.SelEndRow = grdAttributes.Row
950|    grdAttributes.SelStartCol = 1
951|    grdAttributes.SelEndRow = grdAttributes.Cols - 1
952|    n = MsgBox("You have not entered a value in a field that is part of a primary index." &
     CRLF & "This is not allowed.  Do you want to delete this record?", 36)
953|    If n = 6 Then
954|       grdAttributes.Col = 0
955|       grdAttributes = "XXX"
956|       Resume Next
957|      Else
958|       Err = 0
959|       Exit Sub
960|       Resume Next
961|    End If
962| End If
963| If Err = 9 Then 'Subscript out of range
964|     Resume Next 'Try ignoring it
965| End If
966|
967|
968|
969|
970| MsgBox "Error number " & Err & " encountered while saving attributes." & Chr$(13) &
     Chr$(10) & Error$
971| Err = 0
972| Exit Sub
973| Resume
974|
975| End Sub
976|
977| Sub SetIgnoreMovement (ToValue As Long)
978|  flgIgnoreMovement = ToValue
979| End Sub
980|
981| Sub SetQuery (strTblQrySQL As String)
982| 'This subroutine sets the data control to the desired TaBLe, QueRY, or SQL string,
983| ' then sets the labels for the attribute grid.
984| On Error GoTo ERRSetQuery
985|
986| 'Check whether changes need to be saved.
987| If flgChangesWereMade Then
988|       'Present a yes/no/cancel box to the user.
989|    Me.Show
990|    nF = MsgBox("Changes were made to this data set." & CRLF & "Do you wish to save
     them now?", 36)
991|    If nF = 2 Then Exit Sub'user chose cancel, so clear out.
992|    If nF = 6 Then 'User chose to save changes
```

```
 993|      Call mfSaveTableChanges_Click
 994|    End If
 995|    flgChangesWereMade = False  'If the user chose yes or no, reset the flag.
 996|  End If
 997|
 998|  'Set flag to not display cell contents in text box.
 999|  flgIgnoreMovement = True
1000|
1001|  'Clear the grid
1002|  ClearGrid Me.grdAttributes
1003|
1004|  grdAttributes.RowHeight(0) = 600
1005|  'Set the data control to the desired query, table, or SQL statement.
1006|  Data1.RecordSource = strTblQrySQL
1007|  Data1.Refresh
1008|
1009|  If Data1.Recordset.Bookmarkable Then
1010|     'ReDim strBookMark(1 To nRecs)
1011|     flgBookmarkable = True
1012|     ZUnKillMenuItems
1013|   Else
1014|     mfSaveTableChanges.Enabled = False
1015|     MsgBox "You will not be able to save changes to this table or query because it cannot
      be bookmarked."
1016|     ZKillMenuItems
1017|     flgBookmarkable = False
1018|  End If
1019|
1020|
1021|  flgIgnoreMovement = False
1022|  grdAttributes.Visible = True
1023|
1024|  Exit Sub
1025|
1026|  ERRSetQuery:
1027|  If Err = 3061 Then
1028|     MsgBox "You tried selecting a parameter query, and this application doesn't do those.
      Sorry!"
1029|     Err = 0
1030|     Exit Sub
1031|  End If
1032|
1033|  MsgBox "Error number " & Err & " occurred in SetQuery." & CRLF & Error
1034|  Err = 0
1035|  'Set cursor back to normal.
1036|  mousepointer = 0
1037|  grdAttributes.Visible = True
1038|  Exit Sub
1039|  Resume
1040|  End Sub
1041|
1042|  Sub txtAttributes_GotFocus ()
1043|   If chkOverstrike Then
1044|     txtAttributes.SelStart = 0
```

```
1045|     txtAttributes.SelLength = 32765
1046|   End If
1047|
1048|  End Sub
1049|
1050|  Sub txtAttributes_KeyPress (KeyAscii As Integer)
1051|
1052|   If KeyAscii = 13 Then  'For a return key, move focus to the
1053|     KeyAscii = 0        ' grid, then save the changes to the grid.
1054|     'If Not flgBookmarkable Then
1055|     '   Beep
1056|     '   Beep
1057|     '   txtAttributes = grdAttributes
1058|     '   Exit Sub
1059|     'End If
1060|
1061|     flgIgnoreMovement = True
1062|     grdAttributes.Col = 0
1063|     If grdAttributes <> "A" Then grdAttributes = "E"
1064|     grdAttributes.Col = lngCurrentCol
1065|     flgIgnoreMovement = False
1066|     flgChangesWereMade = True
1067|     grdAttributes.SetFocus
1068|   End If
1069|   If KeyAscii = 27 Then   'For an ESCape key, reset the text box to
1070|     txtAttributes = grdAttributes    'the contents of the grid cell.
1071|     KeyAscii = 0
1072|   End If
1073|
1074|
1075|  End Sub
1076|
1077|  Sub txtAttributes_LostFocus ()
1078|     grdAttributes.Text = txtAttributes
1079|  End Sub
1080|
1081|  Sub UndoDataChanges ()
1082|   Call Data1_Reposition
1083|  End Sub
1084|
1085|  Sub ZKillMenuItems ()
1086|    zzEdit.Enabled = False
1087|    mfAbandonTableChanges.Enabled = False
1088|    mfSaveTableChanges.Enabled = False
1089|  End Sub
1090|
1091|  Sub ZUnKillMenuItems ()
1092|    zzEdit.Enabled = True
1093|    mfAbandonTableChanges.Enabled = True
1094|    mfSaveTableChanges.Enabled = True
1095|  End Sub
1096|
```


Label Options
Title for this theme
Name for label theme
Tables/Queries
cmbTblQryList
Fields
cmbFieldList
Text
"Your messsage here, in QUOTES"
X
[Easting UTM14]
Y
[Northing UTM14]
Criteria
Unique Number
[Name]
Font
cmbFontList
Bold
Italic
Font Code
Opaque background
Size
12
Color
0<=>Black
Sample
Left
Center
Right
Top
Baseline
Bottom
Alignment
OK
Cancel

```
VERSION 2.00
Begin Form frmLabelOptions
  Caption        = "Label Options"
  ClientHeight   = 5835
  ClientLeft     = 135
  ClientTop      = 480
  ClientWidth    = 6975
  ControlBox     = 0  'False
  Height         = 6240
  Icon           = LBLOPT22.FRX:0000
  Left           = 75
  LinkTopic      = "Form1"
  MinButton      = 0  'False
  ScaleHeight    = 5835
  ScaleWidth     = 6975
  Top            = 135
  Width          = 7095
  Begin ComboBox cmb_zqFontSize
    FontBold       = 0  'False
    FontItalic     = 0  'False
    FontName       = "MS Sans Serif"
    FontSize       = 8.25
    FontStrikethru = 0  'False
    FontUnderline  = 0  'False
    Height         = 300
    HelpContextID  = 2500
    Left           = 4050
    TabIndex       = 35
    Text           = "12"
    Top            = 2790
    Width          = 2715
  End
  Begin ComboBox cmb_zqColor
    FontBold       = 0  'False
    FontItalic     = 0  'False
    FontName       = "MS Sans Serif"
    FontSize       = 8.25
    FontStrikethru = 0  'False
    FontUnderline  = 0  'False
    Height         = 300
    HelpContextID  = 2000
    Left           = 4050
    TabIndex       = 37
    Text           = "0<=>Black"
    Top            = 3150
    Width          = 2715
  End
  Begin ComboBox cmb_zqLabelText
    FontBold       = 0  'False
    FontItalic     = 0  'False
    FontName       = "MS Sans Serif"
    FontSize       = 8.25
    FontStrikethru = 0  'False
    FontUnderline  = 0  'False
```

```
   Height      = 300
   HelpContextID  = 2600
   Left        = 420
   TabIndex    = 7
   Text        = """Your messsage here, in QUOTES"""
   Top         = 1350
   Width       = 2985
End
Begin ComboBox cmb_zqX
   FontBold    = 0  'False
   FontItalic  = 0  'False
   FontName    = "MS Sans Serif"
   FontSize    = 8.25
   FontStrikethru = 0  'False
   FontUnderline  = 0  'False
   Height      = 300
   HelpContextID  = 1000
   Left        = 3960
   TabIndex    = 9
   Text        = "[Easting UTM14]"
   Top         = 1350
   Width       = 2805
End
Begin ComboBox cmb_zqY
   FontBold    = 0  'False
   FontItalic  = 0  'False
   FontName    = "MS Sans Serif"
   FontSize    = 8.25
   FontStrikethru = 0  'False
   FontUnderline  = 0  'False
   Height      = 300
   HelpContextID  = 1100
   Left        = 3960
   TabIndex    = 11
   Text        = "[Northing UTM14]"
   Top         = 1710
   Width       = 2805
End
Begin ComboBox cmb_zqFeatureID
   FontBold    = 0  'False
   FontItalic  = 0  'False
   FontName    = "MS Sans Serif"
   FontSize    = 8.25
   FontStrikethru = 0  'False
   FontUnderline  = 0  'False
   Height      = 300
   HelpContextID  = 1500
   Left        = 4050
   TabIndex    = 13
   Text        = "[Name]"
   Top         = 2160
   Width       = 2715
End
Begin SSPanel zpnlOuter
```

```
109|     Alignment      = 0 'Left Justify - TOP
110|     AutoSize       = 3 'AutoSize Child To Panel
111|     BackColor      = &H00C0C0C0&
112|     BevelInner     = 1 'Inset
113|     BevelOuter     = 0 'None
114|     BorderWidth    = 15
115|     Font3D         = 0 'None
116|     FontBold       = 0 'False
117|     FontItalic     = 0 'False
118|     FontName       = "MS Sans Serif"
119|     FontSize       = 8.25
120|     FontStrikethru = 0 'False
121|     FontUnderline  = 0 'False
122|     ForeColor      = &H00FFFFFF&
123|     Height         = 1725
124|     Left           = 3690
125|     TabIndex       = 44
126|     Top            = 4050
127|     Width          = 1725
128|     Begin PictureBox picSample
129|       BackColor    = &H00C0C0C0&
130|       BorderStyle  = 0 'None
131|       Height       = 1250
132|       Left         = 240
133|       ScaleHeight  = 1245
134|       ScaleWidth   = 1245
135|       TabIndex     = 46
136|       Top          = 240
137|       Width        = 1250
138|     End
139|     Begin SSPanel zpnlSampleText
140|       Alignment    = 0 'Left Justify - TOP
141|       AutoSize     = 3 'AutoSize Child To Panel
142|       BackColor    = &H00C0C0C0&
143|       BevelOuter   = 0 'None
144|       BorderWidth  = 1
145|       Caption      = "Sample"
146|       Font3D       = 0 'None
147|       FontBold     = 0 'False
148|       FontItalic   = 0 'False
149|       FontName     = "MS Sans Serif"
150|       FontSize     = 8.25
151|       FontStrikethru = 0 'False
152|       FontUnderline = 0 'False
153|       ForeColor    = &H00000000&
154|       Height       = 220
155|       Left         = 0
156|       TabIndex     = 45
157|       Top          = 0
158|       Width        = 735
159|     End
160|   End
161|   Begin CheckBox chkOpaque
162|     Caption        = "&Opaque background"
```

```
163|     FontBold      = 0  'False
164|     FontItalic    = 0  'False
165|     FontName      = "MS Sans Serif"
166|     FontSize      = 8.25
167|     FontStrikethru = 0  'False
168|     FontUnderline = 0  'False
169|     Height        = 225
170|     Left          = 90
171|     TabIndex      = 22
172|     Top           = 4125
173|     Visible       = 0  'False
174|     Width         = 2085
175|   End
176|   Begin TextBox txtThemeName
177|     FontBold      = 0  'False
178|     FontItalic    = 0  'False
179|     FontName      = "MS Sans Serif"
180|     FontSize      = 8.25
181|     FontStrikethru = 0  'False
182|     FontUnderline = 0  'False
183|     Height        = 315
184|     Left          = 1380
185|     TabIndex      = 1
186|     Text          = "Name for label theme"
187|     Top           = 90
188|     Width         = 4860
189|   End
190|   Begin TextBox txtCriteria
191|     FontBold      = 0  'False
192|     FontItalic    = 0  'False
193|     FontName      = "MS Sans Serif"
194|     FontSize      = 8.25
195|     FontStrikethru = 0  'False
196|     FontUnderline = 0  'False
197|     Height        = 735
198|     HelpContextID = 1600
199|     Left          = 600
200|     MultiLine     = -1 'True
201|     ScrollBars    = 2 'Vertical
202|     TabIndex      = 15
203|     Top           = 1890
204|     Width         = 2820
205|   End
206|   Begin TextBox txtSpatialQuerySQL
207|     FontBold      = 0  'False
208|     FontItalic    = 0  'False
209|     FontName      = "MS Sans Serif"
210|     FontSize      = 8.25
211|     FontStrikethru = 0  'False
212|     FontUnderline = 0  'False
213|     Height        = 2430
214|     Left          = 6900
215|     MultiLine     = -1 'True
216|     ScrollBars    = 2 'Vertical
```

```
217|     TabIndex      = 43
218|     Text          = "SQL Statement to produce data displayed on DataBase Form"
219|     Top           = 3360
220|     Width         = 2475
221|   End
222|   Begin TextBox txtMapSQL
223|     FontBold      = 0  'False
224|     FontItalic    = 0  'False
225|     FontName      = "MS Sans Serif"
226|     FontSize      = 8.25
227|     FontStrikethru = 0  'False
228|     FontUnderline = 0  'False
229|     Height        = 2715
230|     Left          = 6900
231|     MultiLine     = -1 'True
232|     ScrollBars    = 2 'Vertical
233|     TabIndex      = 41
234|     Text          = "SQL statement to produce map data"
235|     Top           = 300
236|     Width         = 2445
237|   End
238|   Begin ComboBox cmbTblQryList
239|     DragIcon      = LBLOPT22.FRX:0302
240|     FontBold      = 0  'False
241|     FontItalic    = 0  'False
242|     FontName      = "MS Sans Serif"
243|     FontSize      = 8.25
244|     FontStrikethru = 0  'False
245|     FontUnderline = 0  'False
246|     Height        = 300
247|     Left          = 1425
248|     Style         = 2 'Dropdown List
249|     TabIndex      = 3
250|     Top           = 540
251|     Width         = 5340
252|   End
253|   Begin CommandButton cmdCancel
254|     Caption       = "&Cancel"
255|     FontBold      = 0  'False
256|     FontItalic    = 0  'False
257|     FontName      = "MS Sans Serif"
258|     FontSize      = 8.25
259|     FontStrikethru = 0  'False
260|     FontUnderline = 0  'False
261|     Height        = 315
262|     Left          = 5490
263|     TabIndex      = 39
264|     Top           = 5475
265|     Width         = 1365
266|   End
267|   Begin CommandButton cmdOK
268|     Caption       = "&OK"
269|     FontBold      = 0  'False
270|     FontItalic    = 0  'False
```

```
271|     FontName      = "MS Sans Serif"
272|     FontSize      = 8.25
273|     FontStrikethru = 0  'False
274|     FontUnderline  = 0  'False
275|     Height        = 315
276|     Left          = 5490
277|     TabIndex      = 38
278|     Top           = 5040
279|     Width         = 1365
280|   End
281|   Begin ComboBox cmbFieldList
282|     DragIcon      = LBLOPT22.FRX:0604
283|     FontBold      = 0  'False
284|     FontItalic    = 0  'False
285|     FontName      = "MS Sans Serif"
286|     FontSize      = 8.25
287|     FontStrikethru = 0  'False
288|     FontUnderline  = 0  'False
289|     Height        = 300
290|     Left          = 1425
291|     Sorted        = -1 'True
292|     Style         = 2 'Dropdown List
293|     TabIndex      = 5
294|     Top           = 885
295|     Width         = 5355
296|   End
297|   Begin Frame zzzframeFont
298|     Caption       = "&Font"
299|     FontBold      = 0  'False
300|     FontItalic    = 0  'False
301|     FontName      = "MS Sans Serif"
302|     FontSize      = 8.25
303|     FontStrikethru = 0  'False
304|     FontUnderline  = 0  'False
305|     Height        = 1335
306|     HelpContextID  = 2400
307|     Left          = 90
308|     TabIndex      = 16
309|     Top           = 2790
310|     Width         = 3435
311|     Begin ComboBox cmb_zqFontCode
312|       BackColor     = &H000000FF&
313|       FontBold      = 0  'False
314|       FontItalic    = 0  'False
315|       FontName      = "MS Sans Serif"
316|       FontSize      = 8.25
317|       FontStrikethru = 0  'False
318|       FontUnderline  = 0  'False
319|       ForeColor     = &H00000000&
320|       Height        = 300
321|       HelpContextID  = 2400
322|       Left          = 990
323|       TabIndex      = 21
324|       Text          = "FontCode"
```

```
325|       Top          = 950
326|       Width        = 2355
327|     End
328|     Begin CheckBox chkFontItalic
329|       Caption      = "&Italic"
330|       FontBold     = 0  'False
331|       FontItalic   = 0  'False
332|       FontName     = "MS Sans Serif"
333|       FontSize     = 8.25
334|       FontStrikethru = 0  'False
335|       FontUnderline = 0  'False
336|       Height       = 285
337|       HelpContextID = 2400
338|       Left         = 810
339|       TabIndex     = 19
340|       Top          = 630
341|       Width        = 735
342|     End
343|     Begin CheckBox chkFontBold
344|       Caption      = "&Bold"
345|       FontBold     = 0  'False
346|       FontItalic   = 0  'False
347|       FontName     = "MS Sans Serif"
348|       FontSize     = 8.25
349|       FontStrikethru = 0  'False
350|       FontUnderline = 0  'False
351|       Height       = 285
352|       HelpContextID = 2400
353|       Left         = 90
354|       TabIndex     = 18
355|       Top          = 630
356|       Width        = 735
357|     End
358|     Begin ComboBox cmbFontList
359|       FontBold     = 0  'False
360|       FontItalic   = 0  'False
361|       FontName     = "MS Sans Serif"
362|       FontSize     = 8.25
363|       FontStrikethru = 0  'False
364|       FontUnderline = 0  'False
365|       Height       = 300
366|       HelpContextID = 2400
367|       Left         = 90
368|       Style        = 2  'Dropdown List
369|       TabIndex     = 17
370|       Top          = 270
371|       Width        = 3255
372|     End
373|     Begin Label zzlblFontCode
374|       Caption      = "Font &Code"
375|       FontBold     = 0  'False
376|       FontItalic   = 0  'False
377|       FontName     = "MS Sans Serif"
378|       FontSize     = 8.25
```

```
379|       FontStrikethru = 0  'False
380|       FontUnderline  = 0  'False
381|       Height         = 195
382|       Left           = 90
383|       TabIndex       = 20
384|       Top            = 960
385|       Width          = 915
386|     End
387|   End
388|   Begin Frame zzzframeAlign
389|     FontBold       = 0  'False
390|     FontItalic     = 0  'False
391|     FontName       = "MS Sans Serif"
392|     FontSize       = 8.25
393|     FontStrikethru = 0  'False
394|     FontUnderline  = 0  'False
395|     Height         = 1455
396|     HelpContextID  = 2510
397|     Left           = 90
398|     TabIndex       = 23
399|     Top            = 4320
400|     Width          = 3435
401|     Begin ComboBox cmb_zqAlignment
402|       BackColor    = &H000000FF&
403|       FontBold     = 0  'False
404|       FontItalic   = 0  'False
405|       FontName     = "MS Sans Serif"
406|       FontSize     = 8.25
407|       FontStrikethru = 0  'False
408|       FontUnderline  = 0  'False
409|       ForeColor    = &H00000000&
410|       Height       = 300
411|       Left         = 1260
412|       TabIndex     = 33
413|       Text         = "0<=>Left_Top"
414|       Top          = 900
415|       Width        = 2085
416|     End
417|     Begin Frame zzzframeHorzAlign
418|       Height       = 405
419|       HelpContextID  = 2510
420|       Left         = 100
421|       TabIndex     = 24
422|       Top          = 100
423|       Width        = 2790
424|       Begin OptionButton optRight
425|         Caption    = "&Right"
426|         FontBold   = 0  'False
427|         FontItalic = 0  'False
428|         FontName   = "MS Sans Serif"
429|         FontSize   = 8.25
430|         FontStrikethru = 0  'False
431|         FontUnderline  = 0  'False
432|         Height     = 210
```

```
433|         HelpContextID  =  2510
434|         Left         =  1980
435|         TabIndex      =  27
436|         Top          =  120
437|         Width        =  735
438|       End
439|       Begin OptionButton optCenter
440|         Caption      =  "&Center"
441|         FontBold     =  0  'False
442|         FontItalic   =  0  'False
443|         FontName     =  "MS Sans Serif"
444|         FontSize     =  8.25
445|         FontStrikethru =  0  'False
446|         FontUnderline  =  0  'False
447|         Height       =  195
448|         HelpContextID  =  2510
449|         Left         =  990
450|         TabIndex      =  26
451|         Top          =  120
452|         Width        =  825
453|       End
454|       Begin OptionButton optLeft
455|         Caption      =  "&Left"
456|         FontBold     =  0  'False
457|         FontItalic   =  0  'False
458|         FontName     =  "MS Sans Serif"
459|         FontSize     =  8.25
460|         FontStrikethru =  0  'False
461|         FontUnderline  =  0  'False
462|         Height       =  210
463|         HelpContextID  =  2510
464|         Left         =  45
465|         TabIndex      =  25
466|         Top          =  120
467|         Value        =  -1  'True
468|         Width        =  600
469|       End
470|     End
471|     Begin Frame zzzframeVertAlign
472|       Height       =  825
473|       HelpContextID  =  2510
474|       Left         =  100
475|       TabIndex      =  28
476|       Top          =  500
477|       Width        =  1095
478|       Begin OptionButton optBottom
479|         Caption      =  "&Bottom"
480|         FontBold     =  0  'False
481|         FontItalic   =  0  'False
482|         FontName     =  "MS Sans Serif"
483|         FontSize     =  8.25
484|         FontStrikethru =  0  'False
485|         FontUnderline  =  0  'False
486|         Height       =  210
```

```
487|         HelpContextID  =  2510
488|         Left         =  45
489|         TabIndex      =  31
490|         Top          =  575
491|         Width        =  915
492|       End
493|       Begin OptionButton optBaseline
494|         Caption      =  "Ba&seline"
495|         FontBold     =  0  'False
496|         FontItalic   =  0  'False
497|         FontName     =  "MS Sans Serif"
498|         FontSize     =  8.25
499|         FontStrikethru =  0  'False
500|         FontUnderline  =  0  'False
501|         Height       =  210
502|         HelpContextID  =  2510
503|         Left         =  45
504|         TabIndex      =  30
505|         Top          =  345
506|         Width        =  1000
507|       End
508|       Begin OptionButton optTop
509|         Caption      =  "&Top"
510|         FontBold     =  0  'False
511|         FontItalic   =  0  'False
512|         FontName     =  "MS Sans Serif"
513|         FontSize     =  8.25
514|         FontStrikethru =  0  'False
515|         FontUnderline  =  0  'False
516|         Height       =  210
517|         HelpContextID  =  2510
518|         Left         =  45
519|         TabIndex      =  29
520|         Top          =  120
521|         Value        =  -1  'True
522|         Width        =  645
523|       End
524|     End
525|     Begin Label zzlblTextAlign
526|       Caption      =  "&Alignment"
527|       FontBold     =  0  'False
528|       FontItalic   =  0  'False
529|       FontName     =  "MS Sans Serif"
530|       FontSize     =  8.25
531|       FontStrikethru =  0  'False
532|       FontUnderline  =  0  'False
533|       Height       =  285
534|       Left         =  1260
535|       TabIndex      =  32
536|       Top          =  630
537|       Width        =  915
538|     End
539|   End
540|   Begin Label ACTION
```

```
541|     BackColor      = &H0000FFFF&
542|     ForeColor      = &H000000FF&
543|     Height         = 195
544|     Left           = 720
545|     TabIndex       = 47
546|     Top            = 2160
547|     Visible        = 0  'False
548|     Width          = 1995
549|   End
550|   Begin Image imgFieldName
551|     Height         = 480
552|     Left           = 270
553|     Picture        = LBLOPT22.FRX:0906
554|     Top            = 750
555|     Width          = 480
556|   End
557|   Begin Label zzlblSize
558|     Caption        = "&Size"
559|     FontBold       = 0  'False
560|     FontItalic     = 0  'False
561|     FontName       = "MS Sans Serif"
562|     FontSize       = 8.25
563|     FontStrikethru = 0  'False
564|     FontUnderline  = 0  'False
565|     Height         = 285
566|     Left           = 3682
567|     TabIndex       = 34
568|     Top            = 2790
569|     Width          = 375
570|   End
571|   Begin Line Line2
572|     BorderWidth    = 2
573|     X1             = 6840
574|     X2             = 0
575|     Y1             = 2700
576|     Y2             = 2700
577|   End
578|   Begin Line Line1
579|     BorderWidth    = 3
580|     X1             = 0
581|     X2             = 6840
582|     Y1             = 1260
583|     Y2             = 1260
584|   End
585|   Begin Label zzlblText
586|     Caption        = "&Text"
587|     FontBold       = 0  'False
588|     FontItalic     = 0  'False
589|     FontName       = "MS Sans Serif"
590|     FontSize       = 8.25
591|     FontStrikethru = 0  'False
592|     FontUnderline  = 0  'False
593|     Height         = 285
594|     Left           = 60
```

```
595|     TabIndex      = 6
596|     Top           = 1380
597|     Width         = 555
598|   End
599|   Begin Image zIMGExample
600|     Height        = 480
601|     Left          = 6300
602|     Picture       = LBLOPT22.FRX:0C08
603|     Top           = 15
604|     Width         = 480
605|   End
606|   Begin Label lblThemeName
607|     Caption       = "T&itle for this theme"
608|     FontBold      = 0  'False
609|     FontItalic    = 0  'False
610|     FontName      = "MS Sans Serif"
611|     FontSize      = 8.25
612|     FontStrikethru = 0  'False
613|     FontUnderline = 0  'False
614|     Height        = 225
615|     Left          = 60
616|     TabIndex      = 0
617|     Top           = 90
618|     Width         = 1740
619|   End
620|   Begin Label zzlblColor
621|     Alignment     = 2  'Center
622|     Caption       = "&Color"
623|     FontBold      = 0  'False
624|     FontItalic    = 0  'False
625|     FontName      = "MS Sans Serif"
626|     FontSize      = 8.25
627|     FontStrikethru = 0  'False
628|     FontUnderline = 0  'False
629|     Height        = 285
630|     Left          = 3510
631|     TabIndex      = 36
632|     Top           = 3150
633|     Width         = 615
634|   End
635|   Begin Label zzzzlblCriteia
636|     Caption       = "C&riteria"
637|     FontBold      = 0  'False
638|     FontItalic    = 0  'False
639|     FontName      = "MS Sans Serif"
640|     FontSize      = 8.25
641|     FontStrikethru = 0  'False
642|     FontUnderline = 0  'False
643|     Height        = 645
644|     Left          = 90
645|     TabIndex      = 14
646|     Top           = 1890
647|     Width         = 615
648|   End
```

```
649|   Begin Label zzzzlblDBFormSQL
650|     Caption       = "Spatial Query SQL"
651|     FontBold      = 0  'False
652|     FontItalic    = 0  'False
653|     FontName      = "MS Sans Serif"
654|     FontSize      = 8.25
655|     FontStrikethru = 0  'False
656|     FontUnderline = 0  'False
657|     Height        = 465
658|     Left          = 6900
659|     TabIndex      = 42
660|     Top           = 3120
661|     Width         = 1500
662|   End
663|   Begin Label zzzzlblMapTileSQL
664|     Caption       = "&Map Label SQL"
665|     FontBold      = 0  'False
666|     FontItalic    = 0  'False
667|     FontName      = "MS Sans Serif"
668|     FontSize      = 8.25
669|     FontStrikethru = 0  'False
670|     FontUnderline = 0  'False
671|     Height        = 465
672|     Left          = 6900
673|     TabIndex      = 40
674|     Top           = 60
675|     Width         = 1500
676|   End
677|   Begin Label zzzlblFields
678|     Caption       = "&Fields"
679|     FontBold      = 0  'False
680|     FontItalic    = 0  'False
681|     FontName      = "MS Sans Serif"
682|     FontSize      = 8.25
683|     FontStrikethru = 0  'False
684|     FontUnderline = 0  'False
685|     Height        = 285
686|     Left          = 810
687|     TabIndex      = 4
688|     Top           = 900
689|     Width         = 555
690|   End
691|   Begin Label zzzlblTables
692|     Caption       = "&Tables/Queries"
693|     DragIcon      = LBLOPT22.FRX:0F0A
694|     DragMode      = 1  'Automatic
695|     FontBold      = 0  'False
696|     FontItalic    = 0  'False
697|     FontName      = "MS Sans Serif"
698|     FontSize      = 8.25
699|     FontStrikethru = 0  'False
700|     FontUnderline = 0  'False
701|     Height        = 285
702|     Left          = 0
```

```
703|     TabIndex      = 2
704|     Top           = 540
705|     Width         = 1650
706|   End
707|   Begin Label zzzlblName
708|     Caption       = "&Unique Number"
709|     FontBold      = 0  'False
710|     FontItalic    = 0  'False
711|     FontName      = "MS Sans Serif"
712|     FontSize      = 8.25
713|     FontStrikethru = 0  'False
714|     FontUnderline = 0  'False
715|     Height        = 435
716|     Left          = 3465
717|     TabIndex      = 12
718|     Top           = 2160
719|     Width         = 615
720|   End
721|   Begin Label zzzlblY
722|     Alignment     = 1  'Right Justify
723|     Caption       = "&Y"
724|     FontBold      = 0  'False
725|     FontItalic    = 0  'False
726|     FontName      = "MS Sans Serif"
727|     FontSize      = 8.25
728|     FontStrikethru = 0  'False
729|     FontUnderline = 0  'False
730|     Height        = 255
731|     Left          = 3690
732|     TabIndex      = 10
733|     Top           = 1710
734|     Width         = 225
735|   End
736|   Begin Label zzzlblX
737|     Alignment     = 1  'Right Justify
738|     Caption       = "&X"
739|     FontBold      = 0  'False
740|     FontItalic    = 0  'False
741|     FontName      = "MS Sans Serif"
742|     FontSize      = 8.25
743|     FontStrikethru = 0  'False
744|     FontUnderline = 0  'False
745|     Height        = 255
746|     Left          = 3600
747|     TabIndex      = 8
748|     Top           = 1350
749|     Width         = 330
750|   End
751| End
752| Option Explicit
753| Option Compare Text
754| Dim flgValidSyntax As Integer
755| Dim ListOfFieldsForDBForm As String
756| Dim FlagActivateDragDrop As Integer
```

```
Dim tblLabelMap As Table
Dim FieldsInCurrentQT() As String, FieldListWBrackets() As String
Dim nStoredMaps As Long
Dim intTextAlign As Integer, intFontCode As Integer

Dim flgIgnoreMapSQLChange As Long

Sub ACTION_Change ()
  Dim tmp As String
  tmp = UCase$(Me!ACTION.Caption)
  Select Case tmp
   Case "Decode MapSQL"
      tmp = Me!txtMapSQL
      Call DecodeMapSQL(tmp)
   Case "wait" 'do nothing
   Case "refresh"
      Call Form_Load
   Case Else
  End Select
End Sub

Sub chkFontBold_Click ()
 UpdateFontCode

End Sub

Sub chkFontItalic_Click ()
 UpdateFontCode
End Sub

Sub chkOpaque_Click ()
 SetMapSQL
End Sub

Sub cmb_zqAlignment_Change ()
   SetMapSQL
End Sub

Sub cmb_zqAlignment_Click ()
 SetMapSQL
End Sub

Sub cmb_zqAlignment_DragDrop (Source As Control, X As Single, Y As Single)
  cmb_zqAlignment.AddItem Mid$(cmb_zqAlignment, 1, cmb_zqAlignment.SelStart) & "
[" & CStr(Source) & "] " & Mid$(cmb_zqAlignment, cmb_zqAlignment.SelStart +
cmb_zqAlignment.SelLength + 1, 32768)
  cmb_zqAlignment.ListIndex = cmb_zqAlignment.ListCount - 1
  'Drops the selected text at the current insertion point

End Sub

Sub cmb_zqColor_Change ()
 SetMapSQL
 Dim TempColor As Long
```

```
 809|  TempColor = Val(cmb_zqColor.Text)
 810|  picSample.ForeColor = TempColor
 811| End Sub
 812|
 813| Sub cmb_zqColor_Click ()
 814|  SetMapSQL
 815| End Sub
 816|
 817| Sub cmb_zqColor_DragDrop (Source As Control, X As Single, Y As Single)
 818|  cmb_zqColor.AddItem Mid$(cmb_zqColor, 1, cmb_zqColor.SelStart) & " " & "[" &
      Source & "]" & " " & Mid$(cmb_zqColor, cmb_zqColor.SelStart +
      cmb_zqColor.SelLength + 1, 32768)
 819|  cmb_zqColor.ListIndex = cmb_zqColor.ListCount - 1
 820| 'Drops the selected text at the current insertion point
 821|
 822| End Sub
 823|
 824| Sub cmb_zqFeatureID_Change ()
 825|
 826|  SetMapSQL
 827|  Call DBFormSQLAddField("")
 828| End Sub
 829|
 830| Sub cmb_zqFeatureID_Click ()
 831|   SetMapSQL
 832|   Call DBFormSQLAddField("")
 833| End Sub
 834|
 835| Sub cmb_zqFeatureID_DragDrop (Source As Control, X As Single, Y As Single)
 836|
 837|   cmb_zqFeatureID.AddItem "[" & Source & "]"
 838|   cmb_zqFeatureID.ListIndex = cmb_zqFeatureID.ListCount - 1
 839|
 840| 'Drops the selected text over the entire field, for use on
 841| ' NAME only, because we'll need this for matching and it
 842| ' NAME should never be a calculated field, at least not easily.
 843|
 844| End Sub
 845|
 846| Sub cmb_zqFontCode_Change ()
 847|    SetMapSQL
 848|  Dim i As Integer
 849|
 850|  i = Val(Me!cmb_zqFontCode.Text)
 851|   Me!cmbFontList.ListIndex = IndexOfStringInCOMBOBOX(FontNameFromID(i),
     Me!cmbFontList)
 852|
 853|
 854| End Sub
 855|
 856| Sub cmb_zqFontCode_Click ()
 857|  SetMapSQL
 858|  Dim i As Integer
 859| On Error Resume Next
```

```
860|   i = Me!cmb_zqFontCode.Text
861|   i = IndexOfStringInCOMBOBOX(FontNameFromID(i), Me!cmbFontList)
862|    Me!cmbFontList.ListIndex = i
863| End Sub
864|
865| Sub cmb_zqFontCode_DragDrop (Source As Control, X As Single, Y As Single)
866| cmb_zqFontCode.AddItem Mid$(cmb_zqFontCode, 1, cmb_zqFontCode.SelStart) & " ["
     & CStr(Source) & "] " & Mid$(cmb_zqFontCode, cmb_zqFontCode.SelStart +
     cmb_zqFontCode.SelLength + 1, 32768)
867| cmb_zqFontCode.ListIndex = cmb_zqFontCode.ListCount - 1
868| 'Drops the selected text at the current insertion point
869|
870| End Sub
871|
872| Sub cmb_zqFontSize_Change ()
873|  SetMapSQL
874|  Dim TempSize As Single
875|  TempSize = Val(cmb_zqFontSize.Text)
876|  If TempSize <> 0 Then picSample.FontSize = TempSize
877|  UpdateFontCode
878|  ShowSample
879| End Sub
880|
881| Sub cmb_zqFontSize_Click ()
882|  SetMapSQL
883| End Sub
884|
885| Sub cmb_zqFontSize_DragDrop (Source As Control, X As Single, Y As Single)
886|  cmb_zqFontSize.AddItem Mid$(cmb_zqFontSize, 1, cmb_zqFontSize.SelStart) & " " &
     "[" & Source & "]" & " " & Mid$(cmb_zqFontSize, cmb_zqFontSize.SelStart +
     cmb_zqFontSize.SelLength + 1, 32768)
887|  cmb_zqFontSize.ListIndex = cmb_zqFontSize.ListCount - 1
888| 'Drops the selected text at the current insertion point
889| End Sub
890|
891| Sub cmb_zqLabelText_Change ()
892|  SetMapSQL
893|  ShowSample
894| End Sub
895|
896| Sub cmb_zqLabelText_Click ()
897|  SetMapSQL
898| End Sub
899|
900| Sub cmb_zqLabelText_DragDrop (Source As Control, X As Single, Y As Single)
901|
902|  cmb_zqLabelText.AddItem Mid$(cmb_zqLabelText, 1, cmb_zqLabelText.SelStart) & " ["
     & CStr(Source) & "] " & Mid$(cmb_zqLabelText, cmb_zqLabelText.SelStart +
     cmb_zqLabelText.SelLength + 1, 32768)
903|  cmb_zqLabelText.ListIndex = cmb_zqLabelText.ListCount - 1
904|
905| End Sub
906|
907| Sub cmb_zqX_Change ()
```

```
908| SetMapSQL
909| End Sub
910|
911| Sub cmb_zqX_Click ()
912|    SetMapSQL
913| End Sub
914|
915| Sub cmb_zqX_DragDrop (Source As Control, X As Single, Y As Single)
916| Me!cmb_zqX.AddItem Mid$(Me!cmb_zqX, 1, Me!cmb_zqX.SelStart) & " [" &
     CStr(Source) & "] " & Mid$(Me!cmb_zqX, Me!cmb_zqX.SelStart +
     Me!cmb_zqX.SelLength + 1, 32768)
917| Me!cmb_zqX.ListIndex = Me!cmb_zqX.ListCount
918| 'Drops the selected text at the current insertion point
919| End Sub
920|
921| Sub cmb_zqY_Change ()
922|  SetMapSQL
923| End Sub
924|
925| Sub cmb_zqY_Click ()
926|  SetMapSQL
927| End Sub
928|
929| Sub cmb_zqY_DragDrop (Source As Control, X As Single, Y As Single)
930| 'Y = Mid$(Y, 1, Y.SelStart) & " " & AddBracketsAsNeeded(CStr(Source)) & " " & Mid$(Y,
     Y.SelStart + Y.SelLength + 1, 32768)
931| Me!cmb_zqY.AddItem Mid$(Me!cmb_zqY, 1, Me!cmb_zqY.SelStart) & " [" &
     CStr(Source) & "] " & Mid$(Me!cmb_zqY, Me!cmb_zqY.SelStart +
     Me!cmb_zqY.SelLength + 1, 32768)
932| Me!cmb_zqY.ListIndex = Me!cmb_zqY.ListCount - 1
933| 'Drops the selected text at the current insertion point
934| End Sub
935|
936| Sub cmbFieldList_Click ()
937| 'If FlagActivateDragDrop <> 0 Then
938| '   cmbFieldList.Drag
939| '  Else
940| '    FlagActivateDragDrop = 1
941| 'End If
942| End Sub
943|
944| Sub cmbFieldList_KeyDown (KeyCode As Integer, Shift As Integer)
945|   If KeyCode = KEY_INSERT And Shift = 2 Then 'CTRL-Insert?
946|     clipboard.Clear
947|     clipboard.SetText ("[" & cmbFieldList & "]")
948|   End If
949| End Sub
950|
951| Sub cmbFontList_Click ()
952|  'cmb_zqFontCode = cmbFontList.Index
953|  UpdateFontCode
954| End Sub
955|
956| Sub cmbTblQryList_Click ()
```

```
 957| On Error GoTo ERRcmbTblQryList_Click
 958|
 959| If Me!ACTION.Caption = "wait" Then Exit Sub
 960|  If Not flgIgnoreMapSQLChange Then
 961|    'For usual case, set the SpatialQuerySQL and the theme name
 962|    txtSpatialQuerySQL = "SELECT * FROM " &
      AddBracketsAsNeeded(CStr(cmbTblQryList))
 963|    ListOfFieldsForDBForm = "*"
 964|    txtThemeName = "LABELS: " & cmbTblQryList & "  " & Now
 965|  End If
 966| Dim TableName As String
 967|
 968|  TableName = cmbTblQryList
 969|  Set snpFieldList = MapDB.CreateSnapshot(TableName)
 970|  nFields = snpFieldList.Fields.Count
 971|  cmbFieldList.Clear
 972|  ReDim FieldsInCurrentQT(1 To nFields), FieldListWBrackets(1 To nFields)
 973|
 974|  For i = 1 To nFields
 975|    cmbFieldList.AddItem snpFieldList.Fields(i - 1).Name
 976|    FieldsInCurrentQT(i) = snpFieldList.Fields(i - 1).Name
 977|    FieldListWBrackets(i) = "[" & snpFieldList.Fields(i - 1).Name & "]"
 978|  Next i
 979|
 980|  txtCriteria = ""
 981|  snpFieldList.Close
 982|  cmbFieldList.ListIndex = 0
 983|  FlagActivateDragDrop = 0
 984|
 985| cmb_zqX.Clear
 986| FillCMBFromSGSTLST FieldListWBrackets(), XLookFor, XShowAlways, cmb_zqX
 987| cmb_zqX.ListIndex = 0
 988|
 989| cmb_zqY.Clear
 990| FillCMBFromSGSTLST FieldListWBrackets(), YLookFor, YShowAlways, cmb_zqY
 991| cmb_zqY.ListIndex = 0
 992|
 993| cmb_zqFeatureID.Clear
 994| FillCMBFromSGSTLST FieldListWBrackets(), FeatureNameLookFor,
      FeatureNameShowAlways, cmb_zqFeatureID
 995| cmb_zqFeatureID.ListIndex = 0
 996|
 997| cmb_zqLabelText.Clear
 998| FillCMBFromSGSTLST FieldListWBrackets(), FeatureTextLookFor,
      FeatureTextShowAlways, cmb_zqLabelText
 999| cmb_zqLabelText.ListIndex = 0
1000|
1001| cmb_zqAlignment.Clear
1002| FillCMBFromSGSTLST FieldListWBrackets(), AlignmentLookFor,
      AlignmentShowAlways, cmb_zqAlignment
1003| cmb_zqAlignment.ListIndex = 0
1004|
1005|
1006| cmb_zqFontSize.Clear
```

```
1007|  FillCMBFromSGSTLST FieldListWBrackets(), SizeLookFor, SizeShowAlways,
       cmb_zqFontSize
1008|  cmb_zqFontSize.ListIndex = 0
1009|
1010|  cmb_zqColor.Clear
1011|  FillCMBFromSGSTLST FieldListWBrackets(), ColorLookFor, ColorShowAlways,
       cmb_zqColor
1012|  cmb_zqColor.ListIndex = 0
1013|
1014|  cmb_zqFontCode.Clear
1015|  FillCMBFromSGSTLST FieldListWBrackets(), FontCodeLookFor,
       FontCodeShowAlways, cmb_zqFontCode
1016|  cmb_zqFontCode.ListIndex = 0
1017|
1018|  Exit Sub
1019|  ERRcmbTblQryList_Click:
1020|    MsgBox "Error #" & Err & " occurred in cmbTblQryList_Click." & CRLF & Error
1021|    Err = 0
1022|    Exit Sub
1023|    Resume Next
1024|  End Sub
1025|
1026|  Sub cmdCancel_Click ()
1027|   Me.Hide
1028|  End Sub
1029|
1030|  Sub cmdOK_Click ()
1031|  If cmbTblQryList = "" Then
1032|      MsgBox "A query or table must be selected before trying to display its results." & CRLF
       & "Or you can hit CANCEL to not add a point theme right now and get out of here."
1033|      cmbTblQryList.SetFocus
1034|      SendKeys "%{DOWN}"
1035|      Exit Sub
1036|  End If
1037|  If cmbFontList = "" Then
1038|      MsgBox "A font must be selected before using it as point symbols." & CRLF & "Or you
       can hit CANCEL to not add a point theme right now and get out of here."
1039|      cmbFontList.SetFocus
1040|      SendKeys "%{DOWN}"
1041|      Exit Sub
1042|  End If
1043|       'Call DBFormSQLAddField("")
1044|    cmdOK.SetFocus
1045|  Me.Hide
1046|
1047|    cmdOK.FontItalic = True
1048|
1049|  End Sub
1050|
1051|  Sub DBFormSQLAddField (NewField As String)
1052|  If flgIgnoreMapSQLChange Then Exit Sub
1053|  If NewField <> "" Then
1054|   If ListOfFieldsForDBForm = "*" Then
1055|      ListOfFieldsForDBForm = "[" & NewField & "]"
```

```
1056|    Else
1057|     ListOfFieldsForDBForm = ListOfFieldsForDBForm & ", [" & NewField & "]"
1058|   End If
1059|  End If
1060|  txtSpatialQuerySQL = "SELECT " & ListOfFieldsForDBForm & ", " &
      cmb_zqFeatureID.Text & " AS zqFeatureID FROM " & "[" & cmbTblQryList & "]"
1061|  End Sub
1062|
1063|  Sub DecodeMapSQL (strSQLIn As String)
1064|   Dim i As Long, h As Integer, strTemp As String
1065|   strSQLIn = Trim$(strSQLIn)
1066|   If strSQLIn = "" Then Exit Sub
1067|
1068|
1069|  'If the MapSQL was changed by another form or by the user,
1070|  '   set the comboboxes to reflect the changes as best possible.
1071|   Dim strParameters  As String, strFields As String, strFrom As String, strWhere As
      String, strGroupBy As String, strHaving As String, strOrderBy As String
1072|
1073|
1074|
1075|   If SplitSQLStatement(strSQLIn, strParameters, strFields, strFrom, strWhere,
      strGroupBy, strHaving, strOrderBy) = False Then Exit Sub
1076|
1077|   If strParameters <> "" Then
1078|     MsgBox "WHEAT programs do not run parameter queries.  Please choose another
      map SQL string."
1079|     Exit Sub
1080|   End If
1081|   flgIgnoreMapSQLChange = True
1082|   Dim strProblems As String
1083|
1084|   strFrom = Strip2Brackets(strFrom)
1085|  'Set the source table or query
1086|   i = IndexOfStringInCOMBOBOX(strFrom, Me!cmbTblQryList)
1087|   Me!cmbTblQryList.ListIndex = i
1088|    DoEvents 'Let the click event occur for the QT List
1089|
1090|  'Get field assignments
1091|   Dim strRawName() As String, strAsName() As String, nFields As Long, nF As Long
1092|   Call PullFieldNamesFromString(strFields, strRawName(), strAsName(), nFields)
1093|   'For i = 1 To nFields
1094|   '   dDebug.Print strRawName(i), , strAsName(i)
1095|   'Next i
1096|
1097|    'Set X Coordinate field
1098|   For nF = 1 To nFields
1099|    If strAsName(nF) = "zqX" Then
1100|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqX)
1101|      If i <> -1 Then
1102|        Me.cmb_zqX.ListIndex = i
1103|       Else
1104|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
      for X-coordinates, but this routine cannot find it.  You will have to select this one
```

```
      yourself." & CRLF
1105|     End If
1106|     Exit For
1107|   End If
1108|  Next nF
1109|
1110|   'Set Y Coordinate field
1111|  For nF = 1 To nFields
1112|   If strAsName(nF) = "zqY" Then
1113|     i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqY)
1114|     If i <> -1 Then
1115|       Me.cmb_zqY.ListIndex = i
1116|       Else
1117|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
      for Y-coordinates, but this routine cannot find it.  You will have to select this one
      yourself." & CRLF
1118|     End If
1119|     Exit For
1120|   End If
1121|  Next nF
1122|
1123|   'Set FeatureID field
1124|  For nF = 1 To nFields
1125|   If strAsName(nF) = "zqFeatureID" Then
1126|     i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFeatureID)
1127|     If i <> -1 Then
1128|       Me.cmb_zqFeatureID.ListIndex = i
1129|       Else
1130|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
      for feature ID, but this routine cannot find it.  You will have to select this one yourself." &
      CRLF
1131|     End If
1132|     Exit For
1133|   End If
1134|  Next nF
1135|
1136|   'Font size
1137|  For nF = 1 To nFields
1138|   If strAsName(nF) = "zqFontSize" Then
1139|     i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFontSize)
1140|     If i <> -1 Then
1141|       Me.cmb_zqFontSize.ListIndex = i
1142|       Else
1143|       i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFontSize)
1144|       If i <> -1 Then
1145|         Me.cmb_zqFontSize.ListIndex = i
1146|         Else
1147|          strProblems = strProblems & "This theme originally used " & strRawName(nF) &
      " for the font size, but this routine cannot find it.  You will have to select this one yourself."
      & CRLF
1148|       End If
1149|     End If
1150|     Exit For
1151|   End If
```

```
1152|   Next nF
1153|
1154|      'The color
1155|   For nF = 1 To nFields
1156|    If strAsName(nF) = "zqColor" Then
1157|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqColor)
1158|      If i <> -1 Then
1159|        Me.cmb_zqColor.ListIndex = i
1160|       Else
1161|        i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqColor)
1162|        If i <> -1 Then
1163|          Me.cmb_zqColor.ListIndex = i
1164|         Else
1165|
1166|           strProblems = strProblems & "This theme originally used " & strRawName(nF) &
      " for the color, but this routine cannot find it.  You will have to select this one yourself." &
      CRLF
1167|        End If
1168|      End If
1169|      Exit For
1170|    End If
1171|   Next nF
1172|
1173|     'The font code
1174|   For nF = 1 To nFields
1175|    If strAsName(nF) = "zqFontCode" Then
1176|      'If the fontcode is stored in a field, this will find it.
1177|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFontCode)
1178|      If i <> -1 Then
1179|        Me.cmb_zqFontCode.ListIndex = i
1180|       Else
1181|        'if the font code was not in a field found, it should probably be a
1182|        ' a constant.
1183|        h = Val(strRawName(nF))
1184|        i = IndexOfStringInCOMBOBOX(FontNameFromID(h), Me!cmbFontList)
1185|        If i <> -1 Then
1186|          Me!cmbFontList.ListIndex = i
1187|         Else
1188|          Me!cmbFontList.ListIndex = 0
1189|        End If
1190|        If And2(h, FONT_BOLD) Then
1191|          chkFontBold = 1
1192|         Else
1193|          chkFontBold = 0
1194|        End If
1195|        If And2(h, FONT_ITALIC) Then
1196|          chkFontItalic = 1
1197|         Else
1198|          chkFontItalic = 0
1199|        End If
1200|
1201|        i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFontCode)
1202|      End If
1203|      Exit For
```

```
1204|    End If
1205|   Next nF
1206|
1207|
1208|     'The text itself
1209|   For nF = 1 To nFields
1210|    If strAsName(nF) = "zqLabelText" Then
1211|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqLabelText)
1212|      If i <> -1 Then
1213|        Me.cmb_zqLabelText.ListIndex = i
1214|       Else
1215|        'Hope that this works
1216|        Me!cmb_zqLabelText.AddItem strRawName(nF)
1217|        Me!cmb_zqLabelText.ListIndex = Me!cmb_zqLabelText.NewIndex
1218|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
     for the text, but this routine cannot find it. Added that to the list.  Hope it works.  You will
     have to check this one yourself." & CRLF
1219|      End If
1220|      Exit For
1221|    End If
1222|   Next nF
1223|
1224|
1225|    'Set cmb_zqAlignment
1226|   For nF = 1 To nFields
1227|    If strAsName(nF) = "zqAlignment" Then
1228|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqAlignment)
1229|      If i <> -1 Then
1230|        Me.cmb_zqAlignment.ListIndex = i
1231|       Else
1232|        i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqAlignment)
1233|        If i <> -1 Then
1234|          Me.cmb_zqAlignment.ListIndex = i
1235|         Else
1236|          strProblems = strProblems & "This theme originally used " & strRawName(nF) &
     " for cmb_zqAlignment, but this routine cannot find it.  You will have to select this one
     yourself." & CRLF
1237|        End If
1238|      End If
1239|      Exit For
1240|    End If
1241|   Next nF
1242|    txtCriteria = strWhere
1243|
1244|
1245|   flgIgnoreMapSQLChange = False
1246|    If strProblems <> "" Then
1247|      Beep
1248|      MsgBox strProblems
1249|    End If
1250|
1251|  End Sub
1252|
1253|  Sub Form_Load ()
```

```
1254|  FlagActivateDragDrop = 0
1255|    Me!cmbFontList.Clear
1256|    Call LoadComboBoxWithFontNames(Me!cmbFontList)
1257|    Me!cmbTblQryList.Clear
1258|    For i = 1 To nTblQrys
1259|     Me!cmbTblQryList.AddItem strTblQryList(i)
1260|    Next i
1261|
1262|  End Sub
1263|
1264|  Sub Form_Paint ()
1265|    If cmbFontList.ListIndex = -1 Then cmbFontList.ListIndex = 0
1266|    cmdOK.FontItalic = False
1267|    If cmbTblQryList.ListIndex = -1 Then cmbTblQryList.ListIndex = 0
1268|  End Sub
1269|
1270|  Sub imgFieldName_Click ()
1271|   cmbFieldList.Drag
1272|  End Sub
1273|
1274|  Sub optBaseline_Click ()
1275|   UpdateTextAligmentFromButtons
1276|  End Sub
1277|
1278|  Sub optBottom_Click ()
1279|   UpdateTextAligmentFromButtons
1280|  End Sub
1281|
1282|  Sub optCenter_Click ()
1283|   UpdateTextAligmentFromButtons
1284|  End Sub
1285|
1286|  Sub optLeft_Click ()
1287|   UpdateTextAligmentFromButtons
1288|  End Sub
1289|
1290|  Sub optRight_Click ()
1291|   UpdateTextAligmentFromButtons
1292|  End Sub
1293|
1294|  Sub optTop_Click ()
1295|   UpdateTextAligmentFromButtons
1296|  End Sub
1297|
1298|  Sub picSample_Paint ()
1299|  ShowSample
1300|  End Sub
1301|
1302|  Sub SetMapSQL ()
1303|  If flgIgnoreMapSQLChange Then Exit Sub
1304|  ShowSample
1305|
1306|
1307|  Dim Temp As String
```

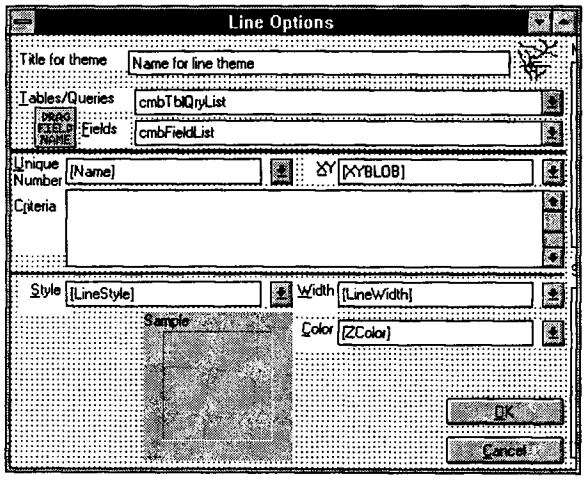
```
1308| Static TempSQL As String
1309|
1310| TempSQL = "SELECT DISTINCTROW "
1311| ' Assemble a new MapSQL statement.
1312| 'TempSQL = TempSQL & GetGoodValueField(cmb_zqX.Text) & " AS zqX, " 'Field 1
1313| 'TempSQL = TempSQL & GetGoodValueField(cmb_zqY.Text) & " AS zqY, " 'Field 2
1314| 'TempSQL = TempSQL & GetGoodValueField(cmb_zqFontSize.Text) & " AS
      zqFontSize, " 'Field 3
1315| 'TempSQL = TempSQL & GetGoodValueField(cmb_zqColor.Text) & " AS zqColor, "
      'Field 4
1316| 'TempSQL = TempSQL & GetGoodValueField(cmb_zqFontCode.Text) & " AS
      zqFontCode, " 'New 940701  'Field 5
1317| 'TempSQL = TempSQL & cmb_zqLabelText.Text & " AS zqLabelText, " ' Field 6
      'New, 940708 'Changed 940831 to not do the GetGoodValue....
1318| 'TempSQL = TempSQL & GetGoodValueField(cmb_zqAlignment.Text) & " AS
      zqAlignment, "'Field 7 'New, 940708
1319| TempSQL = TempSQL & AddVarNameAsNeeded(GetGoodValueField(cmb_zqX.Text),
      "zqX") & " , "
1320| TempSQL = TempSQL & AddVarNameAsNeeded(GetGoodValueField(cmb_zqY.Text),
      "zqY") & ", "
1321| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqFontSize.Text), "zqFontSize") & " ,
      "
1322| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqColor.Text), "zqColor") & " , "
1323| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqFontCode.Text), "zqFontCode") &
      " , "
1324| TempSQL = TempSQL & cmb_zqLabelText.Text & " AS zqLabelText, "
1325| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqAlignment.Text), "zqAlignment") &
      " , "
1326| If chkOpaque Then
1327|     TempSQL = TempSQL & "-1 AS zqOpaque, " 'New, 940708
1328|   Else
1329|     TempSQL = TempSQL & "0 AS zqOpaque, " 'New, 940708    'Field 8
1330| End If
1331|
1332| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqFeatureID.Text), "zqFeatureID") &
      " "
1333|
1334| TempSQL = TempSQL & " FROM [" & cmbTblQryList & "] "
1335| 'Might want to tack on some ORDER BY clauses, to minimize the number
1336| ' of font changes needed.
1337|
1338|     'Do not add any criteria of your own.
1339| If (txtCriteria = "") Then
1340|   TempSQL = TempSQL
1341|  Else
1342|   TempSQL = TempSQL & "  WHERE (" & txtCriteria & ") "
1343| End If
1344|
1345|
```

```
1346|
1347|
1348| 'Build the Order By clause
1349| Dim strOrderBy As String    'changed 951201
1350|   strOrderBy = ""
1351|
1352|   Temp = GetFieldForOrderBy(cmb_zqAlignment.Text)
1353|   If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
1354|
1355|   Temp = GetFieldForOrderBy(cmb_zqFontCode.Text)
1356|   If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
1357|
1358|   Temp = GetFieldForOrderBy(cmb_zqColor.Text)
1359|   If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
1360|
1361|
1362|   Temp = GetFieldForOrderBy(cmb_zqFontSize.Text)
1363|   If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
1364|
1365|   If strOrderBy <> "" Then
1366|     strOrderBy = " ORDER BY " & Left$(strOrderBy, Len(strOrderBy) - 2)
1367|   End If
1368|
1369|   TempSQL = TempSQL & strOrderBy & " ; "
1370|
1371|
1372| ' Now put the SQL statement into the appropriate text box.
1373| flgIgnoreMapSQLChange = True
1374|   txtMapSQL = TempSQL
1375| flgIgnoreMapSQLChange = False
1376| End Sub
1377|
1378| Sub ShowSample ()
1379| Dim OldAlignment As Integer
1380| OldAlignment = SetTextAlign(Me.picSample.hDC, intTextAlign)
1381| 'Display for user
1382|   Dim XPoint As Single, YPoint As Single
1383|   XPoint = .5 * picSample.ScaleWidth
1384|   YPoint = .5 * picSample.ScaleHeight
1385|   picSample.Cls
1386|   picSample.Line (0, YPoint)-(2 * XPoint, YPoint), RGB(255, 0, 0)
1387|   picSample.Line (XPoint, 0)-(XPoint, 2 * YPoint), RGB(255, 0, 0)
1388|   picSample.CurrentX = XPoint
1389|   picSample.CurrentY = YPoint
1390|   SetFontInPIC intFontCode, Me.picSample
1391|   picSample.ForeColor = Val(cmb_zqColor)
1392|   'SetFontCodeInPICxxx intFontCode, Me.picSample
1393|    Dim Holder As Single
1394|    Holder = Val(cmb_zqFontSize)
1395|    If Holder < 4 Then
1396|       Me.picSample.FontSize = 4
1397|     Else
1398|       Me.picSample.FontSize = Holder
1399|    End If
```

```
1400|
1401|    picSample.Print cmb_zqLabelText.Text
1402|
1403|  End Sub
1404|
1405|  Sub txtCriteria_Change ()
1406|   SetMapSQL
1407|  End Sub
1408|
1409|  Sub txtCriteria_DragDrop (Source As Control, X As Single, Y As Single)
1410|  'txtCriteria = Mid$(txtCriteria, 1, txtCriteria.SelStart) & " " &
       AddBracketsAsNeeded(CStr(Source)) & " " & Mid$(txtCriteria, txtCriteria.SelStart +
       txtCriteria.SelLength + 1, 32768)
1411|  txtCriteria = Mid$(txtCriteria, 1, txtCriteria.SelStart) & " " & "[" & Source & "]" & " " &
       Mid$(txtCriteria, txtCriteria.SelStart + txtCriteria.SelLength + 1, 32768)
1412|  'Drops the selected text at the current insertion point
1413|  End Sub
1414|
1415|  Sub txtMapSQL_KeyPress (KeyAscii As Integer)
1416|
1417|  If flgIgnoreMapSQLChange Then Exit Sub
1418|  If KeyAscii <> 13 Then Exit Sub
1419|  KeyAscii = 0'Swallow the return key
1420|   Dim strSQL As String
1421|   strSQL = txtMapSQL
1422|   Call DecodeMapSQL(strSQL)
1423|
1424|
1425|
1426|  End Sub
1427|
1428|  Sub txtSpatialQuerySQL_DragDrop (Source As Control, X As Single, Y As Single)
1429|  Call DBFormSQLAddField(CStr(Source))
1430|  End Sub
1431|
1432|  Sub txtSpatialQuerySQL_KeyPress (KeyAscii As Integer)
1433|  Select Case KeyAscii
1434|     Case 13 'Return/ENTER
1435|     KeyAscii = 0
1436|     'Maybe decode the QS SQL
1437|  End Select
1438|
1439|  End Sub
1440|
1441|  Sub UpdateFontCode ()
1442|
1443|   On Error Resume Next
1444|     intFontCode = cmbFontList.ItemData(cmbFontList.ListIndex)
1445|     If chkFontBold Then intFontCode = intFontCode + FONT_BOLD
1446|     If chkFontItalic Then intFontCode = intFontCode + FONT_ITALIC
1447|      cmb_zqFontCode = Format$(intFontCode)
1448|      SetMapSQL
1449|      ShowSample
1450|  End Sub
```

```
1451|
1452| Sub UpdateTextAligmentFromButtons ()
1453| intTextAlign = 0
1454|
1455| 'Vertical cmb_zqAlignment
1456| If optTop Then intTextAlign = intTextAlign + TA_TOP
1457| If optBaseline Then intTextAlign = intTextAlign + TA_BASELINE
1458| If optBottom Then intTextAlign = intTextAlign + TA_BOTTOM
1459|
1460| 'Horizontal cmb_zqAlignment
1461| If optLeft Then intTextAlign = intTextAlign + TA_LEFT
1462| If optCenter Then intTextAlign = intTextAlign + TA_CENTER
1463| If optRight Then intTextAlign = intTextAlign + TA_RIGHT
1464| UpdateFontCode
1465| ShowSample
1466| cmb_zqAlignment = intTextAlign
1467| End Sub
1468|
1469| Sub zzlblColor_Click ()
1470|   Dim ColorWas As Long, NewColor As Long
1471|   ColorWas = Val(cmb_zqColor)
1472|   NewColor = GetColorWithCmnDialog(ColorWas)
1473|   If NewColor <> ColorWas Then AddAndHighlightItem Format$(NewColor),
      cmb_zqColor
1474| End Sub
1475|
1476| Sub zzlblColor_DragDrop (Source As Control, X As Single, Y As Single)
1477|   cmb_zqColor.AddItem "[" & Source & "]"
1478|   cmb_zqColor.ListIndex = cmb_zqColor.ListCount - 1
1479| End Sub
1480|
1481| Sub zzlblFontCode_DragDrop (Source As Control, X As Single, Y As Single)
1482|   cmb_zqFontCode.AddItem "[" & Source & "]"
1483|   cmb_zqFontCode.ListIndex = cmb_zqFontCode.ListCount - 1
1484|
1485| End Sub
1486|
1487| Sub zzlblSize_DragDrop (Source As Control, X As Single, Y As Single)
1488|   cmb_zqFontSize.AddItem "[" & Source & "]"
1489|   cmb_zqFontSize.ListIndex = cmb_zqFontSize.ListCount - 1
1490| End Sub
1491|
1492| Sub zzlblText_DragDrop (Source As Control, X As Single, Y As Single)
1493|  cmb_zqLabelText.AddItem "[" & Source & "]"
1494|  cmb_zqLabelText.ListIndex = cmb_zqLabelText.ListCount - 1
1495| End Sub
1496|
1497| Sub zzzlblFields_Click ()
1498|  cmbFieldList.Drag
1499| End Sub
1500|
1501| Sub zzzlblName_DragDrop (Source As Control, X As Single, Y As Single)
1502| 'Sets all of the following Z-field to whatever was dropped on it
1503|    cmb_zqFeatureID.AddItem "[" & Source & "]"
```

```
1504|    cmb_zqFeatureID.ListIndex = cmb_zqFeatureID.ListCount - 1
1505|
1506|  End Sub
1507|
1508|  Sub zzzlblX_DragDrop (Source As Control, X As Single, Y As Single)
1509|  Me!cmb_zqX.AddItem Mid$(Me!cmb_zqX, 1, Me!cmb_zqX.SelStart) & " [" &
       CStr(Source) & "] " & Mid$(Me!cmb_zqX, Me!cmb_zqX.SelStart +
       Me!cmb_zqX.SelLength + 1, 32768)
1510|  Me!cmb_zqX.ListIndex = Me!cmb_zqX.ListCount
1511|
1512|  End Sub
1513|
1514|  Sub zzzlblY_DragDrop (Source As Control, X As Single, Y As Single)
1515|     Me!cmb_zqY.AddItem Mid$(Me!cmb_zqY, 1, Me!cmb_zqY.SelStart) & " [" &
       CStr(Source) & "] " & Mid$(Me!cmb_zqY, Me!cmb_zqY.SelStart +
       Me!cmb_zqY.SelLength + 1, 32768)
1516|     Me!cmb_zqY.ListIndex = Me!cmb_zqY.ListCount - 1
1517|  End Sub
1518|
1519|  Sub zzzzlblCriteia_DragDrop (Source As Control, X As Single, Y As Single)
1520|  'txtCriteria = AddBracketsAsNeeded(CStr(Source))
1521|  txtCriteria = "[" & Source & "]"
1522|  End Sub
1523|
1524|  Sub zzzzlblDBFormSQL_DragDrop (Source As Control, X As Single, Y As Single)
1525|  Call DBFormSQLAddField(CStr(Source))
1526|  End Sub
1527|
```


Line Options
Title for theme
Name for line theme
Tables/Queries
cmbTblQryList
DRAG FIELD NAME
Fields
cmbFieldList
Unique Number
[Name]
XY
[XYBLOB]
Criteria
Style
[LineStyle]
Width
[LineWidth]
Sample
Color
[ZColor]
OK
Cancel

```
VERSION 2.00
Begin Form frmLineOptions
  Caption        = "Line Options"
  ClientHeight   = 5055
  ClientLeft     = 180
  ClientTop      = 450
  ClientWidth    = 6540
  ControlBox     = 0  'False
  Height         = 5460
  Icon           = LINOPT02.FRX:0000
  Left           = 120
  LinkTopic      = "Form1"
  MinButton      = 0  'False
  ScaleHeight    = 5055
  ScaleWidth     = 6540
  Top            = 105
  Width          = 6660
  Begin SSPanel zpnlOuter
    Alignment      = 0  'Left Justify - TOP
    AutoSize       = 3  'AutoSize Child To Panel
    BackColor      = &H00C0C0C0&
    BevelInner     = 1  'Inset
    BevelOuter     = 0  'None
    BorderWidth    = 15
    Font3D         = 0  'None
    ForeColor      = &H00FFFFFF&
    Height         = 1725
    Left           = 1530
    TabIndex       = 24
    Top            = 3240
    Width          = 1725
    Begin SSPanel zpnlSampleWord
      Alignment      = 0  'Left Justify - TOP
      AutoSize       = 3  'AutoSize Child To Panel
      BackColor      = &H00C0C0C0&
      BevelOuter     = 0  'None
      BorderWidth    = 1
      Caption        = "Sample"
      Font3D         = 0  'None
      FontBold       = 0  'False
      FontItalic     = 0  'False
      FontName       = "MS Sans Serif"
      FontSize       = 8.25
      FontStrikethru = 0  'False
      FontUnderline  = 0  'False
      ForeColor      = &H00000000&
      Height         = 220
      Left           = 0
      TabIndex       = 26
      Top            = 0
      Width          = 735
    End
    Begin PictureBox picSample
      BackColor      = &H00C0C0C0&
```

```
    BorderStyle    = 0 'None
    Height         = 1250
    Left           = 240
    ScaleHeight    = 21.96
    ScaleMode      = 6 'Millimeter
    ScaleWidth     = 21.96
    TabIndex       = 25
    Top            = 240
    Width          = 1250
  End
End
Begin ComboBox cmb_zqColor
  FontBold       = 0 'False
  FontItalic     = 0 'False
  FontName       = "MS Sans Serif"
  FontSize       = 8.25
  FontStrikethru = 0 'False
  FontUnderline  = 0 'False
  Height         = 300
  HelpContextID  = 2000
  Left           = 3780
  TabIndex       = 17
  Text           = "[ZColor]"
  Top            = 3330
  Width          = 2625
End
Begin ComboBox cmb_zqWidth
  FontBold       = 0 'False
  FontItalic     = 0 'False
  FontName       = "MS Sans Serif"
  FontSize       = 8.25
  FontStrikethru = 0 'False
  FontUnderline  = 0 'False
  Height         = 300
  HelpContextID  = 2100
  Left           = 3780
  TabIndex       = 15
  Text           = "[LineWidth]"
  Top            = 2880
  Width          = 2625
End
Begin ComboBox cmb_zqStyle
  FontBold       = 0 'False
  FontItalic     = 0 'False
  FontName       = "MS Sans Serif"
  FontSize       = 8.25
  FontStrikethru = 0 'False
  FontUnderline  = 0 'False
  Height         = 300
  HelpContextID  = 2200
  Left           = 630
  TabIndex       = 13
  Text           = "[LineStyle]"
  Top            = 2880
```

```
109|     Width         = 2625
110|   End
111|   Begin ComboBox cmb_zqFeatureID
112|     FontBold      = 0  'False
113|     FontItalic    = 0  'False
114|     FontName      = "MS Sans Serif"
115|     FontSize      = 8.25
116|     FontStrikethru = 0  'False
117|     FontUnderline = 0  'False
118|     Height        = 300
119|     HelpContextID = 1500
120|     Left          = 630
121|     TabIndex      = 7
122|     Text          = "[Name]"
123|     Top           = 1440
124|     Width         = 2625
125|   End
126|   Begin ComboBox cmb_zqXYBlobData
127|     FontBold      = 0  'False
128|     FontItalic    = 0  'False
129|     FontName      = "MS Sans Serif"
130|     FontSize      = 8.25
131|     FontStrikethru = 0  'False
132|     FontUnderline = 0  'False
133|     Height        = 300
134|     HelpContextID = 1200
135|     Left          = 3780
136|     TabIndex      = 9
137|     Text          = "[XYBLOB]"
138|     Top           = 1440
139|     Width         = 2625
140|   End
141|   Begin TextBox txtThemeName
142|     FontBold      = 0  'False
143|     FontItalic    = 0  'False
144|     FontName      = "MS Sans Serif"
145|     FontSize      = 8.25
146|     FontStrikethru = 0  'False
147|     FontUnderline = 0  'False
148|     Height        = 285
149|     Left          = 1350
150|     TabIndex      = 1
151|     Text          = "Name for line theme"
152|     Top           = 180
153|     Width         = 4425
154|   End
155|   Begin TextBox txtCriteria
156|     FontBold      = 0  'False
157|     FontItalic    = 0  'False
158|     FontName      = "MS Sans Serif"
159|     FontSize      = 8.25
160|     FontStrikethru = 0  'False
161|     FontUnderline = 0  'False
162|     Height        = 915
```

```
163|     HelpContextID  =  1600
164|     Left         =  630
165|     MultiLine     =  -1  'True
166|     ScrollBars    =  2  'Vertical
167|     TabIndex      =  11
168|     Top          =  1800
169|     Width        =  5775
170|   End
171|   Begin TextBox txtSpatialQuerySQL
172|     FontBold      =  0  'False
173|     FontItalic    =  0  'False
174|     FontName      =  "MS Sans Serif"
175|     FontSize      =  8.25
176|     FontStrikethru =  0  'False
177|     FontUnderline  =  0  'False
178|     Height       =  1995
179|     Left         =  6480
180|     MultiLine     =  -1  'True
181|     ScrollBars    =  2  'Vertical
182|     TabIndex      =  23
183|     Text         =  "SQL Statement to produce data displayed on DataBase Form"
184|     Top          =  2940
185|     Width        =  2805
186|   End
187|   Begin TextBox txtMapSQL
188|     FontBold      =  0  'False
189|     FontItalic    =  0  'False
190|     FontName      =  "MS Sans Serif"
191|     FontSize      =  8.25
192|     FontStrikethru =  0  'False
193|     FontUnderline  =  0  'False
194|     Height       =  2175
195|     Left         =  6480
196|     MultiLine     =  -1  'True
197|     ScrollBars    =  2  'Vertical
198|     TabIndex      =  21
199|     Text         =  "SQL statement to produce map data"
200|     Top          =  300
201|     Width        =  2805
202|   End
203|   Begin ComboBox cmbTblQryList
204|     FontBold      =  0  'False
205|     FontItalic    =  0  'False
206|     FontName      =  "MS Sans Serif"
207|     FontSize      =  8.25
208|     FontStrikethru =  0  'False
209|     FontUnderline  =  0  'False
210|     Height       =  300
211|     Left         =  1425
212|     Style        =  2  'Dropdown List
213|     TabIndex      =  3
214|     Top          =  630
215|     Width        =  4965
216|   End
```

```
Begin CommandButton cmdCancel
  Caption       = "&Cancel"
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 315
  Left          = 5040
  TabIndex      = 19
  Top           = 4680
  Width         = 1365
End
Begin CommandButton cmdOK
  Caption       = "&OK"
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 315
  Left          = 5040
  TabIndex      = 18
  Top           = 4230
  Width         = 1365
End
Begin ComboBox cmbFieldList
  DragIcon      = LINOPT02.FRX:0302
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 300
  Left          = 1425
  Sorted        = -1  'True
  Style         = 2  'Dropdown List
  TabIndex      = 5
  Top           = 990
  Width         = 4965
End
Begin Label ACTION
  BackColor     = &H0000FFFF&
  ForeColor     = &H000000FF&
  Height        = 195
  Left          = 1620
  TabIndex      = 27
  Top           = 2160
  Visible       = 0  'False
  Width         = 1995
End
```

```
271|  Begin Image imgFieldName
272|    Height        = 480
273|    Left          = 270
274|    Picture       = LINOPT02.FRX:0604
275|    Top           = 840
276|    Width         = 480
277|  End
278|  Begin Line Line2
279|    BorderWidth   = 2
280|    X1            = 6390
281|    X2            = 0
282|    Y1            = 2790
283|    Y2            = 2790
284|  End
285|  Begin Line Line1
286|    BorderWidth   = 3
287|    X1            = 6390
288|    X2            = 0
289|    Y1            = 1350
290|    Y2            = 1350
291|  End
292|  Begin Image zImageExample
293|    Height        = 480
294|    Left          = 5850
295|    Picture       = LINOPT02.FRX:0906
296|    Top           = 45
297|    Width         = 480
298|  End
299|  Begin Label lblThemeName
300|    Caption       = "Title for theme"
301|    FontBold      = 0  'False
302|    FontItalic    = 0  'False
303|    FontName      = "MS Sans Serif"
304|    FontSize      = 8.25
305|    FontStrikethru = 0  'False
306|    FontUnderline = 0  'False
307|    Height        = 225
308|    Left          = 90
309|    TabIndex      = 0
310|    Top           = 180
311|    Width         = 1275
312|  End
313|  Begin Label zzzzzlblZColor
314|    Alignment     = 2  'Center
315|    Caption       = "&Color"
316|    FontBold      = 0  'False
317|    FontItalic    = 0  'False
318|    FontName      = "MS Sans Serif"
319|    FontSize      = 8.25
320|    FontStrikethru = 0  'False
321|    FontUnderline = 0  'False
322|    Height        = 375
323|    Left          = 3240
324|    TabIndex      = 16
```

```
325|     Top          = 3330
326|     Width        = 615
327|   End
328|   Begin Label zzzlblZSymbol
329|     Alignment    = 2 'Center
330|     Caption      = "&Style"
331|     FontBold     = 0 'False
332|     FontItalic   = 0 'False
333|     FontName     = "MS Sans Serif"
334|     FontSize     = 8.25
335|     FontStrikethru = 0 'False
336|     FontUnderline = 0 'False
337|     Height       = 375
338|     Left         = 90
339|     TabIndex     = 12
340|     Top          = 2880
341|     Width        = 615
342|   End
343|   Begin Label zzzlblCriteia
344|     Caption      = "C&riteria"
345|     FontBold     = 0 'False
346|     FontItalic   = 0 'False
347|     FontName     = "MS Sans Serif"
348|     FontSize     = 8.25
349|     FontStrikethru = 0 'False
350|     FontUnderline = 0 'False
351|     Height       = 645
352|     Left         = 30
353|     TabIndex     = 10
354|     Top          = 1890
355|     Width        = 615
356|   End
357|   Begin Label zzzlblDBFormSQL
358|     Caption      = "Spatial Query SQL"
359|     FontBold     = 0 'False
360|     FontItalic   = 0 'False
361|     FontName     = "MS Sans Serif"
362|     FontSize     = 8.25
363|     FontStrikethru = 0 'False
364|     FontUnderline = 0 'False
365|     Height       = 465
366|     Left         = 6480
367|     TabIndex     = 22
368|     Top          = 2640
369|     Width        = 1500
370|   End
371|   Begin Label zzzlblMapTileSQL
372|     Caption      = "Map Lines SQL"
373|     FontBold     = 0 'False
374|     FontItalic   = 0 'False
375|     FontName     = "MS Sans Serif"
376|     FontSize     = 8.25
377|     FontStrikethru = 0 'False
378|     FontUnderline = 0 'False
```

```
379|     Height      = 465
380|     Left        = 6480
381|     TabIndex    = 20
382|     Top         = 60
383|     Width       = 1500
384|   End
385|   Begin Label zzzlblFields
386|     Caption     = "&Fields"
387|     FontBold    = 0  'False
388|     FontItalic  = 0  'False
389|     FontName    = "MS Sans Serif"
390|     FontSize    = 8.25
391|     FontStrikethru = 0  'False
392|     FontUnderline  = 0  'False
393|     Height      = 285
394|     Left        = 810
395|     TabIndex    = 4
396|     Top         = 990
397|     Width       = 555
398|   End
399|   Begin Label zzzlblTables
400|     Caption     = "&Tables/Queries"
401|     FontBold    = 0  'False
402|     FontItalic  = 0  'False
403|     FontName    = "MS Sans Serif"
404|     FontSize    = 8.25
405|     FontStrikethru = 0  'False
406|     FontUnderline  = 0  'False
407|     Height      = 270
408|     Left        = 90
409|     TabIndex    = 2
410|     Top         = 630
411|     Width       = 1635
412|   End
413|   Begin Label zzzlblName
414|     Caption     = "&Unique Number"
415|     FontBold    = 0  'False
416|     FontItalic  = 0  'False
417|     FontName    = "MS Sans Serif"
418|     FontSize    = 8.25
419|     FontStrikethru = 0  'False
420|     FontUnderline  = 0  'False
421|     Height      = 495
422|     Left        = 30
423|     TabIndex    = 6
424|     Top         = 1395
425|     Width       = 615
426|   End
427|   Begin Label zzzlblZSize
428|     Alignment   = 2  'Center
429|     Caption     = "&Width"
430|     FontBold    = 0  'False
431|     FontItalic  = 0  'False
432|     FontName    = "MS Sans Serif"
```

```
433|      FontSize      = 8.25
434|      FontStrikethru = 0  'False
435|      FontUnderline  = 0  'False
436|      Height        = 405
437|      Left          = 3285
438|      TabIndex      = 14
439|      Top           = 2880
440|      Width         = 495
441|    End
442|    Begin Label zzzlblX
443|      Alignment     = 1  'Right Justify
444|      Caption       = "&XY"
445|      FontBold      = 0  'False
446|      FontItalic    = 0  'False
447|      FontName      = "MS Sans Serif"
448|      FontSize      = 8.25
449|      FontStrikethru = 0  'False
450|      FontUnderline  = 0  'False
451|      Height        = 255
452|      Left          = 3420
453|      TabIndex      = 8
454|      Top           = 1440
455|      Width         = 330
456|    End
457| End
458| Option Explicit
459| Option Compare Text
460| Dim flgValidSyntax As Integer
461| Dim ListOfFieldsForDBForm As String
462| Dim FlagActivateDragDrop As Integer
463| Dim tblPointMap As Table
464|
465| Dim FieldsInCurrentQT() As String, FieldListWBrackets() As String
466| Dim nStoredMaps As Long
467|
468|
469| Dim flgIgnoreMapSQLChange As Long
470|
471| Sub ACTION_Change ()
472|   Dim tmp As String
473|   tmp = UCase$(Me!ACTION.Caption)
474|   Select Case tmp
475|    Case "Decode MapSQL"
476|       tmp = Me!txtMapSQL
477|       Call DecodeMapSQL(tmp)
478|    Case "wait" 'do nothing
479|    Case "refresh"
480|       Call Form_Load
481|    Case Else
482|   End Select
483| End Sub
484|
485| Sub cmb_zqColor_Change ()
486|   SetMapSQL
```

```
End Sub

Sub cmb_zqColor_Click ()
 SetMapSQL
End Sub

Sub cmb_zqColor_DragDrop (Source As Control, X As Single, Y As Single)
   cmb_zqColor = Mid$(cmb_zqColor, 1, cmb_zqColor.SelStart) & " [" & Source & "] " &
Mid$(cmb_zqColor, cmb_zqColor.SelStart + cmb_zqColor.SelLength + 1, 32768)
End Sub

Sub cmb_zqFeatureID_Change ()
 SetMapSQL
End Sub

Sub cmb_zqFeatureID_Click ()
 SetMapSQL
 Call DBFormSQLAddField("")
End Sub

Sub cmb_zqFeatureID_DragDrop (Source As Control, X As Single, Y As Single)
cmb_zqFeatureID.AddItem "[" & Source & "]"
cmb_zqFeatureID.ListIndex = cmb_zqFeatureID.ListCount

'Drops the selected text over the entire field, for use on
' NAME only, because we'll need this for matching and it
' NAME should never be a calculated field, at least not easily.

End Sub

Sub cmb_zqStyle_Change ()
 SetMapSQL
End Sub

Sub cmb_zqStyle_Click ()
   SetMapSQL
End Sub

Sub cmb_zqStyle_DragDrop (Source As Control, X As Single, Y As Single)
 'LineStyle = Mid$(LineStyle, 1, LineStyle.SelStart) & " " &
AddBracketsAsNeeded(CStr(Source)) & " " & Mid$(LineStyle, LineStyle.SelStart +
LineStyle.SelLength + 1, 32768)
  cmb_zqStyle = Mid$(cmb_zqStyle, 1, cmb_zqStyle.SelStart) & " " & "[" & Source & "]" &
" " & Mid$(cmb_zqStyle, cmb_zqStyle.SelStart + cmb_zqStyle.SelLength + 1, 32768)
'Drops the selected text at the current insertion point
End Sub

Sub cmb_zqWidth_Change ()
 SetMapSQL
End Sub

Sub cmb_zqWidth_Click ()
  SetMapSQL
End Sub
```

```
537|
538| Sub cmb_zqWidth_DragDrop (Source As Control, X As Single, Y As Single)
539|  'cmb_zqWidth = Mid$(cmb_zqWidth, 1, cmb_zqWidth.SelStart) & " " &
     AddBracketsAsNeeded(CStr(Source)) & " " & Mid$(cmb_zqWidth,
     cmb_zqWidth.SelStart + cmb_zqWidth.SelLength + 1, 32768)
540|  cmb_zqWidth = Mid$(cmb_zqWidth, 1, cmb_zqWidth.SelStart) & " [" & Source & "] " &
     Mid$(cmb_zqWidth, cmb_zqWidth.SelStart + cmb_zqWidth.SelLength + 1, 32768)
541| 'Drops the selected text at the current insertion point
542| End Sub
543|
544| Sub cmb_zqXYBlobData_Change ()
545|  SetMapSQL
546| End Sub
547|
548| Sub cmb_zqXYBlobData_Click ()
549|  SetMapSQL
550| End Sub
551|
552| Sub cmb_zqXYBlobData_DragDrop (Source As Control, X As Single, Y As Single)
553| cmb_zqXYBlobData = Mid$(cmb_zqXYBlobData, 1, cmb_zqXYBlobData.SelStart) & " ["
     & Source & "] " & Mid$(cmb_zqXYBlobData, cmb_zqXYBlobData.SelStart +
     cmb_zqXYBlobData.SelLength + 1, 32768)
554| 'Drops the selected text at the current insertion point
555| End Sub
556|
557| Sub cmbFieldList_Click ()
558| 'If FlagActivateDragDrop <> 0 Then
559| '   cmbFieldList.Drag
560| '  Else
561| '    FlagActivateDragDrop = 1
562| 'End If
563| End Sub
564|
565| Sub cmbFieldList_KeyDown (KeyCode As Integer, Shift As Integer)
566|   If KeyCode = KEY_INSERT And Shift = 2 Then 'CTRL-Insert?
567|     clipboard.Clear
568|     clipboard.SetText ("[" & cmbFieldList & "]")
569|   End If
570| End Sub
571|
572| Sub cmbTblQryList_Click ()
573| If Me!ACTION.Caption = "wait" Then Exit Sub
574|
575|  If Not flgIgnoreMapSQLChange Then
576|   'For usual case, set the SpatialQuerySQL and the theme name
577|   txtSpatialQuerySQL = "SELECT * FROM " &
     AddBracketsAsNeeded(CStr(cmbTblQryList))
578|   ListOfFieldsForDBForm = "*"
579|   txtThemeName = "LINES: " & cmbTblQryList & "  " & Now
580|  End If
581|
582| Dim TableName As String
583|
584| On Error GoTo ERRcmbTblQryList_Click
```

```
585  TableName = cmbTblQryList
586  Set snpFieldList = MapDB.CreateSnapshot(TableName)
587  nFields = snpFieldList.Fields.Count
588  cmbFieldList.Clear
589  ReDim FieldListWBrackets(1 To nFields)
590  ReDim FieldsInCurrentQT(1 To nFields), FieldsWBrackets(1 To nFields)
591
592  cmb_zqXYBlobData.Clear
593  For i = 1 To nFields
594    cmbFieldList.AddItem snpFieldList.Fields(i - 1).Name
595    FieldsInCurrentQT(i) = snpFieldList.Fields(i - 1).Name
596    FieldListWBrackets(i) = "[" & snpFieldList.Fields(i - 1).Name & "]"
597    If snpFieldList.Fields(i - 1).Type = DB_LONGBINARY Then 'BLOB
598       cmb_zqXYBlobData.AddItem "[" & snpFieldList.Fields(i - 1).Name & "]"
599    End If
600  Next i
601  snpFieldList.Close
602  cmbFieldList.ListIndex = 0
603  txtCriteria = ""
604
605   If cmb_zqXYBlobData.ListCount = 0 Then
606    Beep
607    MsgBox "This field contains no line data." & CRLF & "Please choose a new query or
     table that does."
608    cmbTblQryList.SetFocus
609    SendKeys "%{DOWN}"
610    Exit Sub
611   Else
612    cmb_zqXYBlobData.ListIndex = 0
613   End If
614
615  cmb_zqFeatureID.Clear
616   FillCMBFromSGSTLST FieldListWBrackets(), FeatureNameLookFor,
     FeatureNameShowAlways, cmb_zqFeatureID
617  cmb_zqFeatureID.ListIndex = 0
618
619  cmb_zqStyle.Clear
620  FillCMBFromSGSTLST FieldListWBrackets(), LineStyleLookFor, LineStyleShowAlways,
     cmb_zqStyle
621  cmb_zqStyle.ListIndex = 0
622
623  cmb_zqColor.Clear
624  FillCMBFromSGSTLST FieldListWBrackets(), ColorLookFor, ColorShowAlways,
     cmb_zqColor
625  cmb_zqColor.ListIndex = 0
626
627  cmb_zqWidth.Clear
628  FillCMBFromSGSTLST FieldListWBrackets(), LineWidthLookFor,
     LineWidthShowAlways, cmb_zqWidth
629  cmb_zqWidth.ListIndex = 0
630
631
632   FlagActivateDragDrop = 0
633
```

```
Exit Sub
ERRcmbTblQryList_Click:
  MsgBox "Error #" & Err & " occurred in cmbTblQryList_Click." & CRLF & Error
  Err = 0
  Exit Sub
  Resume Next
End Sub

Sub cmdCancel_Click ()
Me.Hide
End Sub

Sub cmdOK_Click ()
If cmbTblQryList = "" Then
    MsgBox "A query or table must be selected before trying to display its results." & CRLF
& "Or you can hit CANCEL to not add a point theme right now and get out of here."
    cmbTblQryList.SetFocus
    SendKeys "%{DOWN}"
    Exit Sub
End If

     'Call DBFormSQLAddField("")

  cmdOK.SetFocus
Me.Hide

  cmdOK.FontItalic = True

End Sub

Sub DBFormSQLAddField (NewField As String)
If flgIgnoreMapSQLChange Then Exit Sub
If NewField <> "" Then
 If ListOfFieldsForDBForm = "*" Then
    ListOfFieldsForDBForm = "[" & NewField & "]"
  Else
    ListOfFieldsForDBForm = ListOfFieldsForDBForm & ", [" & NewField & "]"
 End If
End If
txtSpatialQuerySQL = "SELECT " & ListOfFieldsForDBForm & ", " &
cmb_zqFeatureID.Text & " AS zqFeatureID FROM " & "[" & cmbTblQryList & "]"
End Sub

Sub DecodeMapSQL (strSQLIn As String)
  Dim i As Long, h As Integer, strTemp As String
  strSQLIn = Trim$(strSQLIn)
  If strSQLIn = "" Then Exit Sub



'If the MapSQL was changed by another form or by the user,
'  set the comboboxes to reflect the changes as best possible.
  Dim strParameters  As String, strFields As String, strFrom As String, strWhere As
String, strGroupBy As String, strHaving As String, strOrderBy As String
```

```
685|
686|
687|
688|  If SplitSQLStatement(strSQLIn, strParameters, strFields, strFrom, strWhere,
     strGroupBy, strHaving, strOrderBy) = False Then Exit Sub
689|
690|  If strParameters <> "" Then
691|    MsgBox "WHEAT programs do not run parameter queries.  Please choose another
     map SQL string."
692|    Exit Sub
693|  End If
694|  flgIgnoreMapSQLChange = True
695|  Dim strProblems As String
696|
697|  strFrom = Strip2Brackets(strFrom)
698| 'Set the source table or query
699|  i = IndexOfStringInCOMBOBOX(strFrom, Me!cmbTblQryList)
700|  Me!cmbTblQryList.ListIndex = i
701|   DoEvents 'Let the click event occur for the QT List
702|
703| 'Get field assignments
704|  Dim strRawName() As String, strAsName() As String, nFields As Long, nF As Long
705|  Call PullFieldNamesFromString(strFields, strRawName(), strAsName(), nFields)
706|  'For i = 1 To nFields
707|  '   dDebug.Print strRawName(i), , strAsName(i)
708|  'Next i
709|
710|   'Set XY Coordinate field
711|  For nF = 1 To nFields
712|   If strAsName(nF) = "zqXYBlobData" Then
713|     i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqXYBlobData)
714|     If i <> -1 Then
715|       Me.cmb_zqXYBlobData.ListIndex = i
716|      Else
717|       strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
     for the XY-coordinate chain, but this routine cannot find it.  You will have to select this
     one yourself." & CRLF
718|     End If
719|     Exit For
720|   End If
721|  Next nF
722|
723|   'Set FeatureID field
724|  For nF = 1 To nFields
725|   If strAsName(nF) = "zqFeatureID" Then
726|     i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFeatureID)
727|     If i <> -1 Then
728|       Me.cmb_zqFeatureID.ListIndex = i
729|      Else
730|       strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
     for feature ID, but this routine cannot find it.  You will have to select this one yourself." &
     CRLF
731|     End If
732|     Exit For
```

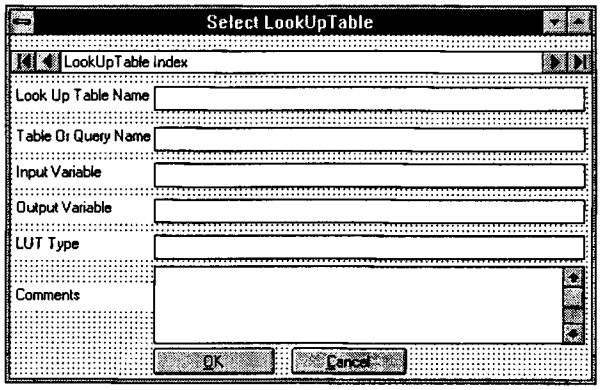
```
733|    End If
734|   Next nF
735|
736|    'Set line width field
737|   For nF = 1 To nFields
738|    If strAsName(nF) = "zqWidth" Then
739|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqWidth)
740|      If i <> -1 Then
741|        Me.cmb_zqWidth.ListIndex = i
742|       Else
743|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
      for line width, but this routine cannot find it.  You will have to select this one yourself." &
      CRLF
744|      End If
745|      Exit For
746|    End If
747|   Next nF
748|
749|
750|     'line color
751|   For nF = 1 To nFields
752|    If strAsName(nF) = "zqColor" Then
753|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqColor)
754|      If i <> -1 Then
755|        Me.cmb_zqColor.ListIndex = i
756|       Else
757|        i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqColor)
758|        If i <> -1 Then
759|          Me.cmb_zqColor.ListIndex = i
760|         Else
761|          strProblems = strProblems & "This theme originally used " & strRawName(nF) &
      " for the color, but this routine cannot find it.  You will have to select this one yourself." &
      CRLF
762|        End If
763|      End If
764|      Exit For
765|    End If
766|   Next nF
767|
768|      'The line style
769|   For nF = 1 To nFields
770|    If strAsName(nF) = "zqStyle" Then
771|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqStyle)
772|      If i <> -1 Then
773|        Me.cmb_zqStyle.ListIndex = i
774|       Else
775|        i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqStyle)
776|        If i <> -1 Then
777|          Me.cmb_zqStyle.ListIndex = i
778|         Else
779|
780|          strProblems = strProblems & "This theme originally used " & strRawName(nF) &
      " for the line style, but this routine cannot find it.  You will have to select this one
      yourself." & CRLF
```

```
781|         End If
782|       End If
783|       Exit For
784|     End If
785|   Next nF
786|
787|   txtCriteria = strWhere
788|
789|
790|   flgIgnoreMapSQLChange = False
791|   If strProblems <> "" Then
792|     Beep
793|     MsgBox strProblems
794|   End If
795|
796| End Sub
797|
798| Sub Form_Load ()
799|
800|   Me!cmbTblQryList.Clear
801|   For i = 1 To nTblQrys
802|    Me!cmbTblQryList.AddItem strTblQryList(i)
803|   Next i
804|    FlagActivateDragDrop = 0
805| End Sub
806|
807| Sub Form_Paint ()
808| ShowSample
809| If cmbTblQryList.ListIndex = -1 Then cmbTblQryList.ListIndex = 0
810| cmdOK.FontItalic = False
811| End Sub
812|
813| Sub imgFieldName_Click ()
814|  cmbFieldList.Drag
815| End Sub
816|
817| Sub SetMapSQL ()
818| If flgIgnoreMapSQLChange Then Exit Sub
819| ShowSample
820|
821| 'SELECT  DISTINCTROW [all section data].[Easting UTM14] AS X, [all section
     data].[Northing UTM14] AS Y, [all section data].[92 Groundwater used in acre-ft] AS Z,
     [all section data].SectionName AS Name
822| 'FROM [all section data]
823| 'WHERE (([all section data].[92 Groundwater used in acre-ft] Is Not Null));
824| Static TempSQL As String
825|
826| TempSQL = "SELECT DISTINCTROW "
827| ' Assemble a new MapSQL statement.
828| TempSQL = TempSQL & AddVarNameAsNeeded(cmb_zqXYBlobData.Text,
     "zqXYBlobData") & " , "
829|
830| TempSQL = TempSQL &
     AddVarNameAsNeeded(GetGoodValueField(cmb_zqWidth.Text), "zqWidth") & " , "
```

```
831| TempSQL = TempSQL &
     AddVarNameAsNeeded(GetGoodValueField(cmb_zqColor.Text), "zqColor") & " , "
832| TempSQL = TempSQL &
     AddVarNameAsNeeded(GetGoodValueField(cmb_zqStyle.Text), "zqStyle") & " , "
833| TempSQL = TempSQL &
     AddVarNameAsNeeded(GetGoodValueField(cmb_zqFeatureID.Text), "zqFeatureID") &
     " "
834|
835|
836| TempSQL = TempSQL & " FROM [" & cmbTblQryList & "] "
837|
838|       'Do not add any criteria of your own.
839| If (txtCriteria = "") Then
840|    TempSQL = TempSQL
841|   Else
842|    TempSQL = TempSQL & "  WHERE (" & txtCriteria & ") "
843| End If
844|
845| 'Build the Order By clause
846| Dim strOrderBy As String, Temp As String    'changed 950118
847|   strOrderBy = ""
848|
849|   Temp = GetFieldForOrderBy(cmb_zqStyle.Text)
850|   If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
851|
852|   Temp = GetFieldForOrderBy(cmb_zqStyle.Text)
853|   If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
854|
855|   Temp = GetFieldForOrderBy(cmb_zqColor.Text)
856|   If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
857|
858| If strOrderBy <> "" Then
859|     strOrderBy = " ORDER BY " & Left$(strOrderBy, Len(strOrderBy) - 2)
860|   End If
861|
862|   TempSQL = TempSQL & strOrderBy & " ; "
863|
864| ' Now put the SQL statement into the appropriate text box.
865| flgIgnoreMapSQLChange = True
866|   txtMapSQL = TempSQL
867| flgIgnoreMapSQLChange = False
868|
869| End Sub
870|
871| Sub ShowSample ()
872| On Error Resume Next
873|  picSample.Cls
874|  picSample.ForeColor = Val(cmb_zqColor.Text)
875|  picSample.DrawStyle = Val(cmb_zqStyle.Text)
876|  picSample.DrawWidth = Val(cmb_zqWidth.Text)
877|
878|  picSample.Line (0, 0)-(10, 5)
879|  picSample.Line -(10, 25)
880|  picSample.Line -(2, 4)
```

```
881|   picSample.Line -(4, 12)
882|   picSample.Line -(19, 24)
883| End Sub
884|
885| Sub txtCriteria_Change ()
886|   SetMapSQL
887| End Sub
888|
889| Sub txtCriteria_DragDrop (Source As Control, X As Single, Y As Single)
890| 'txtCriteria = Mid$(txtCriteria, 1, txtCriteria.SelStart) & " " &
     AddBracketsAsNeeded(CStr(Source)) & " " & Mid$(txtCriteria, txtCriteria.SelStart +
     txtCriteria.SelLength + 1, 32768)
891| txtCriteria = Mid$(txtCriteria, 1, txtCriteria.SelStart) & " [" & Source & "] " &
     Mid$(txtCriteria, txtCriteria.SelStart + txtCriteria.SelLength + 1, 32768)
892| 'Drops the selected text at the current insertion point
893| End Sub
894|
895| Sub txtDBFormSQL_DragDrop (Source As Control, X As Single, Y As Single)
896| Call DBFormSQLAddField(CStr(Source))
897| End Sub
898|
899| Sub txtMapSQL_KeyPress (KeyAscii As Integer)
900|
901| If flgIgnoreMapSQLChange Then Exit Sub
902| If KeyAscii <> 13 Then Exit Sub
903| KeyAscii = 0'Swallow the return key
904|  Dim strSQL As String
905|  strSQL = txtMapSQL
906|  Call DecodeMapSQL(strSQL)
907|
908|
909|
910| End Sub
911|
912| Sub txtSpatialQuerySQL_KeyPress (KeyAscii As Integer)
913| If KeyAscii = 13 Then
914|    KeyAscii = 0
915|    'Other decoding of the SQL statement should go here
916| End If
917|
918| End Sub
919|
920| Sub zzzlblFields_Click ()
921|  cmbFieldList.Drag
922| End Sub
923|
924| Sub zzzlblName_DragDrop (Source As Control, X As Single, Y As Single)
925| cmb_zqFeatureID.AddItem "[" & Source & "]"'Sets all of the following Z-field to whatever
     was dropped on it
926| cmb_zqFeatureID.ListIndex = cmb_zqFeatureID.ListCount - 1
927| End Sub
928|
929| Sub zzzlblX_DragDrop (Source As Control, X As Single, Y As Single)
930| cmb_zqXYBlobData = "[" & Source & "]"'Sets all of the following X-field to whatever was
```

```
      dropped on it
 931| End Sub
 932|
 933| Sub zzzlblZSize_DragDrop (Source As Control, X As Single, Y As Single)
 934|  'LineWidth = AddBracketsAsNeeded(CStr(Source)) 'Sets all of the following Z-field to
      whatever was dropped on it
 935|  cmb_zqWidth = "[" & Source & "]"  'Sets all of the following Z-field to whatever was
      dropped on it
 936| End Sub
 937|
 938| Sub zzzlblZSymbol_DragDrop (Source As Control, X As Single, Y As Single)
 939|  'txtZSymbol = AddBracketsAsNeeded(CStr(Source))'Sets all of the following Z-field to
      whatever was dropped on it
 940|  cmb_zqStyle.AddItem "[" & Source & "]" 'Sets all of the following Z-field to whatever was
      dropped on it
 941|  cmb_zqStyle.ListIndex = cmb_zqStyle.ListCount - 1
 942| End Sub
 943|
 944| Sub zzzzlblCriteia_DragDrop (Source As Control, X As Single, Y As Single)
 945| 'txtCriteria = AddBracketsAsNeeded(CStr(Source))
 946| txtCriteria = "[" & Source & "]"
 947| End Sub
 948|
 949| Sub zzzzlblDBFormSQL_DragDrop (Source As Control, X As Single, Y As Single)
 950| Call DBFormSQLAddField(CStr(Source))
 951| End Sub
 952|
 953| Sub zzzzzlblZColor_Click ()
 954|   Dim ColorWas As Long, NewColor As Long
 955|   ColorWas = Val(cmb_zqColor)
 956|   NewColor = GetColorWithCmnDialog(ColorWas)
 957|   If NewColor <> ColorWas Then AddAndHighlightItem Format$(NewColor),
      cmb_zqColor
 958|
 959| End Sub
 960|
 961| Sub zzzzzlblZColor_DragDrop (Source As Control, X As Single, Y As Single)
 962| AddAndHighlightItem "[" & Source & "]", cmb_zqColor 'Sets all of the following Z-field to
      whatever was dropped on it
 963| End Sub
 964|
```


Select LookUpTable
LookUpTable Index
Look Up Table Name
Table Or Query Name
Input Variable
Output Variable
LUT Type
Comments
OK
Cancel

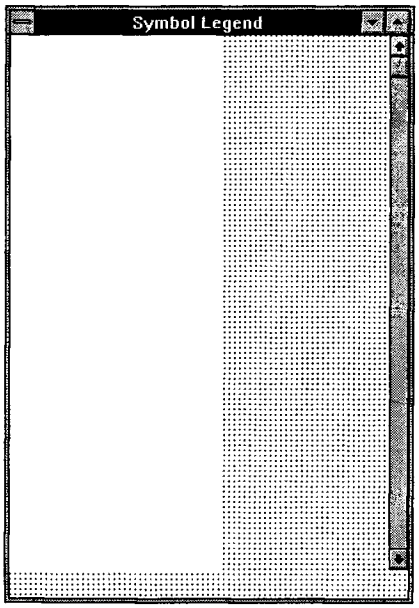
```
VERSION 2.00
Begin Form frmLUTIndex
  BorderStyle    = 3 'Fixed Double
  Caption        = "Select LookUpTable"
  ClientHeight   = 4020
  ClientLeft     = 1215
  ClientTop      = 840
  ClientWidth    = 6720
  Height         = 4425
  Left           = 1155
  LinkTopic      = "Form1"
  MaxButton      = 0 'False
  MinButton      = 0 'False
  ScaleHeight    = 4020
  ScaleWidth     = 6720
  Top            = 495
  Width          = 6840
  Begin CommandButton cmdCancel
    Caption        = "&Cancel"
    FontBold       = 0 'False
    FontItalic     = 0 'False
    FontName       = "MS Sans Serif"
    FontSize       = 8.25
    FontStrikethru = 0 'False
    FontUnderline  = 0 'False
    Height         = 315
    Left           = 3240
    TabIndex       = 13
    Top            = 3660
    Width          = 1335
  End
  Begin CommandButton cmdOK
    Caption        = "&OK"
    FontBold       = 0 'False
    FontItalic     = 0 'False
    FontName       = "MS Sans Serif"
    FontSize       = 8.25
    FontStrikethru = 0 'False
    FontUnderline  = 0 'False
    Height         = 315
    Left           = 1650
    TabIndex       = 12
    Top            = 3660
    Width          = 1395
  End
  Begin TextBox txt
    DataField      = "LookUpTableName"
    DataSource     = "Data1"
    FontBold       = 0 'False
    FontItalic     = 0 'False
    FontName       = "MS Sans Serif"
    FontSize       = 8.25
    FontStrikethru = 0 'False
    FontUnderline  = 0 'False
```

```
    Height      = 915
    Index       = 4
    Left        = 1650
    MultiLine   = -1 'True
    ScrollBars  = 2 'Vertical
    TabIndex    = 10
    Top         = 2700
    Width       = 4995
  End
  Begin TextBox txt
    DataField     = "LookUpTableType"
    DataSource    = "Data1"
    FontBold      = 0 'False
    FontItalic    = 0 'False
    FontName      = "MS Sans Serif"
    FontSize      = 8.25
    FontStrikethru = 0 'False
    FontUnderline = 0 'False
    Height        = 285
    Index         = 3
    Left          = 1650
    TabIndex      = 8
    Top           = 2340
    Width         = 5000
  End
  Begin TextBox txt
    DataField     = "OutputVariable"
    DataSource    = "Data1"
    FontBold      = 0 'False
    FontItalic    = 0 'False
    FontName      = "MS Sans Serif"
    FontSize      = 8.25
    FontStrikethru = 0 'False
    FontUnderline = 0 'False
    Height        = 285
    Index         = 2
    Left          = 1650
    TabIndex      = 6
    Top           = 1920
    Width         = 5000
  End
  Begin TextBox txt
    DataField     = "InputVariable"
    DataSource    = "Data1"
    FontBold      = 0 'False
    FontItalic    = 0 'False
    FontName      = "MS Sans Serif"
    FontSize      = 8.25
    FontStrikethru = 0 'False
    FontUnderline = 0 'False
    Height        = 285
    Index         = 1
    Left          = 1650
    TabIndex      = 4
```

```
109|     Top         = 1500
110|     Width       = 5000
111|   End
112|   Begin TextBox txt
113|     DataField     = "TableOrQueryName"
114|     DataSource    = "Data1"
115|     FontBold      = 0  'False
116|     FontItalic    = 0  'False
117|     FontName      = "MS Sans Serif"
118|     FontSize      = 8.25
119|     FontStrikethru = 0  'False
120|     FontUnderline = 0  'False
121|     Height        = 285
122|     Index         = 0
123|     Left          = 1650
124|     TabIndex      = 2
125|     Top           = 1080
126|     Width         = 5000
127|   End
128|   Begin TextBox txtLookUpTableName
129|     DataField     = "LookUpTableName"
130|     DataSource    = "Data1"
131|     FontBold      = 0  'False
132|     FontItalic    = 0  'False
133|     FontName      = "MS Sans Serif"
134|     FontSize      = 8.25
135|     FontStrikethru = 0  'False
136|     FontUnderline = 0  'False
137|     Height        = 285
138|     Left          = 1650
139|     TabIndex      = 0
140|     Top           = 600
141|     Width         = 5000
142|   End
143|   Begin Data Data1
144|     Caption       = "LookUpTable Index"
145|     Connect       = ""
146|     DatabaseName  = ""
147|     Exclusive     = 0  'False
148|     FontBold      = 0  'False
149|     FontItalic    = 0  'False
150|     FontName      = "MS Sans Serif"
151|     FontSize      = 8.25
152|     FontStrikethru = 0  'False
153|     FontUnderline = 0  'False
154|     Height        = 270
155|     Left          = 0
156|     Options       = 0
157|     ReadOnly      = 0  'False
158|     RecordSource  = "zWHEATz_LookupTableIndex"
159|     Top           = 180
160|     Width         = 6735
161|   End
162|   Begin Label lbl
```

```
163|     Caption      = "Comments"
164|     FontBold     = 0  'False
165|     FontItalic   = 0  'False
166|     FontName     = "MS Sans Serif"
167|     FontSize     = 8.25
168|     FontStrikethru = 0  'False
169|     FontUnderline  = 0  'False
170|     Height       = 255
171|     Index        = 4
172|     Left         = 60
173|     TabIndex     = 11
174|     Top          = 2940
175|     Width        = 1815
176|   End
177|   Begin Label lbl
178|     Caption      = "LUT Type"
179|     FontBold     = 0  'False
180|     FontItalic   = 0  'False
181|     FontName     = "MS Sans Serif"
182|     FontSize     = 8.25
183|     FontStrikethru = 0  'False
184|     FontUnderline  = 0  'False
185|     Height       = 255
186|     Index        = 3
187|     Left         = 60
188|     TabIndex     = 9
189|     Top          = 2340
190|     Width        = 1815
191|   End
192|   Begin Label lbl
193|     Caption      = "Output Variable"
194|     FontBold     = 0  'False
195|     FontItalic   = 0  'False
196|     FontName     = "MS Sans Serif"
197|     FontSize     = 8.25
198|     FontStrikethru = 0  'False
199|     FontUnderline  = 0  'False
200|     Height       = 255
201|     Index        = 2
202|     Left         = 60
203|     TabIndex     = 7
204|     Top          = 1920
205|     Width        = 1815
206|   End
207|   Begin Label lbl
208|     Caption      = "Input Variable"
209|     FontBold     = 0  'False
210|     FontItalic   = 0  'False
211|     FontName     = "MS Sans Serif"
212|     FontSize     = 8.25
213|     FontStrikethru = 0  'False
214|     FontUnderline  = 0  'False
215|     Height       = 255
216|     Index        = 1
```

```
217|     Left        = 60
218|     TabIndex     = 5
219|     Top         = 1500
220|     Width       = 1815
221|   End
222|   Begin Label lbl
223|     Caption     = "Table Or Query Name"
224|     FontBold    = 0  'False
225|     FontItalic  = 0  'False
226|     FontName    = "MS Sans Serif"
227|     FontSize    = 8.25
228|     FontStrikethru = 0  'False
229|     FontUnderline  = 0  'False
230|     Height      = 255
231|     Index       = 0
232|     Left        = 60
233|     TabIndex    = 3
234|     Top         = 1080
235|     Width       = 1815
236|   End
237|   Begin Label lblLUTName
238|     Caption     = "Look Up Table Name"
239|     FontBold    = 0  'False
240|     FontItalic  = 0  'False
241|     FontName    = "MS Sans Serif"
242|     FontSize    = 8.25
243|     FontStrikethru = 0  'False
244|     FontUnderline  = 0  'False
245|     Height      = 255
246|     Left        = 60
247|     TabIndex    = 1
248|     Top         = 600
249|     Width       = 1815
250|   End
251| End
252| Option Explicit
253|
254| Sub cmdCancel_Click ()
255|   Me.Hide
256|   Me!cmdOK.FontItalic = False
257| End Sub
258|
259| Sub cmdOK_Click ()
260|  Me.Hide
261|  Me!cmdOK.FontItalic = True
262| End Sub
263|
264| Sub Form_Paint ()
265|  Me!cmdOK.FontItalic = False
266| End Sub
267|
```


Symbol Legend

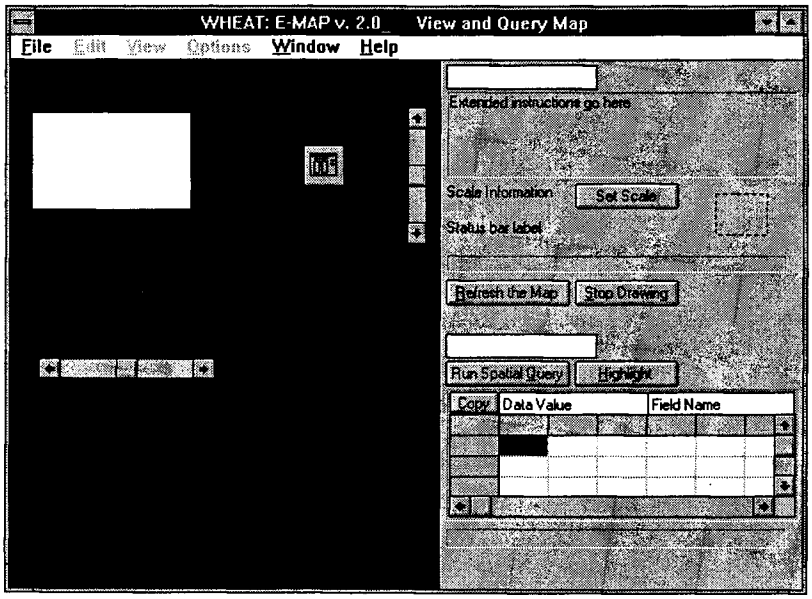
```
VERSION 2.00
Begin Form frmSymbolLegend
  Caption        = "Symbol Legend"
  ClientHeight   = 6585
  ClientLeft     = 1080
  ClientTop      = 465
  ClientWidth    = 4605
  Height         = 6990
  Left           = 1020
  LinkTopic      = "Form1"
  MDIChild       = -1 'True
  ScaleHeight    = 116.152
  ScaleMode      = 6 'Millimeter
  ScaleWidth     = 81.227
  Top            = 120
  Width          = 4725
  Begin VScrollBar VScroll
    Height       = 6255
    Left         = 4380
    TabIndex     = 1
    Top          = 0
    Width        = 225
  End
  Begin PictureBox picLegend
    AutoRedraw   = -1 'True
    BorderStyle  = 0 'None
    Height       = 6255
    Left         = 0
    ScaleHeight  = 110.331
    ScaleMode    = 6 'Millimeter
    ScaleWidth   = 43.127
    TabIndex     = 0
    Top          = 0
    Width        = 2445
  End
End
Option Explicit

Dim PointColor As Long, MouseX As Single, MouseY As Single
Dim R As Integer, G As Integer, B As Integer

Sub Form_Resize ()
  Me.picLegend.Width = Me.ScaleWidth - 5
  'Me.picLegend.Height = Me.ScaleHeight
  Me!VScroll.Left = Me.ScaleWidth - 4.5
  Me!VScroll.Height = Me.ScaleHeight
End Sub

Sub VScroll_Change ()
    Me!picLegend.Top = -VScroll
End Sub

```


WHEAT: E-MAP v. 2.0 View and Query Map
File Edit View Options Window Help
Extended instructions go here
Scale Information
Set Scale
Status bar label
Refresh the Map
Stop Drawing
Run Spatial Query
Highlight
Copy
Data Value
Field Name

```
VERSION 2.00
Begin Form frmMap
  BackColor      = &H00400000&
  Caption        = "WHEAT: E-MAP v. 2.0_    View and Query Map"
  ClientHeight   = 6165
  ClientLeft     = 120
  ClientTop      = 750
  ClientWidth    = 9180
  Height         = 6855
  Icon           = MAP005.FRX:0000
  Left           = 60
  LinkTopic      = "Form1"
  MDIChild       = -1 'True
  ScaleHeight    = 108.744
  ScaleMode      = 6 'Millimeter
  ScaleWidth     = 161.925
  Top            = 120
  Width          = 9300
  Begin CommonDialog CMDialog1
    Left         = 3375
    Top          = 990
  End
  Begin HScrollBar XScroll
    Height       = 240
    LargeChange  = 5
    Left         = 330
    Max          = 200
    TabIndex     = 2
    Top          = 3495
    Value        = 100
    Width        = 2025
  End
  Begin PictureBox picMap
    Height       = 1155
    Left         = 240
    ScaleHeight  = 1068
    ScaleMode    = 0 'User
    ScaleWidth   = 1183
    TabIndex     = 0
    Top          = 600
    Width        = 1845
  End
  Begin VScrollBar YScroll
    Height       = 1605
    LargeChange  = 5
    Left         = 4590
    Max          = 0
    Min          = 200
    TabIndex     = 1
    Top          = 540
    Value        = 100
    Width        = 227
  End
  Begin SSPanel pnl3dSide
```

```
   55|     BackColor      = &H00C0C0C0&
   56|     BorderWidth    = 1
   57|     Font3D         = 0 'None
   58|     ForeColor      = &H00FFFFFF&
   59|     Height         = 6465
   60|     Left           = 4980
   61|     TabIndex       = 11
   62|     Top            = -15
   63|     Width          = 4230
   64|     Begin CommandButton cmdSetScale
   65|       Caption      = "Set Scale"
   66|       FontBold     = 0 'False
   67|       FontItalic   = 0 'False
   68|       FontName     = "MS Sans Serif"
   69|       FontSize     = 8.25
   70|       FontStrikethru = 0 'False
   71|       FontUnderline = 0 'False
   72|       Height       = 300
   73|       Left         = 1560
   74|       TabIndex     = 20
   75|       Top          = 1440
   76|       Width        = 1200
   77|     End
   78|     Begin TextBox txtCurrCoords
   79|       FontBold     = 0 'False
   80|       FontItalic   = 0 'False
   81|       FontName     = "MS Sans Serif"
   82|       FontSize     = 8.25
   83|       FontStrikethru = 0 'False
   84|       FontUnderline = 0 'False
   85|       Height       = 285
   86|       Left         = 60
   87|       TabIndex     = 19
   88|       Top          = 60
   89|       Width        = 1755
   90|     End
   91|     Begin TextBox txtLastQSCoords
   92|       FontBold     = 0 'False
   93|       FontItalic   = 0 'False
   94|       FontName     = "MS Sans Serif"
   95|       FontSize     = 8.25
   96|       FontStrikethru = 0 'False
   97|       FontUnderline = 0 'False
   98|       Height       = 285
   99|       Left         = 60
  100|       TabIndex     = 18
  101|       Top          = 3210
  102|       Width        = 1755
  103|     End
  104|     Begin SSPanel pnlQSResults
  105|       Alignment    = 1 'Left Justify - MIDDLE
  106|       BackColor    = &H00C0C0C0&
  107|       BevelInner   = 1 'Inset
  108|       BevelOuter   = 0 'None
```

```
109|       BorderWidth    = 1
110|       Font3D         = 0 'None
111|       FontBold       = 0 'False
112|       FontItalic     = 0 'False
113|       FontName       = "MS Sans Serif"
114|       FontSize       = 8.25
115|       FontStrikethru = 0 'False
116|       FontUnderline  = 0 'False
117|       ForeColor      = &H00000000&
118|       Height         = 255
119|       Left           = 60
120|       TabIndex       = 16
121|       Top            = 5460
122|       Width          = 3975
123|     End
124|     Begin CommandButton cmdRunSpatialQuery
125|       Caption        = "Run Spatial &Query"
126|       FontBold       = 0 'False
127|       FontItalic     = 0 'False
128|       FontName       = "MS Sans Serif"
129|       FontSize       = 8.25
130|       FontStrikethru = 0 'False
131|       FontUnderline  = 0 'False
132|       Height         = 300
133|       Left           = 60
134|       TabIndex       = 4
135|       Top            = 3520
136|       Width          = 1440
137|     End
138|     Begin CommandButton cmdHighlightPosition
139|       Caption        = "&Highlight"
140|       FontBold       = 0 'False
141|       FontItalic     = 0 'False
142|       FontName       = "MS Sans Serif"
143|       FontSize       = 8.25
144|       FontStrikethru = 0 'False
145|       FontUnderline  = 0 'False
146|       Height         = 300
147|       Left           = 1560
148|       TabIndex       = 3
149|       Top            = 3525
150|       Width          = 1200
151|     End
152|     Begin SSPanel pnl3dInstructions
153|       Alignment      = 0 'Left Justify - TOP
154|       BackColor      = &H00C0C0C0&
155|       BevelInner     = 1 'Inset
156|       BevelOuter     = 0 'None
157|       BorderWidth    = 1
158|       Caption        = "Extended instructions go here"
159|       Font3D         = 0 'None
160|       FontBold       = 0 'False
161|       FontItalic     = 0 'False
162|       FontName       = "MS Sans Serif"
```

```
163|       FontSize      = 8.25
164|       FontStrikethru = 0  'False
165|       FontUnderline  = 0  'False
166|       ForeColor      = &H00000000&
167|       Height         = 1035
168|       Left           = 60
169|       TabIndex       = 15
170|       Top            = 360
171|       Width          = 4095
172|     End
173|     Begin SSPanel pnl3dPercentDone
174|       BackColor      = &H00C0C0C0&
175|       BevelInner     = 1 'Inset
176|       BevelOuter     = 0 'None
177|       BorderWidth    = 1
178|       Caption        = "Percentage Done"
179|       FloodType      = 1 'Left To Right
180|       Font3D         = 0 'None
181|       ForeColor      = &H00FFFFFF&
182|       Height         = 220
183|       Left           = 60
184|       TabIndex       = 13
185|       Top            = 2280
186|       Width          = 3975
187|     End
188|     Begin CommandButton cmdStopDrawing
189|       Caption        = "&Stop Drawing"
190|       FontBold       = 0  'False
191|       FontItalic     = 0  'False
192|       FontName       = "MS Sans Serif"
193|       FontSize       = 8.25
194|       FontStrikethru = 0  'False
195|       FontUnderline  = 0  'False
196|       Height         = 300
197|       Left           = 1560
198|       TabIndex       = 6
199|       Top            = 2580
200|       Width          = 1200
201|     End
202|     Begin CommandButton cmdRefreshMap
203|       Caption        = "&Refresh the Map"
204|       FontBold       = 0  'False
205|       FontItalic     = 0  'False
206|       FontName       = "MS Sans Serif"
207|       FontSize       = 8.25
208|       FontStrikethru = 0  'False
209|       FontUnderline  = 0  'False
210|       Height         = 300
211|       Left           = 60
212|       TabIndex       = 5
213|       Top            = 2580
214|       Width          = 1440
215|     End
216|     Begin SSPanel pnl3dGrid
```

```
217|       BackColor     = &H00C0C0C0&
218|       BevelInner    = 1 'Inset
219|       BevelOuter    = 0 'None
220|       BorderWidth   = 1
221|       Font3D        = 0 'None
222|       ForeColor     = &H00FFFFFF&
223|       Height        = 1560
224|       Left          = 60
225|       TabIndex      = 12
226|       Top           = 3840
227|       Width         = 4125
228|       Begin CommandButton cmdCopyDataGrid
229|         Caption     = "Copy"
230|         FontBold    = 0 'False
231|         FontItalic  = 0 'False
232|         FontName    = "MS Sans Serif"
233|         FontSize    = 8.25
234|         FontStrikethru = 0 'False
235|         FontUnderline = 0 'False
236|         Height      = 255
237|         Left        = 45
238|         TabIndex    = 10
239|         Top         = 45
240|         Width       = 570
241|       End
242|       Begin Grid grdMapData
243|         Cols        = 12
244|         FontBold    = 0 'False
245|         FontItalic  = 0 'False
246|         FontName    = "MS Sans Serif"
247|         FontSize    = 8.25
248|         FontStrikethru = 0 'False
249|         FontUnderline = 0 'False
250|         Height      = 1200
251|         Left        = 45
252|         Rows        = 4
253|         TabIndex    = 7
254|         Top         = 300
255|         Width       = 4020
256|       End
257|       Begin TextBox txtMapGridDataShow
258|         FontBold    = 0 'False
259|         FontItalic  = 0 'False
260|         FontName    = "MS Sans Serif"
261|         FontSize    = 8.25
262|         FontStrikethru = 0 'False
263|         FontUnderline = 0 'False
264|         Height      = 285
265|         Left        = 615
266|         TabIndex    = 8
267|         Text        = "Data Value"
268|         Top         = 45
269|         Width       = 1740
270|       End
```

```
271|       Begin TextBox txtMapGridFieldShow
272|         FontBold      = 0  'False
273|         FontItalic    = 0  'False
274|         FontName      = "MS Sans Serif"
275|         FontSize      = 8.25
276|         FontStrikethru = 0  'False
277|         FontUnderline = 0  'False
278|         Height        = 285
279|         Left          = 2340
280|         TabIndex      = 9
281|         Text          = "Field Name"
282|         Top           = 45
283|         Width         = 1710
284|       End
285|     End
286|     Begin Label lblScaleWords
287|       BackStyle      = 0 'Transparent
288|       Caption        = "Scale Information"
289|       FontBold       = 0  'False
290|       FontItalic     = 0  'False
291|       FontName       = "MS Sans Serif"
292|       FontSize       = 8.25
293|       FontStrikethru = 0  'False
294|       FontUnderline  = 0  'False
295|       Height         = 195
296|       Left           = 60
297|       TabIndex       = 17
298|       Top            = 1440
299|       Width          = 1635
300|     End
301|     Begin Image imgStatus
302|       Height         = 495
303|       Left           = 3180
304|       Top            = 1560
305|       Width          = 615
306|     End
307|     Begin Label lblPercentDone
308|       AutoSize       = -1 'True
309|       BackColor      = &H00C0C0C0&
310|       Caption        = "Status bar label"
311|       FontBold       = 0  'False
312|       FontItalic     = 0  'False
313|       FontName       = "MS Sans Serif"
314|       FontSize       = 8.25
315|       FontStrikethru = 0  'False
316|       FontUnderline  = 0  'False
317|       Height         = 555
318|       Left           = 60
319|       TabIndex       = 14
320|       Top            = 1860
321|       Width          = 3975
322|       WordWrap       = -1 'True
323|     End
324|   End
```

```
325|  Begin Menu zzFile
326|    Caption       = "&File"
327|    Begin Menu mfOpen
328|      Caption     = "Open &Database..."
329|    End
330|    Begin Menu mfClose
331|      Caption     = "&Close Database"
332|      Enabled     = 0  'False
333|    End
334|    Begin Menu zzzzz5
335|      Caption     = "-"
336|    End
337|    Begin Menu mfNewMapLayer
338|      Caption     = "&New Map Theme"
339|      Enabled     = 0  'False
340|      Begin Menu mfCreateLayerSymbols
341|        Caption   = "&Symbols..."
342|      End
343|      Begin Menu mfCreateLayerLines
344|        Caption   = "&Lines..."
345|      End
346|      Begin Menu mfCreateLayerTiles
347|        Caption   = "&Tiles..."
348|      End
349|      Begin Menu mfCreateLayerPolygons
350|        Caption   = "&Polygons..."
351|      End
352|      Begin Menu mfCreateLayerTextLabels
353|        Caption   = "&Text Labels..."
354|      End
355|      Begin Menu mfCreateLayerRasterGrid
356|        Caption   = "Raster &Grid..."
357|      End
358|    End
359|    Begin Menu mfOpenMapLayer
360|      Caption     = "&Open Map Theme..."
361|      Enabled     = 0  'False
362|    End
363|    Begin Menu mfSaveMapLayerAs
364|      Caption     = "&Save Map Theme..."
365|      Enabled     = 0  'False
366|    End
367|    Begin Menu zzzzz4
368|      Caption     = "-"
369|    End
370|    Begin Menu mfExportMapAs
371|      Caption     = "&Export Map As..."
372|      Enabled     = 0  'False
373|    End
374|    Begin Menu mfPrintMap
375|      Caption     = "&Print..."
376|      Enabled     = 0  'False
377|    End
378|    Begin Menu zzzz20
```

```
379|       Caption       = "-"
380|     End
381|     Begin Menu mfDisplayColorLegend
382|       Caption       = "Display Color Legend"
383|       Enabled       = 0  'False
384|     End
385|     Begin Menu zzzzz7
386|       Caption       = "-"
387|     End
388|     Begin Menu mfSpatialAnalysis
389|       Caption       = "Spa&tial Analysis..."
390|     End
391|     Begin Menu zzz16
392|       Caption       = "-"
393|     End
394|     Begin Menu mfExit
395|       Caption       = "E&xit"
396|     End
397|   End
398|   Begin Menu zzzEdit
399|     Caption       = "&Edit"
400|     Enabled       = 0  'False
401|     Begin Menu meCut
402|       Caption       = "Cu&t"
403|       Shortcut      = +{DEL}
404|       Visible       = 0  'False
405|     End
406|     Begin Menu meCopy
407|       Caption       = "&Copy"
408|       Shortcut      = ^{INSERT}
409|     End
410|     Begin Menu meCopyMap
411|       Caption       = "Copy &Map"
412|       Shortcut      = ^M
413|     End
414|     Begin Menu mePaste
415|       Caption       = "&Paste"
416|       Shortcut      = +{INSERT}
417|     End
418|     Begin Menu zzzz14
419|       Caption       = "-"
420|     End
421|     Begin Menu meLayers
422|       Caption       = "&Layers"
423|       Visible       = 0  'False
424|       Begin Menu melDefinition
425|         Caption       = "&Definition"
426|       End
427|     End
428|     Begin Menu meAdd
429|       Caption       = "&Add"
430|       Begin Menu meaPoints
431|         Caption       = "&Points"
432|       End
```

```
433|       Begin Menu meaLines
434|         Caption       = "&Lines"
435|       End
436|       Begin Menu meaPolygons
437|         Caption       = "&Polygons"
438|       End
439|       Begin Menu zzz15
440|         Caption       = "-"
441|       End
442|       Begin Menu meaStop
443|         Caption       = "&Stop adding features"
444|       End
445|     End
446|   End
447|   Begin Menu zzView
448|     Caption       = "&View"
449|     Enabled       = 0  'False
450|     Begin Menu mvSpecifyViewRegion
451|       Caption       = "Specify &View Region..."
452|     End
453|     Begin Menu zzzz12
454|       Caption       = "-"
455|     End
456|     Begin Menu mnuVZoomin
457|       Caption       = "Zoom &In"
458|       Shortcut      = ^I
459|     End
460|     Begin Menu mnuVZoomout
461|       Caption       = "Zoom &Out"
462|       Shortcut      = ^O
463|     End
464|     Begin Menu mvZoomByRectangle
465|       Caption       = "Zoom by Re&ctangle"
466|       Shortcut      = ^B
467|     End
468|     Begin Menu mvZoomWayOut
469|       Caption       = "Zoom all the &way out"
470|       Shortcut      = ^W
471|     End
472|     Begin Menu zz2
473|       Caption       = "-"
474|     End
475|     Begin Menu mnuVRight
476|       Caption       = "Shift &Right"
477|       Shortcut      = ^R
478|     End
479|     Begin Menu mnuVLeft
480|       Caption       = "Shift &Left"
481|       Shortcut      = ^L
482|     End
483|     Begin Menu mnuVUp
484|       Caption       = "Shift &Up"
485|       Shortcut      = ^U
486|     End
```

```
   Begin Menu mnuVDown
     Caption      = "Shift &Down"
     Shortcut     = ^D
   End
   Begin Menu zzz12
     Caption      = "-"
   End
   Begin Menu mvStopDrawing
     Caption      = "&Stop drawing"
     Shortcut     = ^S
   End
   Begin Menu moRefreshMap
     Caption      = "Re&fresh map"
     Shortcut     = ^F
   End
   Begin Menu zzzzzz
     Caption      = "-"
   End
   Begin Menu mvGraticule
     Caption      = "&Graticule Grid"
   End
   Begin Menu mvScaleBarMain
     Caption      = "Scale Bar"
   End
 End
 Begin Menu zzzzOptions
   Caption      = "&Options"
   Enabled      = 0  'False
   Begin Menu meThemeOrder
     Caption      = "Theme &Order..."
   End
   Begin Menu moThemesVisible
     Caption      = "Themes &Visible..."
   End
   Begin Menu zzz17
     Caption      = "-"
   End
   Begin Menu mqSpatial
     Caption      = "&Spatial Query..."
   End
   Begin Menu moShowLastQuery
     Caption      = "&Show Last Query Results"
     Checked      = -1  'True
   End
   Begin Menu zzzzz6
     Caption      = "-"
   End
   Begin Menu moAutoDrawOnViewShift
     Caption      = "Redraw Map &Whenever View Shifts"
     Checked      = -1  'True
   End
   Begin Menu moAutoredraw
     Caption      = "Keep Copy of Map in &Memory"
   End
```

```
541|      Begin Menu moUnloadFormsAfterUse
542|        Caption       =  "&Unload forms after use to free memory"
543|      End
544|      Begin Menu zzzzz8
545|        Caption       =  "-"
546|      End
547|      Begin Menu moReset
548|        Caption       =  "Rese&t"
549|        Shortcut      =  ^T
550|      End
551|      Begin Menu moMapBackColor
552|        Caption       =  "Map &Background Color"
553|      End
554|      Begin Menu moAutoExpandDatabounds
555|        Caption       =  "&Expand map area to include all data"
556|        Checked       =  -1  'True
557|      End
558|    End
559|    Begin Menu mnuWindow
560|      Caption         =  "&Window"
561|      WindowList      =  -1  'True
562|    End
563|    Begin Menu zmnuHelp
564|      Caption         =  "&Help"
565|      Begin Menu mnuHelp
566|        Caption       =  "&Contents"
567|        Index         =  0
568|      End
569|      Begin Menu mnuHelp
570|        Caption       =  "&Search on..."
571|        Index         =  1
572|      End
573|      Begin Menu mnuHelp
574|        Caption       =  "-"
575|        Index         =  2
576|      End
577|      Begin Menu mnuHelp
578|        Caption       =  "&How to Use Help"
579|        Index         =  3
580|      End
581|      Begin Menu mnuHelp
582|        Caption       =  "-"
583|        Index         =  4
584|      End
585|      Begin Menu mnuHelp
586|        Caption       =  "About EMapKGS2"
587|        Index         =  5
588|      End
589|    End
590|  End
591|  Option Base 1
592|  Option Explicit
593|
594|  'Begin Menu mnuHelp
```

```
595I '    Caption      = "&Help"
596I '  End
597I
598I
599I Dim n As Long, L As Long, M As Long
600I Dim QSRasterGridNumber As Long 'Stores the index of the raster grid to show values
     for on each mouse_move
601I Dim flgIgnoreNextClick As Integer, ReturnVal As Long, flgRunMacro As Integer,
     MacroName As String, KeysToSend() As String
602I Dim GDIColorList(1 To 256) As Long, nColorsUsed As Long
603I Dim VBColorList() As Long
604I Dim FrmTempName As String * 8
605I
606I Dim flgDrawGraticule As Long, GraticuleSize  As Single, GraticuleColor As Long,
     GraticuleThickness As Long
607I
608I
609I 'The array ThemeOrder holds a list of indexes into the Themes() array, indicating the
     order in
610I ' which themes are to be plotted.  Positive values of ThemeOrder indicate that the theme
     should be plotted.
611I ' Negative values of ThemeOrder indicate that the theme is currently invisible.
612I Dim ThemeOrder() As Long, nThemesShownHere As Long
613I
614I 'Variables for the spatial querying activities
615I Dim lngQSHighlightColor As Long, sngQSHighlightRadius As Single, LastQS As
     RealPoint
616I
617I Dim CurrentTool As Integer, PreviousTool As Integer       'Which of the above tools is in
     use.
618I Dim LastCursor  As Integer
619I
620I 'Last mouse location from MouseMove
621I Dim MouseX As Single, MouseY As Single, MousePt As RealPoint
622I
623I 'Scroll bar, and thus worldwindow, values
624I Dim LastScrollX As Single, NewScrollX As Single
625I Dim LastScrollY As Single, NewScrollY As Single
626I Dim flgDontMove As Integer, iflgAutoExpandViewBounds  As Integer
627I Dim flgSkipRepainting As Integer, flgLoadDIB As Integer
628I
629I
630I
631I Dim dynSpatialQuery As dynaset, dynQSResults As dynaset, QSTheme As typTheme
632I Dim lngSearchType As Long, SearchRadius As Single, lngSearchThemeID As Long
633I Dim QSDataOut() As typTaggedIDNum
634I
635I
636I Dim flgUpdateGrdTxt As Integer 'Indicate whether to update the text box associated with
     the grid (True)
637I             ' or just sit there during grd_ColRowChange events.
638I Dim ZoomRect As RealRect, flgDrawRectangle As Integer, flgSkipSpatialQuery As
     Integer
639I Dim StartX As Single, StartY As Single, OldX As Single, OldY As Single
```

```
640
641
642
643 'These were originally in VARDECL.bas, and global.
644 'They were moved here 940429 to allow them to be form-level, so
645 ' I can make this a MDI application more readily, if each Me
646 '  keeps track of it own view variables.
647 Dim CurrentSize As RealPoint 'the current view zone size
648 Dim MaxScaleFactor As Single, ScaleFactor As Single
649 Dim OldScaleFactor As Single  'the scale factor before a size change (the current one)
650 Dim NewScaleFactor As Single  ' the scale factor now that the or map has changed size
651 Dim MapBaseColor As Long
652 Dim FormAspect As Single ' (width/height ) for form
653
654
655
656
657
658 ' These should probably be moved to Me, so that
659 '   multiple instances of Me could occur.  GWP 940303
660 'They were move 940516
661 Dim ViewBounds As RealRect 'the current bounding rectangle of
662             ' the view region.  This changes during
663             ' zooms, pans, and when new data is plotted.
664 Dim QryBounds As RealRect  'For passing the geographic extents of queries back and
    forth
665 Dim PixelRect As APIRECT   'For performing the World to Pixel mappings.
666 Dim SpareAPIRect As APIRECT 'Might come in handy, and it's only eight bytes.
667
668
669 'Variables used to scale Real*4 numbers to GDI numbers, mainly in drawing
670 '  polygons and their ilk.
671 Dim OriginalScale As RealRect, XPMult As Single, YPMult As Single, XOffset As Single,
    YOffset As Single
672 Dim MinPixX As Integer, MinPixY As Integer, MaxPixX As Integer, MaxPixY As Integer
673
674
675
676 Dim strUserPointsOut() As String
677 Dim PointString As String', nUserPoints As Long
678 Dim strUserLinesOut() As String, strUserPolygonsOut() As String
679 Dim UserLine() As RealPoint, nPtsInLine As Long
680
681 Dim nUserPoints As Long, nUserLines As Long, nUserPolys As Long
682
683 Dim ViewPortRect As APIRECT
684
685
686 Dim idxOnClickOpenFile As Long  'The field number (index) of a field named
687             ' OnClickOpenFile in the current Spatail Query
688
689 Dim FormsToUnloadFirst() As form, nDependents As Long
690
691 Dim iDrawScaleBar As Integer
```

```
692|
693| Sub AdjustMapScaleView (BoundsBox As RealRect)
694| 'This subroutine  [AdjustMapScaleView] adjusts the scale and screen area to that
695| '  necessary to enclose all of the data.
696| Dim MinX As Single, MaxX As Single, MinY As Single, MaxY As Single
697| MinX = BoundsBox.Left
698| MinY = BoundsBox.Bottom
699| MaxX = BoundsBox.Right
700| MaxY = BoundsBox.Top
701|
702|
703|
704| ' The following subroutine adjusts the size
705| '  of the Data area (also known as the window in
706| '  many computer graphics texts) to the maximum necessary to include
707| '  the old data and the four points being passed it.
708| Call zMyAddDataChangeBounds(MinX, MinY, MaxX, MaxY)
709|
710| Dim a As Single
711| Dim B As Single
712|
713| ViewBounds.Left = DataBounds.Left
714| ViewBounds.Top = DataBounds.Top
715| ViewBounds.Right = ViewBounds.Left + MaxScaleFactor * Me.picMap.Width
716| ViewBounds.Bottom = ViewBounds.Top - MaxScaleFactor * Me.picMap.Height
717|
718| CurrentSize.Y = .5 * (ViewBounds.Top - ViewBounds.Bottom)
719| CurrentSize.X = .5 * (ViewBounds.Right - ViewBounds.Left)
720|
721| Me.picMap.Scale (ViewBounds.Left, ViewBounds.Top)-(ViewBounds.Right,
     ViewBounds.Bottom)
722| a = (ViewBounds.Right - ViewBounds.Left) / (Me.picMap.Width)
723| B = (ViewBounds.Top - ViewBounds.Bottom) / (Me.picMap.Height)
724|
725|
726| If a > B Then
727|    OldScaleFactor = a
728|   Else
729|    OldScaleFactor = B
730| End If
731| NewScaleFactor = OldScaleFactor
732| ScaleFactor = OldScaleFactor
733|  ' dDebug.Print ScaleFactor
734|
735| End Sub
736|
737| Sub ClipPaste ()
738| ' Declare local variables
739| Dim ClipText As String
740| ClipText = Clipboard.GetText()
741| Dim nRows As Long
742| Dim nCols As Long
743| Dim TabChar As String * 1
744| Dim CR As String * 1
```

```
745  CR = Chr$(13)
746  TabChar = Chr$(9)
747  Dim n As Long
748  ' Find number of rows (delimeted by asc(13))
749  For n = 1 To Len(ClipText)
750    If Mid$(ClipText, n, 1) = CR Then
751      nRows = nRows + 1
752    End If
753    If Mid$(ClipText, n, 1) = TabChar Then
754      nCols = nCols + 1
755    End If
756  Next n
757    If nRows = 0 Then
758     nRows = 1
759    End If
760    If nCols = 0 Then
761      nCols = 1
762    End If
763
764    nCols = nCols \ nRows
765
766    grdMapData.SelEndCol = grdMapData.SelStartCol + nCols
767    grdMapData.SelEndRow = grdMapData.SelStartRow + nRows - 1
768
769      grdMapData.Clip = ClipText
770  End Sub
771
772  Sub cmdCopyDataGrid_Click ()
773   CopyQSResultsToClipboard
774  End Sub
775
776  Sub cmdHighlightPosition_Click ()
777
778  If nThemes = 0 Then
779      Beep
780      Exit Sub
781  End If
782  Dim ShowX As Single, ShowY As Single
783  Dim Msg As String, answer As Integer
784
785  ' First, get the desired location
786   Msg = txtLastQSCoords
787   Call ClipTextToXYPair(Msg, ShowX, ShowY)
788
789
790  'xxx1  Quick modification GWP 950227  It no longer offers to shift the
791  '  view, just doesn't plot anything if it is outside the view region
792  '  and beeps at you.
793
794  'Second, check to see whether the specified coordinates lie
795  '   within the current view_window.
796  If ShowX < ViewBounds.Left Or ShowX > ViewBounds.Right Or ShowY <
       ViewBounds.Bottom Or ShowY > ViewBounds.Top Then
797     Beep   'xxx1
```

```
798|    Exit Sub 'xxx1
799|
800|    Msg = "That point is outside the current view range." & CRLF & "(The area displayed
       on the screen right now does not include that point.)" & CRLF
801|    Msg = Msg & "Do want to adjust the view range to include the point?"
802|    answer = MsgBox(Msg, 36, "Problem showing highlight point")
803|    If answer = 7 Then Exit Sub
804|    If answer = 6 Then 'user chose YES
805|    'Third, check that the specified coordinates lie within DataBounds
806|    If ShowX < DataBounds.Left Or ShowX > DataBounds.Right Or ShowY <
       DataBounds.Bottom Or ShowY > DataBounds.Top Then
807|       Msg = "That point is outside the range of your current dataset. " & CRLF
808|       Msg = Msg & "Are you sure you want to show it anyway?"
809|       answer = MsgBox(Msg, 36, "")
810|       If answer = 7 Then Exit Sub
811|    End If
812|    Call ZZoom(ShowX - CurrentSize.X, ShowY + CurrentSize.Y, 1#)
813|    Call cmdRefreshMap_Click
814|    End If
815| End If
816|
817|
818| Dim DDrawMode As Integer, DDrawStyle As Integer, PhilColor As Long, PhilStyle As
       Integer
819|
820|
821| PhilColor = picMap.FillColor
822| PhilStyle = picMap.FillStyle
823| DDrawMode = picMap.DrawMode
824| DDrawStyle = picMap.DrawStyle
825|
826| Me!picMap.DrawStyle = 6
827| Me!picMap.DrawMode = 7
828| Me!picMap.FillStyle = 0
829| Me!picMap.FillColor = RGB(255, 255, 255)
830| Me!picMap.Circle (ShowX, ShowY), CurrentSize.X * .075, RGB(255, 255, 255),
       -.00000000001, -1.570796326795
831| Me!picMap.Circle (ShowX, ShowY), CurrentSize.X * .075, RGB(255, 255, 255),
       -3.14159265359, -4.712388980385
832|
833|
834| Me!picMap.FillColor = PhilColor
835| Me!picMap.FillStyle = PhilStyle
836| Me!picMap.DrawMode = DDrawMode
837| Me!picMap.DrawStyle = DDrawStyle
838|
839|
840| End Sub
841|
842| Sub cmdRefreshMap_Click ()
843| flgStopDrawing = True'Kill any current drawing
844| DoEvents
845| flgStopDrawing = False
846| Call ShowThemes
```

```
847| flgStopDrawing = False 'Re-enable automatic drawing
848| End Sub
849|
850| Sub cmdRunSpatialQuery_Click ()
851|  'Might want this to trigger a MOVE_MOUSE_TO_GRID_COORDINATES event.
852|  Dim tmp As String
853|  tmp = txtLastQSCoords
854|  Call ClipTextToXYPair(tmp, MouseX, MouseY)
855|
856| Call picMap_DblClick
857|
858| End Sub
859|
860| Sub cmdSetScale_Click ()
861|    "Let the user set the scale by entering a stipulated scale
862|    " in an input box.  Maybe improve this to include a drop-down list of standard scales
863|    If Me!picMap.ScaleMode <> 0 Then Exit Sub'if not in the user defined scale mode,
     leave
864|
865|    Call cmdStopDrawing_Click
866|    DoEvents
867|
868|    Dim sngNewScale As Single, XCenter As Single, YCenter As Single, dX As Single, dY
     As Single
869|    Dim tmpString As String
870|    tmpString = MapScaleInWords(Me!picMap.ScaleWidth, Me!picMap.Width)
871|    tmpString = InputBox$("Enter the desired scale factor", "Setting scale factor",
     tmpString)
872|    tmpString = ReplaceString(tmpString, ",", "")
873|    sngNewScale = Val(tmpString)
874|
875|
876|    If sngNewScale = 0 Then Exit Sub
877|
878| 'Actual MapScale is
879| 'MapScale = ScaleFudgeFactor() * (X_World * MapUnitsToMeters_Multiplier() * 1000) /
     (X_Screen_mm)
880| '
881| '  MapScale = ScaleFudgeFactor() * (X_World * MapUnitsToMeters_Multiplier() * 1000) /
     (X_Screen_mm)
882| '
883| '
884| '
885| '  (ScaleFactor is a local variable updated on every view shift)
886|
887|   Dim sngMultiplier As Single
888|   sngMultiplier = .001 / (ScaleFudgeFactor() * MapUnitsToMeters_Multiplier()) *
     sngNewScale
889|
890|   'Also need to adjust Scalefactor
891|    XCenter = .5 * (ViewBounds.Right + ViewBounds.Left)
892|    YCenter = .5 * (ViewBounds.Top + ViewBounds.Bottom)
893|    dX = .5 * Me!picMap.Width * sngMultiplier
894|    dY = .5 * Me!picMap.Height * sngMultiplier
```

```
895|     ViewBounds.Left = XCenter - dX
896|     ViewBounds.Right = XCenter + dX
897|     ViewBounds.Top = YCenter + dY
898|     ViewBounds.Bottom = YCenter - dY
899|     ScaleFactor = (ViewBounds.Right - ViewBounds.Left) / (Me.picMap.Width)
900|     Call ZZAdjustMapScale
901|     Call cmdRefreshMap_Click
902|
903| End Sub
904|
905| Sub cmdStopDrawing_Click ()
906|  flgStopDrawing = True
907|  'DoEvents
908|  'flgStopDrawing = False
909| End Sub
910|
911| Sub comments ()
912| ' nothing actually happens here, just a conventient place
913| ' for strategic comment cards.
914|
915| ' Basic plan.
916| ' 1) there is one huge array of points which contains the
917| '    the location of everything.  This array is called
918| '    rpt(s1,s2), where rpt is an array of Real PoinTs,
919| '    s1 goes from 0 to 8191, and s2 goes from 0 to nblocks -1
920| '    the reason for this is that array subscripts are integer*2,
921| '   and we wish to add blocks of data to this array while
922| '   preserving the current contents.  Thus, we initially
923| '   redim rpt(0 to 8191,0), which uses 64 KB to store 8192 points.
924| '   every time we need more space, we increment the size of the
925| '    the array by 64 KB with REDIM preserve rpt(8196, nblocks+1)
926| '
927| ' 2) there is one huge array of descriptions of the points
928| '    called its().  This is an array of structures, and each
929| '    structure contains a serial number,
930| '    a feature type (ie, point, line, polygon .....)
931| '    a feature code (like for primary highway, secondary highway)
932| '    the location of the first point in the element
933| '    the number of points in the element.
934| '
935| ' All map forms display the same data, but different views
936| '  are accomodated by having different view regions (a form property)
937| '  and different legends.
938| '  ? Possibly, there should be one big legend with
939| '          pointers into the big legend for each map sheet?
940| '
941| '  Map data can be pasted in from outside using a spreadsheet-like
942| '   import function, classes of features can be output
943| '   to mapdatasheet, as can features selected.
944| '  These data can be stored in either ascii or binary format.
945| '  Legends can also be stored in ascii or binary format.
946| '
947| ' A map file would then consist of a set of xy locations,
948| '  a set of feature descriptions indicating which points are
```

```
 949| ' associated with what features and how,
 950| ' a legend describing how to plot the features.
 951|
 952| ' Perhaps some allowance needs to be made for a feature-attribute
 953| ' table internal storage, or a companion database? that would
 954| ' have DDE or OLE dovetailed into this.  XXX This problem solved by
 955| '   buying VB 3.0
 956|
 957| End Sub
 958|
 959| Sub CopyQSResultsToClipboard ()
 960|    ' Declare local variables
 961|    Dim ClipText As String
 962| If (grdMapData.SelStartCol <> grdMapData.SelEndCol) Then
 963|   grdMapData.FixedRows = 0
 964|   grdMapData.SelStartRow = 0
 965| End If
 966|    ClipText = grdMapData.Clip
 967| grdMapData.FixedRows = 1
 968| grdMapData.SelStartRow = 1
 969|    Clipboard.Clear
 970|    Clipboard.SetText ClipText
 971| End Sub
 972|
 973| Sub CopyTextToAreaOfGrid ()
 974| Dim n1 As Integer
 975| Dim n2 As Integer
 976| Dim junktxt As String
 977| junktxt = txtMapGridDataShow
 978|
 979| For n1 = grdMapData.SelStartRow To grdMapData.SelEndRow
 980|    grdMapData.Row = n1
 981|    For n2 = grdMapData.SelStartCol To grdMapData.SelEndCol
 982|      grdMapData.Col = n2
 983|      grdMapData.Text = junktxt
 984|    Next n2
 985|
 986| Next n1
 987|
 988| End Sub
 989|
 990| Sub DrawGlobalHatchLine (hGMemAllPoints As Integer, nStart As Long, nPoints As
      Long)
 991|  If nPoints = 0 Then Exit Sub
 992| On Error Resume Next
 993|  ReDim RlPts(1 To nPoints) As RealPoint
 994|  Dim n  As Long
 995|    Call GetRlPntsFromGlobalMem(hGMemAllPoints, RlPts(1), nStart, nPoints)
 996|    For n = 1 To (nPoints - 1) Step 2
 997|   picMap.Line (RlPts(n).X, RlPts(n).Y)-(RlPts(n + 1).X, RlPts(n + 1).Y)
 998|    Next n
 999| End Sub
1000|
1001| Sub DrawGlobalLine (hGMemAllPoints As Integer, nStart As Long, nPoints As Long)
```

```
1002|  ReDim RlPts(1 To nPoints) As RealPoint
1003|  On Error GoTo DrawGlobalLineErr
1004|  Dim n  As Long
1005|  Dim flgExitSub As Integer
1006|     Call GetRlPntsFromGlobalMem(hGMemAllPoints, RlPts(1), nStart, nPoints)
1007|     picMap.CurrentX = RlPts(1).X
1008|     'If flgExitSub Then Exit Sub
1009|     picMap.CurrentY = RlPts(1).Y
1010|     'If flgExitSub Then Exit Sub
1011|     For n = 2 To nPoints
1012|    picMap.Line -(RlPts(n).X, RlPts(n).Y)
1013|    'If flgExitSub Then Exit Sub
1014|     Next n
1015|  Exit Sub
1016|  DrawGlobalLineErr:
1017|  'MsgBox "Error number " & Err & " occurred in DrawGlobalLine." & CRLF & Error
1018|  'Beep
1019|
1020|  If Err = 6 Then 'An overflow condition occurred , probably due to VBasic's mapping of
       coordinates.
1021|    flgExitSub = True
1022|    Resume Next
1023|  End If
1024|  End Sub
1025|
1026|  Sub DrawGlobalPolygon (hGMemAllPoints As Integer, nStart As Long, nPoints As Long,
       Flag As Long)
1027|  'GWPouch 940613
1028|  'Modified from my original DrawR4Polygon, in that this one does not
1029|  '  get its own scaling.  The calling routine should make sure the
1030|  ' scalemode of picMap has already been set to 3-pixel, and
1031|  ' that the formlevel variables XOffset,YOffset,XPMult,YPMult
1032|  ' have all been set.
1033|  'After the caller is done, the scale mode should be restored by the caller.
1034|  'Flag indicate whether to draw a filled polygon,  1
1035|  ' its outline                                    2
1036|  ' or both.                                      3
1037|  'Fillcolor, FillStyle, DrawWidth, and Outline colors should all be set by the caller.
1038|
1039|  If nPoints < 3 Then Exit Sub
1040|
1041|  On Error GoTo DrawGlobalPolyErr
1042|  ReDim PolyIn(1 To nPoints) As RealPoint, GDIPoly(1 To nPoints) As APIPoint
1043|  Call GetRlPntsFromGlobalMem(hGMemAllPoints, PolyIn(1), nStart, nPoints)
1044|  Dim Temp As Single
1045|  Dim n As Long
1046|  For n = 1 To nPoints
1047|       Temp = (PolyIn(n).X + XOffset) * XPMult
1048|       If Temp < MinPixX Then Temp = MinPixX
1049|       If Temp > MaxPixX Then Temp = MaxPixX
1050|     GDIPoly(n).X = Temp
1051|     'GDIPoly(n).X = (PolyIn(n).X + XOffset) * XPMult
1052|      Temp = (PolyIn(n).Y + YOffset) * YPMult
1053|      If Temp < MinPixY Then Temp = MinPixY
```

```
1054|     If Temp > MaxPixX Then Temp = MaxPixY
1055|     'GDIPoly(n).Y = (PolyIn(n).Y + YOffset) * YPMult
1056|     GDIPoly(n).Y = Temp
1057| Next n
1058|
1059|
1060| Dim jPoints
1061| jPoints = nPoints
1062|
1063| If Flag = 1 Or Flag = 3 Then
1064|    SetPolyFillMode Me.picMap.hDC, ALTERNATE
1065|    ReturnVal = Polygon(Me.picMap.hDC, GDIPoly(1), jPoints)
1066| End If
1067|
1068| If Flag = 2 Or Flag = 3 Then ReturnVal = Polyline(Me.picMap.hDC, GDIPoly(1), jPoints)
1069|
1070|
1071|
1072| Exit Sub
1073|
1074|
1075| DrawGlobalPolyErr:
1076|       'See if I can cleverly trap overflow and underflow errors.
1077| MsgBox "Error nubmer " & Err & " in DrawGlobalPolygon." & Error
1078| Err = 0
1079| Exit Sub
1080| Resume
1081|
1082|
1083| End Sub
1084|
1085| Sub DrawLine (LinesIn As LineDeclPlain, ZWidth As Integer, ZColor As Long, ZStyle As
      Integer)
1086|
1087|   Me.picMap.DrawWidth = ZWidth
1088|   Me.picMap.ForeColor = ZColor
1089|   Me.picMap.DrawStyle = ZStyle
1090|   Call DrawGlobalLine(hGMemAllPoints, LinesIn.nStart, LinesIn.nPoints)
1091|
1092| End Sub
1093|
1094| Sub DrawR4PolygonPrepare ()
1095| SaveScaling Me.picMap, OriginalScale
1096| GetScalingWorldToGDI Me.picMap, XOffset, YOffset, XPMult, YPMult
1097|
1098| End Sub
1099|
1100| Function FindClosestGlobalPoint (X As Single, Y As Single, nStart As Long, nEnd As
      Long, hGMemAllPoints As Integer) As Long
1101|  Dim MinDist As Single, Dist As Single, ClosestPt As RealPoint
1102|  Dim n As Long, nClosest As Long, M  As Long, L As Long
1103|  Dim nPoints As Long, nChunks As Long, nCh As Long
1104|  Dim nPos As Long
1105|  MinDist = 1E+34
```

```
1106|   nPoints = nEnd - nStart + 1
1107|   nChunks = nPoints \ 4096
1108|   nPos = 0
1109|   ReDim Pts(1 To 4096) As RealPoint
1110|   For nCh = 1 To nChunks
1111|     Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pts(1), nStart + nPos, 4096)
1112|       For M = 1 To 4096
1113|    Dist = (X - Pts(M).X) * (X - Pts(M).X) + (Y - Pts(M).Y) * (Y - Pts(M).Y)
1114|    If Dist < MinDist Then
1115|      MinDist = Dist
1116|      nClosest = M + nPos
1117|      LSet ClosestPt = Pts(M)
1118|    End If
1119|       Next M
1120|       nPos = nPos + 4096
1121|   Next nCh
1122|   L = nPoints - nPos
1123|   ReDim Points(1 To L)
1124|   Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pts(1), nStart + nPos, L)
1125|       For M = 1 To L
1126|    Dist = (X - Pts(M).X) * (X - Pts(M).X) + (Y - Pts(M).Y) * (Y - Pts(M).Y)
1127|    If Dist < MinDist Then
1128|      MinDist = Dist
1129|      nClosest = M + nPos
1130|      LSet ClosestPt = Pts(M)
1131|    End If
1132|       Next M
1133| FindClosestGlobalPoint = nClosest 'This is its location in the global point list.
1134|
1135|
1136|
1137| End Function
1138|
1139| Sub Form_DblClick ()
1140| ShowThemes
1141| End Sub
1142|
1143| Sub Form_Load ()
1144| 'Parameters for highlighting a spatial query
1145|     lngQSHighlightColor = DefaultHighLightColor
1146|     sngQSHighlightRadius = DefaultHighLightRadius
1147|
1148|
1149| GraticuleColor = RGB(0, 0, 128): GraticuleThickness = 1
1150|
1151|
1152|
1153|
1154| 'Generate a name for this form from the system time.
1155|   FrmTempName = "~wht" & Format$(Hex$(Time * 32768), "0000")
1156|
1157| 'Set the cursor to an arrow
1158|   ToolPick(-1).hCursor = OriginalCursorStyle
1159|   CurrentTool = -1
```

```
1160|  ViewBounds = DataBounds
1161|  ReDim UserLine(1)
1162|
1163|
1164|  AdjustMapScaleView DataBounds
1165|
1166|  MapBaseColor = APPLICATION_WORKSPACE
1167|  Dim n As Integer
1168|  LastScrollX = XScroll
1169|  LastScrollY = YScroll
1170|
1171|
1172|  'Set row labels for the data grid
1173|  grdMapData.Col = 0
1174|  grdMapData.Row = 0
1175|  grdMapData = "Field"
1176|  grdMapData.Row = 1
1177|  grdMapData = "Value"
1178|
1179|  Screen.MousePointer = 11'Set to HourGlass
1180|
1181|
1182|  Screen.MousePointer = 0'Restore default pointer
1183|
1184|  'Initialize 3D panel colors.
1185|  pnl3DGrid.BackColor = &HC0C0C0
1186|  pnl3dInstructions.BackColor = &HC0C0C0
1187|  pnl3dPercentDone.BackColor = &HC0C0C0
1188|    'pnl3DPosition.BackColor = &HC0C0C0
1189|  pnl3DSide.BackColor = &HC0C0C0
1190|  pnlQSResults.BackColor = &HC0C0C0
1191|  End Sub
1192|
1193|  Sub Form_Paint ()
1194|   flgStopDrawing = False
1195|
1196|  End Sub
1197|
1198|  Sub Form_QueryUnload (Cancel As Integer, UnloadMode As Integer)
1199|  MouseX = SetClassWord(Me.picMap.hWnd, GCW_HCURSOR, OriginalCursorStyle)
1200|
1201|   Call myUnloadDependentForms
1202|   'Unload frmMapScale: Set frmMapScale = Nothing
1203|
1204|
1205|
1206|
1207|
1208|   If nCursors > 0 Then
1209|     ZapCursors
1210|     ZapIcons
1211|
1212|   End If
1213|
```

```
1214  ShutDownAndFreeMem
1215  Do While (GlobalUnlock(hGMemAllPoints)): Loop
1216  Do While (GlobalFree(hGMemAllPoints)): Loop
1217  '940621  Do While (GlobalFree(hGMemGDIPoints))
1218  '940621  Loop
1219
1220
1221  'MouseX = GlobalFree(hGMemAllPoints)
1222  If Dir(FrmTempName & ".DIB") <> "" Then Kill FrmTempName & ".DIB"
1223  End Sub
1224
1225  Sub Form_Resize ()
1226  If Me.WindowState = 1 Then Exit Sub 'kludge to avoid resizing
1227            ' when the form is iconized
1228
1229
1230  FormAspect = (Me.ScaleWidth - 5) / (Me.ScaleHeight - 5)
1231  Dim picMapAspect As Single
1232
1233  If FormAspect < DataAspect Then
1234     picMapAspect = FormAspect
1235    Else
1236     picMapAspect = DataAspect
1237  End If
1238
1239  Me.picMap.Top = 0
1240  Me.picMap.Height = Me.ScaleHeight - 5
1241  Me.picMap.Left = 0
1242  Me.picMap.Width = Me.picMap.Height ' * picMapAspect
1243                  'Modified 940712 in kludegey attempt
1244                  ' to avoid resizing problems.  Trouble
1245                  'arises because the form tries using dataaspect
1246               ' when the view area is zoomed in.
1247
1248  'ViewBounds.Right = ViewBounds.Left + OldScaleFactor * Me.picMap.Width
1249  'ViewBounds.Bottom = ViewBounds.Top - OldScaleFactor * Me.picMap.Height
1250  '
1251  'Me.picMap.ScaleMode = 0
1252  'Me.picMap.Scale (ViewBounds.Left, ViewBounds.Top)-(ViewBounds.Right,
      ViewBounds.Bottom)
1253
1254  Me.YScroll.Top = 0
1255  Me.YScroll.Height = Me.picMap.Height
1256  Me.YScroll.Left = Me.picMap.Width + .25
1257  Me.YScroll.Width = 4
1258  Dim RightEdgeOfMapoid As Single
1259  RightEdgeOfMapoid = Me.YScroll.Left + Me.YScroll.Width + .5
1260  pnl3DSide.Left = RightEdgeOfMapoid
1261  pnl3DSide.Width = Me.ScaleWidth - RightEdgeOfMapoid
1262  pnl3DSide.Height = Me.ScaleHeight '- 5.5
1263  'Because 3DPanels don't get resize events, must set the widths and heights here
1264  ' of the daughter elements of this panel.
1265  Dim Panel3DWidth As Single
1266  Panel3DWidth = pnl3DSide.Width * 57.16 - 125
```

```
1267| pnl3dPercentDone.Width = Panel3DWidth
1268| lblPercentDone.Width = Panel3DWidth
1269| pnl3dInstructions.Width = Panel3DWidth
1270| pnlQSResults.Width = Panel3DWidth
1271|
1272| txtCurrCoords.Width = Panel3DWidth
1273| txtLastQSCoords.Width = Panel3DWidth
1274|
1275|
1276| 'pnl3DPosition.Width = Panel3DWidth
1277| 'grdPosition.Width = pnl3DPosition.Width - 90
1278| 'grdPosition.ColWidth(0) = grdPosition.Width / 2
1279| 'grdPosition.ColWidth(1) = grdPosition.Width / 2
1280|
1281| pnl3DGrid.Left = 50
1282| pnl3DGrid.Width = Panel3DWidth
1283| 'pnl3DGrid.Top = pnl3DSide.Height * 57.16 - 100 - pnl3DGrid.Height
1284|
1285| grdMapData.Width = pnl3DGrid.Width - 100
1286|
1287|
1288|
1289| txtMapGridDataShow.Top = 50
1290|
1291| txtMapGridDataShow.Width = .6 * (grdMapData.Width) '- txtMapGridDataShow.Left)
1292| txtMapGridFieldShow.Left = txtMapGridDataShow.Width + txtMapGridDataShow.Left
1293| txtMapGridFieldShow.Width = grdMapData.Width - txtMapGridDataShow.Width -
    txtMapGridDataShow.Left + 50
1294|
1295| grdMapData.Left = 50
1296|
1297|  Me.XScroll.Left = 0
1298|  Me.XScroll.Width = Me.picMap.Width
1299|  Me.XScroll.Top = Me.picMap.Height + .25
1300|  Me.XScroll.Height = 4
1301|
1302| 'End If 'end kludgey if loop on not an icon
1303|
1304| End Sub
1305|
1306| Sub Form_Unload (Cancel As Integer)
1307| MouseX = SetClassWord(Me.picMap.hWnd, GCW_HCURSOR, OriginalCursorStyle)
1308|
1309|  Call myUnloadDependentForms
1310|
1311|  If nCursors > 0 Then
1312|    ZapCursors
1313|    ZapIcons
1314|    MouseX = SetClassWord(Me.picMap.hWnd, GCW_HCURSOR, OriginalCursorStyle)
1315|  End If
1316|  ShutDownAndFreeMem
1317|  Do While (GlobalUnlock(hGMemAllPoints)): Loop
1318|  Do While (GlobalFree(hGMemAllPoints)): Loop
1319|  '  MouseX = GlobalFree(hGMemAllPoints)
```

```
1320|  End
1321|  End Sub
1322|
1323|  Sub grdMapData_RowColChange ()
1324|   'every time a new cell is selected, the text appears
1325|   ' in textbox text1 and text1 receives focus.
1326|   'Updates of text1 automatically appear in the grid.
1327|
1328|
1329|  Dim NTemp As Integer
1330|
1331|  If flgUpdateGrdTxt Then
1332|   flgUpdateGrdTxt = False
1333|   txtMapGridDataShow = grdMapData.Text
1334|   NTemp = grdMapData.Row
1335|   grdMapData.Row = 0
1336|   txtMapGridFieldShow = grdMapData.Text
1337|   grdMapData.Row = NTemp
1338|   flgUpdateGrdTxt = True
1339|  End If
1340|
1341|  End Sub
1342|
1343|  Sub IntializeMacro (MacroFileName As String)
1344|  'For GMD2's data-hydrograph
1345|  If MacroFileName = "" Then Exit Sub
1346|  On Error GoTo ERRStartMacro
1347|  Dim MacNumber As Integer, Message As String
1348|  MacNumber = FreeFile
1349|  Open MacroFileName For Input As MacNumber
1350|     Dim nLines As Long
1351|     nLines = 1
1352|     Do Until EOF(1)
1353|    ReDim Preserve KeysToSend(1 To nLines)
1354|    Line Input #MacNumber, KeysToSend(nLines)
1355|    nLines = nLines + 1
1356|     Loop
1357|     nLines = nLines - 1
1358|     Close MacNumber
1359|
1360|     AppActivate KeysToSend(1)
1361|     AppActivate MDI_WHEAT.Caption
1362|  Exit Sub
1363|  ERRStartMacro:
1364|     Select Case Err
1365|    Case 5 'Could not find application
1366|    Beep
1367|    Beep
1368|    Message = "You chose to run a macro, stored in " & UCase$(MacroFileName) & " ,
       everytime you run a spatial query, probably to graph the data." & CRLF
1369|    Message = Message & "E-Map cannot find the program-data file specified as " &
       KeysToSend(1) & CRLF
1370|    Message = Message & "Before proceeding, you must start the other application and
       possibly open the data file."
```

```
    MsgBox Message
    Resume
      Case 53 'File not found
    MacroFileName = InputBox$("You chose to run a macro in a file called " &
UCase$(MacroFileName) & " and E-Map cannot find it." & CRLF & "Please enter the
correct name, or hit cancel to not run the macro.", "Error finding file", MacroFileName)
    If MacroFileName <> "" Then
       Resume
      Else
       Err = 0
       Exit Sub
       Resume
     End If
      Case Else
    MsgBox "Error #" & Err & " occurred in InitializeMacro." & CRLF & Error
    Err = 0
    Exit Sub
    Resume
     End Select

End Sub

Sub lblScaleWords_DblClick ()
    Call cmdSetScale_Click
End Sub

Sub meaLines_Click ()
   PreviousTool = CurrentTool
   CurrentTool = TOOL_DRAWLINE
End Sub

Sub meaPoints_Click ()
   PreviousTool = CurrentTool
   CurrentTool = TOOL_DRAWPOINT
End Sub

Sub meaPolygons_Click ()
  PreviousTool = CurrentTool
  CurrentTool = TOOL_DRAWPOLYGON
  frmUserPolys.Show
End Sub

Sub meaStop_Click ()
    CurrentTool = PreviousTool
    If CurrentTool = TOOL_DRAWPOINT Or CurrentTool = TOOL_DRAWLINE Or
CurrentTool = TOOL_DRAWPOLYGON Then
      CurrentTool = TOOL_GunSight1
    End If
End Sub

Sub meCopy_Click ()
     Clipboard.Clear
    If ActiveControl Is Me!grdMapData Then Call CopyQSResultsToClipboard
    If ActiveControl Is Me!picMap Then Call meCopyMap_Click
```

```
1422|   If TypeOf ActiveControl Is TextBox Then Clipboard.SetText ActiveControl.Text
1423| End Sub
1424|
1425| Sub meCopyMap_Click ()
1426|    'GWP 950227  Do not use this one in preference to the home version
1427|    ' This menu item was added so that single stroke copying
1428|    ' of point could be done without selecting the map.
1429|    ' Purpose is to allow other applications to copy the
1430|    '  map to the clipboard by using SENDKEYS.
1431|    Clipboard.Clear
1432|    Call WriteMetaFileOut("") 'Call cmdCopyCoordinates_Click
1433|    'Put a bitmap copy of the map on the clipboard
1434|    Clipboard.SetData Me!picMap.Image, 2'For a bitmap
1435|    'Clipboard.SetData Me!picMap.Picture, 8'For a DIB
1436|    'Put together a tab-delimited string describing the current map view
1437|   Dim strMapView As String, n  As Long, M As Long
1438|   strMapView = ViewBounds.Bottom & TabChar & ViewBounds.Top & TabChar &
     ViewBounds.Left & TabChar & ViewBounds.Right & CRLF
1439|   For n = 1 To nThemesShownHere
1440|      M = Abs(ThemeOrder(n))
1441|      strMapView = strMapView & Themes(M).ThemeName & TabChar &
     Themes(M).MapSource & TabChar & Themes(M).DBSource & Themes(M).ThemeType
     & CRLF
1442|   Next n
1443|    Clipboard.SetText strMapView
1444|
1445|
1446|
1447| End Sub
1448|
1449| Sub mePaste_Click ()
1450|  ClipPaste
1451| End Sub
1452|
1453| Sub meThemeOrder_Click ()
1454|    cmdStopDrawing_Click
1455|    DoEvents
1456|    frmThemeOrder.Caption = "Theme Order"
1457|    frmThemeOrder.ZlblThemesToShow.Caption = "Themes to show"
1458|    frmThemeOrder.lstThemesAvail.Clear
1459|    For n = 1 To nThemes
1460|   frmThemeOrder.lstThemesAvail.AddItem Themes(n).ThemeName & Format(n, "
     000")
1461|    Next n
1462|    frmThemeOrder.lstThemesToShow.Clear
1463|    For n = 1 To nThemesShownHere
1464|   M = Abs(ThemeOrder(n))
1465|   frmThemeOrder.lstThemesToShow.AddItem Themes(M).ThemeName & Format(M, "
     000")
1466|    Next n
1467|    frmThemeOrder.Show 1
1468|    Dim nThemeHere As Long, nThemeThere As Long
1469|    If frmThemeOrder.cmdOK.FontItalic Then
1470|   ZUnloadThemeList
```

```
1471|    For nThemeThere = 1 To frmThemeOrder.lstThemesToShow.ListCount
1472|       nThemeHere = Val(Right$(frmThemeOrder.lstThemesToShow.List(nThemeThere -
      1), 3))
1473|       ZLoadThemeToMenu Themes(nThemeHere)
1474|    Next nThemeThere
1475|     End If
1476|     If moUnloadFormsAfterUse.Checked Then Unload frmThemeOrder
1477| End Sub
1478|
1479| Sub mfClose_Click ()
1480|  'MapDB.Close
1481|     Set MapDB = Nothing
1482|     strMapDBName = ""
1483|     nThemes = 0
1484|     ReDim Lines(0 To 0)
1485|     ReDim Points(0 To 0)
1486|     ReDim Tiles(0 To 0)
1487|     ReDim HatchLines(0 To 0)
1488|     ReDim Themes(0 To 0)
1489|     Erase Themes
1490|     hGMemLastPoint = 0 'Allow program to overwrite old points
1491|     ZUnloadThemeList
1492|
1493|     strMapDBName = ""
1494|     moAutoExpandDataBounds.Checked = True
1495|     mfClose.Enabled = False
1496| End Sub
1497|
1498| Sub mfCreateLayerLines_Click ()
1499| On Error GoTo ERRCreateLines
1500| frmLineOptions.Show 1  'Show the line dialog box in modal form.
1501| 'If frmLineOptions.ActiveControl.Caption = "&OK" Then
1502| If frmLineOptions.cmdOK.FontItalic Then
1503| ' This was changed 940505 to avoid crashes when the user closes a form rather
1504| '  than clicking on OK or Cancel.
1505|  Me.Show
1506|  Me.SetFocus
1507|     nThemes = nThemes + 1
1508|     ReDim Preserve Themes(nThemes)
1509|     Themes(nThemes).myIndex = nThemes
1510|     Themes(nThemes).ThemeType = JOINTEDLINE
1511|     Themes(nThemes).FirstLoc = UBound(Lines) + 1
1512|     Themes(nThemes).ThemeName = frmLineOptions.txtThemeName
1513| ' Add name to list of spatial queries.
1514|     frmSpatialQuery.cmbLayerList.AddItem Themes(nThemes).ThemeName
1515|     frmSpatialQuery.cmbLayerList.ItemData(frmSpatialQuery.cmbLayerList.NewIndex) =
    nThemes
1516|
1517|     Themes(nThemes).MapSource = frmLineOptions.txtMapSQL
1518|     Themes(nThemes).DBSource = frmLineOptions.txtSpatialQuerySQL
1519|     Dim Bounds As RealRect
1520|     Set snpMapLineQuery = MapDB.CreateSnapshot(Themes(nThemes).MapSource)
1521|     Call LoadLines(snpMapLineQuery, Lines(), False, Bounds)
1522|     Themes(nThemes).LastLoc = UBound(Lines)
```

```
1523|    If Themes(nThemes).LastLoc < Themes(nThemes).FirstLoc Then GoSub
      CancelThemeLines
1524|    If Bounds.Left > 1E+33 Or Bounds.Bottom > 1E+33 Or Bounds.Top < -1E+33 Or
      Bounds.Right < -1E+33 Then GoSub CancelThemeLines
1525|    If nThemes = 1 Then LSet DataBounds = Bounds
1526|    If moAutoExpandDataBounds.Checked Then Call AdjustMapScaleView(Bounds)
1527|    snpMapLineQuery.Close
1528|    Set snpMapLineQuery = Nothing
1529|
1530|    Call ZLoadThemeToMenu(Themes(nThemes))
1531|
1532|    Call ShowThemes
1533|   End If
1534|   If moUnloadFormsAfterUse.Checked Then Unload frmLineOptions
1535|   Exit Sub
1536| 'xxx add something to leap out of here on error, especially 3061--managed to sneak past
      error checking
1537| ERRCreateLines:
1538| Select Case Err
1539|    Case 3061
1540|   'Wrong number of parameters supplied, meaning a field is listed
1541|   ' in the select clause that is not in the table or query
1542|   GoSub CancelThemeLines
1543|   Err = 0
1544|   Exit Sub
1545|   Resume
1546|    Case Else
1547|      MsgBox "Error #" & Err & " occurred trying to create a line layer." & CRLF & Error
1548|      Err = 0
1549|      GoSub CancelThemeLines
1550|      Exit Sub
1551|      Resume
1552| End Select
1553|
1554|
1555|
1556| CancelThemeLines:
1557|  Beep
1558|  MsgBox "That resulted in no data being retrieved", 0, "Line Layer Creation Error"
1559|   flgStopDrawing = False
1560|   flgSkipRepainting = False
1561|   nThemes = nThemes - 1
1562|   If nThemes = 0 Then
1563|    Erase Themes
1564|    Else          'ReDim Preserve Themes(0 To nThemes)
1565|     ReDim Preserve Themes(1 To nThemes)
1566|   End If
1567| Exit Sub
1568| Return
1569|
1570| End Sub
1571|
1572| Sub mfCreateLayerPolygons_Click ()
1573| On Error GoTo ERRCreatePolygons
```

```
frmPolyOptions.Show 1  'Show the polygon dialog box in modal form.
Me.Show
Me.SetFocus

If frmPolyOptions.cmdOK.FontItalic Then
  'If frmPolyOptions.chkOutlinePolygon Or frmPolyOptions.chkFillPolygon Then

    nThemes = nThemes + 1
    ReDim Preserve Themes(nThemes)
' Begin by loading the polygons
    Themes(nThemes).myIndex = nThemes
    Themes(nThemes).ThemeType = SOLIDPOLYGON
    Themes(nThemes).FirstLoc = UBound(Polys) + 1
    Themes(nThemes).ThemeName = frmPolyOptions.txtThemeName
' Add name to list of spatial queries.
    Themes(nThemes).MapSource = frmPolyOptions.txtMapSQL
    Themes(nThemes).DBSource = frmPolyOptions.txtSpatialQuerySQL
    Dim Bounds As RealRect
    Set snpMapPolyQuery = MapDB.CreateSnapshot(Themes(nThemes).MapSource)
    Call LoadPolygons(snpMapPolyQuery, Polys(), False, Bounds)

    Themes(nThemes).LastLoc = UBound(Polys)
    If Themes(nThemes).LastLoc < Themes(nThemes).FirstLoc Then GoSub
  CancelThemePolygons
    If Bounds.Left > 1E+33 Or Bounds.Bottom > 1E+33 Or Bounds.Top < -1E+33 Or
  Bounds.Right < -1E+33 Then GoSub CancelThemePolygons
    If nThemes = 1 Then LSet DataBounds = Bounds
    If moAutoExpandDataBounds.Checked Then Call AdjustMapScaleView(Bounds)
    Call ZLoadThemeToMenu(Themes(nThemes))
    frmSpatialQuery.cmbLayerList.AddItem Themes(nThemes).ThemeName
    frmSpatialQuery.cmbLayerList.ItemData(frmSpatialQuery.cmbLayerList.NewIndex) =
  nThemes

    snpMapPolyQuery.Close
    Set snpMapPolyQuery = Nothing
    Call ShowThemes
  End If
  If moUnloadFormsAfterUse.Checked Then Unload frmPolyOptions
  Exit Sub
'xxx add something to leap out of here on error, especially 3061--managed to sneak past
  error checking
ERRCreatePolygons:
Select Case Err
    Case 3061
   'Wrong number of parameters supplied, meaning a field is listed
   ' in the select clause that is not in the table or query
   GoSub CancelThemePolygons
   Err = 0
   Exit Sub
   Resume
    Case Else
      MsgBox "Error #" & Err & " occurred trying to create a polygon layer." & CRLF &
  Error
      Err = 0
```

```
1623|       GoSub CancelThemePolygons
1624|       Exit Sub
1625|       Resume Next
1626| End Select
1627|
1628|
1629|
1630| CancelThemePolygons:
1631|  Beep
1632|  MsgBox "That resulted in no data being retrieved", 0, "Polygon Layer Creation Error"
1633|    flgStopDrawing = False
1634|    flgSkipRepainting = False
1635|    nThemes = nThemes - 1
1636|    If nThemes = 0 Then
1637|      Erase Themes
1638|     Else            'ReDim Preserve Themes(0 To nThemes)
1639|      ReDim Preserve Themes(1 To nThemes)
1640|    End If
1641| Exit Sub
1642| Return
1643|
1644| End Sub
1645|
1646| Sub mfCreateLayerRasterGrid_Click ()
1647| On Error GoTo ERRCreateRaster
1648| ReDim TypesAllowed(1 To 1)  As Integer
1649| Dim GridName As String
1650| TypesAllowed(1) = DB_LONG
1651|
1652| GridName = UserChoiceOfRasterGrids(TypesAllowed(), MapDB, "Choose a RasterGrid
      to display in color")
1653|
1654| If GridName = "" Then Exit Sub
1655| 'frmRasterOptions.Show 1  'Show the Point dialog box in modal form.
1656|   Dim myID As Long
1657|
1658| 'If frmRasterOptions.cmdOK.FontItalic Then
1659|
1660|    nThemes = nThemes + 1
1661|    ReDim Preserve Themes(nThemes)
1662|    Themes(nThemes).myIndex = nThemes
1663|    Themes(nThemes).ThemeType = RASTERGRID
1664|    myID = UBound(RasterGrids) + 1
1665|    Themes(nThemes).FirstLoc = myID
1666|    Themes(nThemes).ThemeName = GridName'frmRasterOptions.lstGrids
1667|
1668|    Themes(nThemes).MapSource = "SELECT * FROM zWHEATtblRasterGrids WHERE
      GridName ='" & Themes(nThemes).ThemeName & "'"
1669|    Themes(nThemes).DBSource = Themes(nThemes).MapSource
1670|    ReDim Preserve RasterGrids(0 To myID)
1671|    Call LoadRasterGridIntoGlobalMem(Themes(nThemes).ThemeName,
      RasterGrids(myID), MapDB)
1672|
1673|    Dim Bounds As RealRect
```

```
1674|     Bounds.Left = RasterGrids(myID).X_Origin - .5 * RasterGrids(myID).DeltaX
1675|     Bounds.Bottom = RasterGrids(myID).Y_Origin - .5 * RasterGrids(myID).DeltaY
1676|     Bounds.Right = Bounds.Left + (RasterGrids(myID).nColsX + 1) *
       RasterGrids(myID).DeltaX
1677|     Bounds.Top = Bounds.Bottom + (RasterGrids(myID).nRowsY + 1) *
       RasterGrids(myID).DeltaY
1678|
1679|     Themes(nThemes).LastLoc = Themes(nThemes).FirstLoc
1680|     If nThemes = 1 Then LSet DataBounds = Bounds
1681|     If moAutoExpandDataBounds.Checked Then Call AdjustMapScaleView(Bounds)
1682|
1683|     Call ZLoadThemeToMenu(Themes(nThemes))
1684|
1685|     Call ShowThemes
1686|   'End If
1687|   'If moUnloadFormsAfterUse.Checked Then Unload frmRasterOptions
1688|   Exit Sub
1689|  ERRCreateRaster:
1690|  Select Case Err
1691|      Case 3011, 3078
1692|    ZDefineGridTable MapDB
1693|    MsgBox "There are no pre-existing RasterGrids.  Choose Cancel from next window."
1694|    Resume
1695|     Case 3061
1696|    'Wrong number of parameters supplied, meaning a field is listed
1697|    ' in the select clause that is not in the table or query
1698|    GoSub CancelThemeRaster
1699|    Err = 0
1700|    Exit Sub
1701|    Resume
1702|     Case Else
1703|       MsgBox "Error #" & Err & " occurred trying to create a symbol layer." & CRLF & Error
1704|       Err = 0
1705|       GoSub CancelThemeRaster
1706|       Exit Sub
1707|       Resume
1708|  End Select
1709|  CancelThemeRaster:
1710|       Beep
1711|       MsgBox "That resulted in no data being retrieved", 0, "Symbol Layer Creation Error"
1712|    flgStopDrawing = False
1713|    flgSkipRepainting = False
1714|       nThemes = nThemes - 1
1715|       If nThemes = 0 Then
1716|    Erase Themes
1717|    Else            'ReDim Preserve Themes(0 To nThemes)
1718|    ReDim Preserve Themes(1 To nThemes)
1719|       End If
1720|       Exit Sub
1721|       Return
1722|
1723|  End Sub
1724|
1725|  Sub mfCreateLayerSymbols_Click ()
```

```
1726| On Error GoTo ERRCreateSymbol
1727| frmPointOptionsSimple.Show 1  'Show the Point dialog box in modal form.
1728|
1729|
1730| If frmPointOptionsSimple.cmdOK.FontItalic Then
1731|    nThemes = nThemes + 1
1732|    ReDim Preserve Themes(nThemes)
1733|    Themes(nThemes).myIndex = nThemes
1734|    Themes(nThemes).ThemeType = ONEPOINT
1735|    Themes(nThemes).FirstLoc = UBound(Points) + 1
1736|    Themes(nThemes).ThemeName = frmPointOptionsSimple.txtThemeName
1737| ' Add name to list of spatial queries.
1738|    frmSpatialQuery.cmbLayerList.AddItem Themes(nThemes).ThemeName
1739|    frmSpatialQuery.cmbLayerList.ItemData(frmSpatialQuery.cmbLayerList.NewIndex) =
     nThemes
1740|
1741|    Themes(nThemes).MapSource = frmPointOptionsSimple.txtMapSQL
1742|    Themes(nThemes).DBSource = frmPointOptionsSimple.txtSpatialQuerySQL
1743|    Dim Bounds As RealRect
1744|    Set snpMapPointQuery = MapDB.CreateSnapshot(Themes(nThemes).MapSource)
1745|    Call LoadPoints(snpMapPointQuery, Points(), False, Bounds)
1746|    Themes(nThemes).LastLoc = UBound(Points)
1747|    If Themes(nThemes).LastLoc < Themes(nThemes).FirstLoc Then GoSub
     CancelThemeSymbols
1748|    If Bounds.Left > 1E+33 Or Bounds.Bottom > 1E+33 Or Bounds.Top < -1E+33 Or
     Bounds.Right < -1E+33 Then GoSub CancelThemeSymbols
1749|    If nThemes = 1 Then LSet DataBounds = Bounds
1750|    If moAutoExpandDataBounds.Checked Then Call AdjustMapScaleView(Bounds)
1751|    snpMapPointQuery.Close
1752|    Set snpMapPointQuery = Nothing
1753|
1754|    Call ZLoadThemeToMenu(Themes(nThemes))
1755|
1756|    Call ShowThemes
1757|  End If
1758|  If moUnloadFormsAfterUse.Checked Then Unload frmPointOptionsSimple
1759|  Exit Sub
1760| ERRCreateSymbol:
1761| Select Case Err
1762|    Case 3061
1763|   'Wrong number of parameters supplied, meaning a field is listed
1764|   ' in the select clause that is not in the table or query
1765|   GoSub CancelThemeSymbols
1766|   Err = 0
1767|   Exit Sub
1768|   Resume
1769|    Case Else
1770|      MsgBox "Error #" & Err & " occurred trying to create a symbol layer." & CRLF & Error
1771|      Err = 0
1772|      GoSub CancelThemeSymbols
1773|      Exit Sub
1774|      Resume
1775| End Select
1776| CancelThemeSymbols:
```

```
1777|       Beep
1778|       MsgBox "That resulted in no data being retrieved", 0, "Symbol Layer Creation Error"
1779|    flgStopDrawing = False
1780|    flgSkipRepainting = False
1781|       nThemes = nThemes - 1
1782|       If nThemes = 0 Then
1783|     Erase Themes
1784|    Else            'ReDim Preserve Themes(0 To nThemes)
1785|     ReDim Preserve Themes(1 To nThemes)
1786|       End If
1787|       Exit Sub
1788|       Return
1789| End Sub
1790|
1791| Sub mfCreateLayerTextLabels_Click ()
1792| On Error GoTo ERRCreateLabels
1793|
1794|  frmLabelOptions.Show 1
1795| If frmLabelOptions.cmdOK.FontItalic Then
1796|    nThemes = nThemes + 1
1797|    ReDim Preserve Themes(nThemes)
1798|    Themes(nThemes).myIndex = nThemes
1799|    Themes(nThemes).ThemeType = LABELTEXT
1800|    Themes(nThemes).FirstLoc = UBound(Labels) + 1
1801|    Themes(nThemes).ThemeName = frmLabelOptions.txtThemeName
1802| ' Add name to list of spatial queries.
1803|    "frmSpatialQuery.cmbLayerList.AddItem Themes(nThemes).ThemeName
1804|    "frmSpatialQuery.cmbLayerList.ItemData(frmSpatialQuery.cmbLayerList.NewIndex) =
      nThemes
1805|
1806|    Themes(nThemes).MapSource = frmLabelOptions.txtMapSQL
1807|    Themes(nThemes).DBSource = frmLabelOptions.txtSpatialQuerySQL
1808|    Dim Bounds As RealRect
1809|    Set snpMapLabelQuery = MapDB.CreateSnapshot(Themes(nThemes).MapSource)
1810|    Call LoadLabels(snpMapLabelQuery, Labels(), False, Bounds)
1811|    Themes(nThemes).LastLoc = UBound(Labels)
1812|    If Themes(nThemes).LastLoc < Themes(nThemes).FirstLoc Then GoSub
      CancelThemeLabels
1813|    If Bounds.Left > 1E+33 Or Bounds.Bottom > 1E+33 Or Bounds.Top < -1E+33 Or
      Bounds.Right < -1E+33 Then GoSub CancelThemeLabels
1814|    If nThemes = 1 Then LSet DataBounds = Bounds
1815|    If moAutoExpandDataBounds.Checked Then Call AdjustMapScaleView(Bounds)
1816|
1817|    snpMapLabelQuery.Close
1818|    Set snpMapLabelQuery = Nothing
1819|
1820|    Call ZLoadThemeToMenu(Themes(nThemes))
1821|
1822|    Call ShowThemes
1823|   End If
1824|   If moUnloadFormsAfterUse.Checked Then Unload frmLabelOptions
1825| Exit Sub
1826| ERRCreateLabels:
1827| Select Case Err
```

```
1828|     Case 3061
1829|    'Wrong number of parameters supplied, meaning a field is listed
1830|    ' in the select clause that is not in the table or query
1831|    GoSub CancelThemeLabels
1832|    Err = 0
1833|    Exit Sub
1834|    Resume
1835|     Case Else
1836|       MsgBox "Error #" & Err & " occurred trying to create a text label layer." & CRLF &
      Error
1837|       Err = 0
1838|       GoSub CancelThemeLabels
1839|       Exit Sub
1840|       Resume
1841| End Select
1842|
1843|
1844|
1845| CancelThemeLabels:
1846|  Beep
1847|  MsgBox "That resulted in no data being retrieved", 0, "Text Label Layer Creation Error"
1848|    flgStopDrawing = False
1849|    flgSkipRepainting = False
1850|    nThemes = nThemes - 1
1851|    If nThemes = 0 Then
1852|      Erase Themes
1853|     Else            'ReDim Preserve Themes(0 To nThemes)
1854|      ReDim Preserve Themes(1 To nThemes)
1855|    End If
1856| Exit Sub
1857| Return
1858|
1859| End Sub
1860|
1861| Sub mfCreateLayerTiles_Click ()
1862| On Error GoTo ERRmfCreateLayerTiles
1863|
1864| frmTileOptions.Show 1  'Show the tiles dialog box in modal form.
1865|
1866| If Not frmTileOptions.cmdOK.FontItalic Then Exit Sub
1867| ' The form sets the cmdOK font to Italic if selected, and
1868| ' unsets it during repaints, so it is only italic if the
1869| ' user chose OK.  This avoids an ugly memory leak bug in Visual Basic.
1870| ' It the user chose cancel, skip out of the subroutine.
1871|
1872|  flgStopDrawing = True
1873|  flgSkipRepainting = True
1874|  Me.Show
1875|  Me.SetFocus
1876|    nThemes = nThemes + 1
1877|    ReDim Preserve Themes(nThemes)
1878|    Themes(nThemes).myIndex = nThemes
1879|    Themes(nThemes).ThemeType = TILEBLOCK
1880|    Themes(nThemes).FirstLoc = UBound(Tiles) + 1
```

```
1881|     Themes(nThemes).ThemeName = frmTileOptions.txtThemeName
1882|  ' Add name to list of spatial queries.
1883|     frmSpatialQuery.cmbLayerList.AddItem Themes(nThemes).ThemeName
1884|     frmSpatialQuery.cmbLayerList.ItemData(frmSpatialQuery.cmbLayerList.NewIndex) =
      nThemes
1885|
1886|     Themes(nThemes).MapSource = frmTileOptions.txtMapSQL
1887|     Themes(nThemes).DBSource = frmTileOptions.txtSpatialQuerySQL
1888|     Dim Bounds As RealRect
1889|     Set snpMapTileQuery = MapDB.CreateSnapshot(Themes(nThemes).MapSource)
1890|     Call LoadTiles(snpMapTileQuery, Tiles(), False, Bounds) 'xxx
1891|     Themes(nThemes).LastLoc = UBound(Tiles)
1892|  'GWP940510 did not work. Need to handle zero-number of records layeres.
1893|     If Themes(nThemes).LastLoc < Themes(nThemes).FirstLoc Then GoSub
      CancelThemeTiles
1894|     If Bounds.Left > 1E+33 Or Bounds.Bottom > 1E+33 Or Bounds.Top < -1E+33 Or
      Bounds.Right < -1E+33 Then GoSub CancelThemeTiles
1895|     If nThemes = 1 Then LSet DataBounds = Bounds 'Eliminate earlier databounds and
      viewbounds for first theme.
1896|     If moAutoExpandDataBounds.Checked Then Call AdjustMapScaleView(Bounds)
1897|     snpMapTileQuery.Close
1898|     Set snpMapTileQuery = Nothing
1899|
1900|    " moLayerName(0).Enabled = False
1901|  'Append new colors to color palette.  May need to make this more clever later on, to
1902|  '  avoid exploding when nColors>256.
1903|     'Dim nNewColors As Long
1904|     'nNewColors = Val(frmTileOptions.txtNClass)
1905|     'Call UpdateColors(nNewColors)
1906|     Call ZLoadThemeToMenu(Themes(nThemes))
1907|
1908|     flgSkipRepainting = False
1909|     flgStopDrawing = False
1910|     Call ShowThemes
1911|  If moUnloadFormsAfterUse.Checked Then Unload frmTileOptions
1912|  Exit Sub
1913|  ERRmfCreateLayerTiles:
1914|  Select Case Err
1915|     Case 3061
1916|    'Wrong number of parameters supplied, meaning a field is listed
1917|    ' in the select clause that is not in the table or query
1918|    GoSub CancelThemeTiles
1919|    Err = 0
1920|    Exit Sub
1921|    Resume
1922|     Case Else
1923|       MsgBox "Error #" & Err & " occurred trying to create a tiles layer." & CRLF & Error
1924|       Err = 0
1925|       GoSub CancelThemeTiles
1926|       Exit Sub
1927|       Resume
1928|  End Select
1929|
1930|
```

```
1931|
1932| CancelThemeTiles:
1933|   Beep
1934|   MsgBox "That resulted in no data being retrieved", 0, "Tiles Layer Creation Error"
1935|     flgStopDrawing = False
1936|     flgSkipRepainting = False
1937|     nThemes = nThemes - 1
1938|     If nThemes = 0 Then
1939|       Erase Themes
1940|     Else            'ReDim Preserve Themes(0 To nThemes)
1941|       ReDim Preserve Themes(1 To nThemes)
1942|     End If
1943| Exit Sub
1944| Return
1945|
1946| End Sub
1947|
1948| Sub mfDisplayColorLegend_Click ()
1949|     'xxxx
1950|     Dim strCLUTName As String
1951|     strCLUTName = GetLUTNameFromUser(strMapDBName)
1952|     If strCLUTName = "" Then Exit Sub
1953|      Call ShowColorLegend(strCLUTName)
1954|
1955|
1956|
1957|
1958| End Sub
1959|
1960| Sub mfExit_Click ()
1961| MouseX = SetClassWord(Me.picMap.hWnd, GCW_HCURSOR, OriginalCursorStyle)
1962|
1963| ShutDownAndFreeMem
1964| End Sub
1965|
1966| Sub mfExportMapAs_Click ()
1967|     Dim Outfile As String
1968|
1969|     Me.CMDialog1.DialogTitle = "Windows Metafile to Export Current Map into"
1970|     Me.CMDialog1.Filter = "Windows Metafile (*.wmf)|*.wmf"
1971|     Me.CMDialog1.Filename = Outfile
1972|     Me.CMDialog1.Flags = OFN_OVERWRITEPROMPT Or OFN_PATHMUSTEXIST
1973|
1974|     Me.CMDialog1.Action = 1
1975|
1976|     Outfile = UCase$(Me.CMDialog1.Filename)
1977|     If Outfile = "" Then Exit Sub
1978|     If InStr(Outfile, ".WMF") = 0 Then Outfile = Outfile & ".WMF"
1979|     Call WriteMetaFileOut(Outfile)
1980|
1981| End Sub
1982|
1983| Sub mfOpen_Click ()
1984| 'open an old file of the form filename.mdb for a MicroSoft DataBase
```

```
1985| On Error GoTo ERRmfOpen
1986| OpenDataBaseForEMap
1987| If strMapDBName = "" Then Exit Sub
1988| 'Enable dimmed menu options.
1989| mfClose.Enabled = True
1990| zzView.Enabled = True
1991| zzzzOptions.Enabled = True
1992| zzzEdit.Enabled = True
1993| mfOpenMapLayer.Enabled = True
1994| mfSaveMapLayerAs.Enabled = True
1995| mfPrintMap.Enabled = True
1996| mfExportMapAs.Enabled = True
1997| mfNewMapLayer.Enabled = True
1998| mfDisplayColorLegend.Enabled = True
1999| mfSpatialAnalysis.Enabled = True
2000| SendKeys "%F"
2001| SendKeys "N"
2002|
2003|
2004| Exit Sub
2005| ERRmfOpen:
2006| Select Case Err
2007| Case 3051 'Could not open file becasue someone else has exclusive lock on it.
2008|   MsgBox "Someone else has opened that database in exclusive mode." & CRLF & "You
      can either get the other user or application to close it and re-open it in non-exclusive
      mode, or open in it read-only mode."' & CRLF & "Do you wish to open it in read only
      mode?", 35, "Error opening database")
2009|    strMapDBName = ""
2010|
2011|    Err = 0
2012|    Exit Sub
2013|    Resume Next
2014|
2015|  Case Else
2016|   MsgBox "Error #" & Err & " occurred while opening the database." & CRLF & Error
2017|   Err = 0
2018|   Exit Sub
2019|  Resume Next
2020| End Select
2021| End Sub
2022|
2023| Sub mfOpenMapLayer_Click ()
2024|  On Error GoTo ERRmfOpenMapLayer_Click
2025|   cmdStopDrawing_Click
2026|   DoEvents
2027|   frmThemeOrder.Caption = "Themes to load"
2028|   frmThemeOrder.ZlblThemesToShow.Caption = "Theme definitions to LOAD from disk
      into map"
2029|   frmThemeOrder.lstThemesAvail.Clear
2030|   frmThemeOrder.lstThemesToShow.Clear
2031| 'add the currently saved themes to a list to be shown to the user
2032|   Dim tblThemeDefs As table
2033|   Set tblThemeDefs = MapDB.OpenTable("zWHEATtblThemes")
2034|   tblThemeDefs.MoveFirst
```

```
2035|    Do Until tblThemeDefs.EOF
2036|     frmThemeOrder.lstThemesAvail.AddItem tblThemeDefs("ThemeName") '&
      tblThemeDefs("WhoModified") & tblThemeDefs("DateModified")
2037|     tblThemeDefs.MoveNext
2038|    Loop
2039|    tblThemeDefs.Close
2040|    Set tblThemeDefs = Nothing
2041|
2042|    frmThemeOrder.Show 1
2043|
2044|    If frmThemeOrder.cmdOK.FontItalic Then 'User chose OK
2045|      Dim nThemeThere  As Long
2046|   For nThemeThere = 1 To frmThemeOrder.lstThemesToShow.ListCount
2047|       LoadThemeDefinitionFromTable MapDB,
      CStr(frmThemeOrder.lstThemesToShow.List(nThemeThere - 1))
2048|       ZLoadThemeToMenu Themes(nThemes)
2049|   Next nThemeThere
2050|
2051|       If moAutoExpandDataBounds.Checked Then Call
      AdjustMapScaleView(DataBounds)
2052|    End If
2053| If moUnloadFormsAfterUse.Checked Then Unload frmThemeOrder
2054| Exit Sub
2055|
2056|
2057| ERRmfOpenMapLayer_Click:
2058|  Select Case Err
2059|  Case 3011  'Table not found
2060|   Call ZDefineThemesTable(MapDB)
2061|   Resume
2062|  Case Else
2063|    MsgBox "Error #" & Err & "occurred trying to read the zWHEATtblThemes." & CRLF &
      Error
2064|    Err = 0
2065|    Exit Sub
2066|    Resume
2067|  End Select
2068| End Sub
2069|
2070| Sub mfPrintMap_Click ()
2071| Call cmdStopDrawing_Click
2072|
2073| frmPrinterOptions.txtPrintScale = MapScaleInWords(Me!picMap.ScaleWidth,
      Me!picMap.Width)
2074|
2075| frmPrinterOptions.Show 1
2076|
2077| If frmPrinterOptions.cmdOK.FontItalic Then
2078|
2079|   PrinterRectangle.Right = frmPrinterOptions.txtRightMargin
2080|   PrinterRectangle.Top = frmPrinterOptions.txtTopMargin
2081|   PrinterRectangle.Left = frmPrinterOptions.txtLeftMargin
2082|   PrinterRectangle.Bottom = frmPrinterOptions.txtBottomMargin
2083|
```

```
2084|    Dim rctWorldAreaToPrint As RealRect
2085|
2086|
2087|  If frmPrinterOptions.optScreenArea Then
2088|    'Set the scale of the printer up so that the
2089|    ' corners of the ViewRect map into the corners of
2090|    '  the PrintRect.
2091|    rctWorldAreaToPrint = ViewBounds
2092|   Else 'Constrained by map scale
2093|    'Center the center point of screen on center point of map area
2094|     Dim sngNewScale As Single, XCenter As Single, YCenter As Single, dX As Single, dY
      As Single
2095|     Dim tmpString As String
2096|     tmpString = frmPrinterOptions.txtPrintScale
2097|     tmpString = ReplaceString(tmpString, ",", "")
2098|     sngNewScale = Val(tmpString)
2099|     Dim sngMultiplier As Single
2100|     'Do not user ScaleFudgeFactor, as that only applies to screen.
2101|     sngMultiplier = .001 / (MapUnitsToMeters_Multiplier()) * sngNewScale
2102|     XCenter = .5 * (ViewBounds.Right + ViewBounds.Left)
2103|     YCenter = .5 * (ViewBounds.Top + ViewBounds.Bottom)
2104|     dX = .5 * (PrinterRectangle.Right - PrinterRectangle.Left) * sngMultiplier
2105|     dY = -.5 * (PrinterRectangle.Top - PrinterRectangle.Bottom) * sngMultiplier
2106|
2107|     rctWorldAreaToPrint.Left = XCenter - dX
2108|     rctWorldAreaToPrint.Right = XCenter + dX
2109|     rctWorldAreaToPrint.Top = YCenter + dY
2110|     rctWorldAreaToPrint.Bottom = YCenter - dY
2111|  End If
2112|
2113|
2114|
2115|
2116|
2117|  Dim strTitle As String, strCaption As String
2118|  Dim posTitle As RealPoint, posCaption As RealPoint
2119|     ' If frmPrinterOptions.txtXTitle <> "" And frmPrinterOptions.txtYTitle <> "" Then
2120|     '    posTitle.X = frmPrinterOptions.txtXTitle
2121|     '    posTitle.Y = frmPrinterOptions.txtYTitle
2122|     '    strTitle = frmPrinterOptions.txtMapTitle
2123|     ' End If
2124|
2125|
2126|     Call cmdStopDrawing_Click
2127|     'DoEvents
2128|     'flgSkipRepainting = False:   flgStopDrawing = False
2129|     Call SetPrintWidthMultiplier(Val(frmPrinterOptions.txtWidthMultiplier))
2130|
2131|     Call PrintMap(rctWorldAreaToPrint, PrinterRectangle)
2132|    Dim rctBlurb As RealRect
2133|    For n = 0 To 2
2134|     Printer.ForeColor = RGB(0, 0, 0)
2135|     Printer.ScaleMode = 6'Put the printer back into millimeters
2136|     If frmPrinterOptions.txtBlurbX(n) <> "" And frmPrinterOptions.txtBlurbY(n) <> "" Then
```

```
2137|
2138|      rctBlurb.Left = frmPrinterOptions.txtBlurbX(n)
2139|      rctBlurb.Top = frmPrinterOptions.txtBlurbY(n)
2140|      rctBlurb.Right = Printer.ScaleWidth - 25
2141|      rctBlurb.Bottom = Printer.ScaleHeight
2142|      Printer.FontName = frmPrinterOptions.lblSpecimen(n).FontName
2143|      Printer.FontBold = frmPrinterOptions.lblSpecimen(n).FontBold
2144|      Printer.FontItalic = frmPrinterOptions.lblSpecimen(n).FontItalic
2145|      Printer.FontSize = frmPrinterOptions.lblSpecimen(n).FontSize
2146|       Call Printer_PrintInBox(CStr(frmPrinterOptions.txtMapBlurb(n).Text), rctBlurb)
2147|      Printer.Print
2148|    End If
2149|  Next n
2150|
2151|  Printer.FontName = "Arial"
2152|  Printer.FontSize = 10
2153|  Printer.FontBold = False
2154|  Printer.FontItalic = False
2155|  If frmPrinterOptions.chkTimeDate Then
2156|    Printer.CurrentX = 12
2157|    Printer.CurrentY = 12
2158|    Printer.Print Now
2159|  End If
2160|  'Unconditionally print the coordinate blurb
2161|    Printer.CurrentY = 12: Printer.CurrentX = Printer.ScaleWidth -
      Printer.TextWidth(CoordinateBlurb()) - 12
2162|    Printer.Print CoordinateBlurb()
2163|
2164|
2165|  Printer.EndDoc
2166|  Screen.MousePointer = 0
2167|  Beep
2168|  Me.lblPercentDone = "DONE PRINTING"
2169|
2170|  End If
2171|  If moUnloadFormsAfterUse.Checked Then
2172|    Unload frmPrinterOptions
2173|    Set frmPrinterOptions = Nothing
2174|  End If
2175|
2176|  End Sub
2177|
2178|  Sub mfSaveMapLayerAs_Click ()
2179|     cmdStopDrawing_Click
2180|     DoEvents
2181|     frmThemeOrder.Caption = "Themes to save"
2182|     frmThemeOrder.ZlblThemesToShow.Caption = "Theme definitions to SAVE to table"
2183|     frmThemeOrder.lstThemesToShow.Clear
2184|     frmThemeOrder.lstThemesAvail.Clear
2185|     For n = 1 To nThemes
2186|    frmThemeOrder.lstThemesAvail.AddItem Themes(n).ThemeName & Format(n, "
      000")
2187|     Next n
2188|
```

```
2189|     frmThemeOrder.Show 1
2190|
2191|     If frmThemeOrder.cmdOK.FontItalic Then 'User chose OK
2192|       Dim nThemeHere As Long, nThemeThere As Long
2193|     For nThemeThere = 1 To frmThemeOrder.lstThemesToShow.ListCount
2194|       nThemeHere = Val(Right$(frmThemeOrder.lstThemesToShow.List(nThemeThere -
      1), 3))
2195|       SaveThemeToTable MapDB, Themes(nThemeHere)
2196|     Next nThemeThere
2197|     End If
2198| If moUnloadFormsAfterUse.Checked Then Unload frmThemeOrder
2199| End Sub
2200|
2201| Sub mfSpatialAnalysis_Click ()
2202|   frmSpatialAnalysis.lblTrigger = "CLEAR"
2203|   frmSpatialAnalysis.lblTrigger = Now
2204|   frmSpatialAnalysis.Show
2205|   If moUnloadFormsAfterUse.Checked Then Unload frmSpatialAnalysis
2206| End Sub
2207|
2208| Sub mnuHelp_Click (Index As Integer)
2209|  Dim iRetVal As Integer
2210|     Select Case Index
2211|   Case 0 'Contents
2212|     iRetVal = WinHelp(Me.hWnd, App.HelpFile, gkiHELP_INDEX, 0&)
2213|   Case 1 'Search
2214|     iRetVal = WinHelp(Me.hWnd, App.HelpFile, gkiHELP_PARTIALKEY, "")
2215|   Case 3 'How to use help
2216|     iRetVal = WinHelp(Me.hWnd, App.HelpFile, gkiHELP_HELPONHELP, 0&)
2217|   Case 5
2218|     Unload frmAboutBox
2219|     frmAboutBox.Timer1.Interval = 0 'Disable the timer
2220|     frmAboutBox.Show 1
2221|     Unload frmAboutBox
2222|     Set frmAboutBox = Nothing
2223|    End Select
2224| End Sub
2225|
2226| Sub mnuVDown_Click ()
2227| cmdStopDrawing_Click
2228| flgStopDrawing = False
2229| Call ZZShiftY(-.25 * CurrentSize.Y)
2230| If moAutoDrawOnViewShift.Checked Then Call ShowThemes
2231| End Sub
2232|
2233| Sub mnuVLeft_Click ()
2234|  cmdStopDrawing_Click
2235|  flgStopDrawing = False
2236|  ZZShiftX (-.25 * CurrentSize.X)
2237| If moAutoDrawOnViewShift.Checked Then Call ShowThemes
2238| End Sub
2239|
2240| Sub mnuVRight_Click ()
2241|  cmdStopDrawing_Click
```

```
2242|  flgStopDrawing = False
2243|  ZZShiftX (.25 * CurrentSize.X)
2244|  If moAutoDrawOnViewShift.Checked Then Call ShowThemes
2245|  End Sub
2246|
2247|  Sub mnuVUp_Click ()
2248|  cmdStopDrawing_Click
2249|  flgStopDrawing = False
2250|  Call ZZShiftY(0 + .25 * CurrentSize.Y)
2251|  If moAutoDrawOnViewShift.Checked Then Call ShowThemes
2252|  End Sub
2253|
2254|  Sub mnuVZoomin_Click ()
2255|  cmdStopDrawing_Click
2256|  flgStopDrawing = False
2257|  Call ZZoom(ViewBounds.Left + CurrentSize.X * .146446609, ViewBounds.Top -
       CurrentSize.Y * .146446609, .70710678)
2258|  If moAutoDrawOnViewShift.Checked Then Call ShowThemes
2259|  End Sub
2260|
2261|  Sub mnuVZoomout_Click ()
2262|  cmdStopDrawing_Click
2263|  flgStopDrawing = False
2264|  Call ZZoom(ViewBounds.Left - CurrentSize.X * .207106781, ViewBounds.Top +
       CurrentSize.Y * .207106781, 1.414213562)
2265|  If moAutoDrawOnViewShift.Checked Then Call ShowThemes
2266|  End Sub
2267|
2268|  Sub moAutoDrawOnViewShift_Click ()
2269|  moAutoDrawOnViewShift.Checked = Not moAutoDrawOnViewShift.Checked
2270|  If moAutoDrawOnViewShift.Checked Then Call ShowThemes
2271|  End Sub
2272|
2273|  Sub moAutoExpandDatabounds_Click ()
2274|  moAutoExpandDataBounds.Checked = Not moAutoExpandDataBounds.Checked
2275|
2276|  End Sub
2277|
2278|  Sub moAutoredraw_Click ()
2279|  'Operates a toggle switch controlling whether a copy of the
2280|  '  map should be kept in memory.
2281|  moAutoredraw.Checked = Not moAutoredraw.Checked
2282|  Me.picMap.AutoRedraw = moAutoredraw.Checked
2283|  If moAutoredraw.Checked Then ShowThemes
2284|
2285|
2286|  End Sub
2287|
2288|  Sub moMapBackColor_Click ()
2289|  Me.CMDialog1.CancelError = True
2290|  On Error GoTo LeaveColorAlone2
2291|  Me.CMDialog1.DialogTitle = "Map Background Color"
2292|  Me.CMDialog1.Flags = &H1&
2293|  Me.CMDialog1.Action = 3
```

```
2294|  MapBaseColor = Me.CMDialog1.Color
2295|  'Colors(0) = MapBaseColor
2296|
2297|  LeaveColorAlone2:
2298|   'User chose cancel, do nothing
2299|   Exit Sub
2300|
2301|  End Sub
2302|
2303|  Sub mombcOther_Click ()
2304|  'Open a common dialog box to allow user to select
2305|  ' a custom color for the background.
2306|  'This will be color(0).
2307|
2308|  Me.CMDialog1.CancelError = True
2309|  On Error GoTo LeaveColorAlone
2310|  Me.CMDialog1.DialogTitle = "Map Background Color"
2311|  Me.CMDialog1.Flags = &H1&
2312|  Me.CMDialog1.Action = 3
2313|  MapBaseColor = Me.CMDialog1.Color
2314|  'Colors(0) = MapBaseColor
2315|
2316|  LeaveColorAlone:
2317|   'User chose cancel, do nothing
2318|   Exit Sub
2319|  End Sub
2320|
2321|  Sub moRefreshMap_Click ()
2322|     cmdRefreshMap_Click
2323|  End Sub
2324|
2325|  Sub moReset_Click ()
2326|  M = SetCursorByHCURSOR(Me.picMap, OriginalCursorStyle)
2327|  'ZResetTblQryList
2328|  Call LoadMDB
2329|    flgStopDrawing = False
2330|    flgSkipRepainting = False
2331|    flgLoadDIB = False
2332|  RestoreDrawingSettings
2333|  End Sub
2334|
2335|  Sub moShowLastQuery_Click ()
2336|  moShowLastQuery.Checked = Not moShowLastQuery.Checked
2337|  End Sub
2338|
2339|  Sub moThemesVisible_Click ()
2340|   ZAskUserForThemesVisible
2341|
2342|  End Sub
2343|
2344|  Sub moUnloadFormsAfterUse_Click ()
2345|  'This is a toggle switch on the checked status of the menu item
2346|     moUnloadFormsAfterUse.Checked = Not moUnloadFormsAfterUse.Checked
2347|     If moUnloadFormsAfterUse.Checked Then
```

```
2348|    Unload frmTileOptions
2349|    Unload frmPointOptionsSimple
2350|    Unload frmLineOptions
2351|    Unload frmPolyOptions
2352|    Unload frmLabelOptions
2353|    Unload frmPrinterOptions
2354|    Unload frmViewPortOptions
2355|     End If
2356| End Sub
2357|
2358| Sub mqSpatial_Click ()
2359|
2360|    On Error GoTo ERR_mqSpatial_Click
2361|    frmSpatialQuery.txtHighlightRadius = sngQSHighlightRadius
2362|    frmSpatialQuery.Picture1.BackColor = lngQSHighlightColor
2363|    frmSpatialQuery.Show 1
2364|    Dim Temp2 As String
2365|
2366|    If frmSpatialQuery.cmdOK.FontItalic Then
2367|      'always clear the grid
2368|      Call ClearGrid(Me!grdMapData)
2369|
2370|
2371|       'Macro graph only for GMD2, remove ASAP
2372|       If frmSpatialQuery.chkRunMacro <> 0 And Not (frmSpatialQuery.optNone.Value)
      Then
2373|       MacroName = frmSpatialQuery.txtMacroName
2374|       IntializeMacro MacroName
2375|      Else
2376|       MacroName = ""
2377|       End If
2378|
2379|       lngQSHighlightColor = frmSpatialQuery.Picture1.BackColor
2380|       sngQSHighlightRadius = frmSpatialQuery.txtHighlightRadius
2381|       If sngQSHighlightRadius < 0 Then sngQSHighlightRadius = 0
2382|       ReDim QSDataOut(0 To 0)
2383|    If frmSpatialQuery.optClosestPoint Then
2384|       lngSearchType = QS_CLOSESTPOINT
2385|       Temp2 = "Closest point "
2386|       CurrentTool = TOOL_QueryByPoint
2387|       'LastCursor = SetCursorByName(Me.picMap, "QueryByPoint")
2388|      End If
2389|
2390|    If frmSpatialQuery.optNearBox Then
2391|       lngSearchType = QS_NEARCURSOR
2392|       Temp2 = "Near cursor "
2393|       CurrentTool = TOOL_BOX10
2394|       'LastCursor = SetCursorByName(Me.picMap, "QueryByPoint")
2395|      End If
2396|
2397|    If frmSpatialQuery.optRadius Then
2398|       lngSearchType = QS_RADIUSSEARCH
2399|       QSDataOut(0).Dist1 = frmSpatialQuery.txtSearchRadius
2400|       Temp2 = "All within radius of " & Format(QSDataOut(0).Dist1)
```

```
2401|       CurrentTool = TOOL_QueryWithinRadius1
2402|       'LastCursor = SetCursorByName(Me.picMap, "QueryWithinRadius1")
2403|     End If
2404|
2405|   If frmSpatialQuery.optWithinBox Then
2406|       lngSearchType = QS_WITHINBOX
2407|       Temp2 = "Within box "
2408|       CurrentTool = TOOL_QueryByBox1
2409|       'LastCursor = SetCursorByName(Me.picMap, "QueryByBox1")
2410|     End If
2411|
2412|   If frmSpatialQuery.optPolygon Then
2413|       lngSearchType = QS_WITHINPOLY
2414|       Temp2 = "Enclosing polygons "
2415|       CurrentTool = TOOL_QueryByPolygon1
2416|       'LastCursor = SetCursorByName(Me.picMap, "QueryByPolygon1")
2417|     End If
2418|
2419|   If frmSpatialQuery.optTile Then
2420|       lngSearchType = QS_WITHINTILE
2421|       Temp2 = "Enclosing tile "
2422|       CurrentTool = TOOL_QueryByTile
2423|     End If
2424|
2425|   If frmSpatialQuery.optLine Then
2426|       lngSearchType = QS_LINEDIST
2427|       Temp2 = "Find closest line "
2428|       CurrentTool = TOOL_QueryByClosestLine
2429|       'LastCursor = SetCursorByName(Me.picMap, "QueryByPolygon1")
2430|     End If
2431|
2432|   If frmSpatialQuery.optNone Then
2433|       lngSearchType = QS_NONE
2434|       Temp2 = "No spatial query "
2435|       CurrentTool = TOOL_GunSight1
2436|       Erase QSDataOut
2437|       LastQS.X = -1E+34
2438|       LastQS.Y = 1E+34
2439|       'LastCursor = SetCursorByName(Me.picMap, "Gunsight1")
2440|     End If
2441|   If frmSpatialQuery.optRaster Then
2442|    'Load raster grid
2443|     Dim strGridWanted  As String
2444|     strGridWanted = frmSpatialQuery.txtRasterGrid
2445|     If strGridWanted = "" Then Exit Sub
2446|     QSRasterGridNumber = UBound(RasterGrids) + 1 'Set a flag so Mouse_Move knows
      to run the spatial query
2447|     ReDim Preserve RasterGrids(0 To QSRasterGridNumber)
2448|     Call LoadRasterGridIntoGlobalMem(strGridWanted,
    RasterGrids(QSRasterGridNumber), MapDB)
2449|     Else
2450|     'If the user has not chosen continous update on the raster grid,
2451|     '  discard the previously defined RasterGrid for spatial querying
2452|     ReturnVal = GlobalFree(RasterGrids(QSRasterGridNumber).hGMem)
```

```
2453|      RasterGrids(QSRasterGridNumber).hGMem = 0
2454|      RasterGrids(QSRasterGridNumber).LayerName = "DISCARDED ALREADY"
2455|      QSRasterGridNumber = 0
2456|      'These should be harmless enough, even if QSRasterGridNumber=0
2457|       'Leave the cursor alone.
2458|     End If
2459|      LastCursor = SetCursorByHCURSOR(Me.picMap, ToolPick(CurrentTool).hCursor)
2460|
2461| 'Figure out how to get the index of a theme from its name.                'xxxxxx This use
      of index is iffy.
2462|   Dim strQSsql As String, Temp As String
2463|   Temp = frmSpatialQuery.cmbLayerList
2464|   For n = 1 To nThemes
2465|      If Themes(n).ThemeName = Temp Then
2466|      Call CopyTYPTheme(Themes(n), QSTheme)
2467|      'LSet QSTheme = Themes(n)
2468|      strQSsql = Themes(n).DBSource
2469|      lngSearchThemeID = n
2470|       End If
2471|   Next n
2472|   Me.pnl3dInstructions = Temp2 & " from: " & Temp
2473|   Set dynSpatialQuery = MapDB.CreateDynaset(strQSsql)
2474|   grdMapData.Cols = 1 + dynSpatialQuery.Fields.Count
2475|
2476|   flgUpdateGrdTxt = False
2477|    Call LabelGridFromDynaset(grdMapData, dynSpatialQuery)
2478|   flgUpdateGrdTxt = True
2479|    'Kluge 950609 GWPouch at KGS
2480|    ' Added code to allow E-Map to open and run a file every time
2481|    '  the user selects a feature with a spatial query.  The file
2482|    '  to open must be in a field called OnClickOpenFile, and should
2483|    '  be a filename with an extension recognized by file manager.
2484|   'prepare for opening files as a spatial query result
2485|
2486|   idxOnClickOpenFile = -1
2487|   For n = 0 To dynSpatialQuery.Fields.Count - 1
2488|      If UCase$(dynSpatialQuery.Fields(n).Name) = "ONCLICKOPENFILE" Then
2489|    idxOnClickOpenFile = n
2490|    Exit For
2491|      End If
2492|   Next n
2493|   'At this point, idxOnClickOpenFile contains -1 if none of the fields
2494|    '  were named OnClickOpenFile, or the index of the field if there was one.
2495|
2496|
2497|
2498| 'Set up display of spatial query results in a second grid, occupying its
2499| '  own form.
2500|
2501|
2502| frmGridDataEdit.Data1.DatabaseName = strMapDBName
2503| frmGridDataEdit.lblQSType = Temp2 & " from: " & Temp
2504| frmGridDataEdit.lblQSSource = "CLEAR"
2505| frmGridDataEdit.lblQSSource = strQSsql
```

```
2506|
2507|  Call InitializeGridForDynaset(frmGridDataEdit.grdAttributes, dynSpatialQuery)
2508|  Call ZAddDBLabeledColumn(frmGridDataEdit.grdAttributes, "X LastPoint 06", "")
2509|  Call ZAddDBLabeledColumn(frmGridDataEdit.grdAttributes, "Y LastPoint 06", "")
2510|  Call ZAddDBLabeledColumn(frmGridDataEdit.grdAttributes, "Date/Time 08", "")
2511|
2512|  Me.Show
2513|  End If
2514|    If moUnloadFormsAfterUse.Checked Then Unload frmSpatialQuery
2515|  Exit Sub
2516|  ERR_mqSpatial_Click:
2517|  If Err = 91 Then 'The user closed the form, but did not hit cancel, so
2518|        ' there is no active control on frmSpatQuery
2519|    Err = 0
2520|    Exit Sub
2521|    Resume
2522|  End If
2523|     MsgBox "Error number " & Err & " occurred while setting the spatial query, in
      mqSpatial." & CRLF & Error$
2524|     Err = 0
2525|     Exit Sub
2526|     Resume Next
2527|  End Sub
2528|
2529|  Sub mvGraticule_Click ()
2530|   Me!mvGraticule.Checked = Not Me!mvGraticule.Checked
2531|   If Me!mvGraticule.Checked Then
2532|      flgDrawGraticule = True
2533|      Call DrawGraticule(Me!picMap, GraticuleSize, GraticuleColor, GraticuleThickness)
2534|      Me!mvGraticule.Caption = "Map Graticule " & Format$(GraticuleSize, "#,###.#") & " "
      & CoordinateUnits()
2535|     Else
2536|      flgDrawGraticule = False
2537|      Me!mvGraticule.Caption = "Show Map Graticule"
2538|   End If
2539|  End Sub
2540|
2541|  Sub mvScaleBarMain_Click ()
2542|    Me!mvScaleBarMain.Checked = Not Me!mvScaleBarMain.Checked
2543|    If Me!mvScaleBarMain.Checked Then
2544|      iDrawScaleBar = 2 'To draw in lower right corner
2545|     Else
2546|      iDrawScaleBar = 0
2547|    End If
2548|    If iDrawScaleBar Then Call DrawScaleBar(Me!picMap, iDrawScaleBar)
2549|
2550|
2551|
2552|  End Sub
2553|
2554|  Sub mvSpecifyViewRegion_Click ()
2555|  On Error GoTo ERRmvSpecifyViewRegion_Click
2556|  frmViewPortOptions.lblDXMin = DataBounds.Left
2557|  frmViewPortOptions.lblDXMax = DataBounds.Right
```

```
2558| frmViewPortOptions.lblDYMax = DataBounds.Top
2559| frmViewPortOptions.lblDYMin = DataBounds.Bottom
2560|
2561| frmViewPortOptions.lblVXMin = ViewBounds.Left
2562| frmViewPortOptions.lblVXMax = ViewBounds.Right
2563| frmViewPortOptions.lblVYMax = ViewBounds.Top
2564| frmViewPortOptions.lblVYMin = ViewBounds.Bottom
2565|
2566| frmViewPortOptions.lblTrigger = Now
2567|  'NEW 950418
2568| 'New addition as of 950418 at KGS, named viewports and their handling.
2569| ' At this point, acquire a list of named viewports and add them to
2570| '  the combo box
2571|   frmViewPortOptions.cmbNamedViews.Clear
2572|   Dim nViews As Long, nNext As Long, nV As Long, nSerialNumber As Long
2573|   Set tblViews = MapDB.OpenTable("zWHEATViews")
2574|
2575|   tblViews.MoveFirst : tblViews.MoveLast : tblViews.MoveFirst
2576|   nViews = tblViews.RecordCount
2577|   Do Until tblViews.EOF
2578|   If Not IsNull(tblViews("ViewSerialNumber")) Then
2579|    nSerialNumber = tblViews("ViewSerialNumber")
2580|    If nSerialNumber > nNext Then nNext = nSerialNumber
2581|    frmViewPortOptions.cmbNamedViews.AddItem tblViews("ViewName")
2582|    'add serial number to the list
2583|    frmViewPortOptions.cmbNamedViews.ItemData(frmViewPortOptions.cmbNamedViews.NewIndex) = r
2584|    tblViews.MoveNext
2585|   End If
2586|   Loop
2587|   frmViewPortOptions.cmbNamedViews.Tag = nNext + 1
2588|
2589| frmViewPortOptions.Show 1
2590|
2591| 'Get rid of the named views table   'NEW 950418
2592|   tblViews.Close : Set tblViews = Nothing
2593|
2594|
2595| If frmViewPortOptions.cmdOK.FontItalic Then
2596|  Dim myZoomRect As RealRect
2597|  myZoomRect.Left = frmViewPortOptions.txtLeftMargin
2598|  myZoomRect.Right = frmViewPortOptions.txtRightMargin
2599|  myZoomRect.Top = frmViewPortOptions.txtTopMargin
2600|  myZoomRect.Bottom = frmViewPortOptions.txtBottomMargin
2601|  Call ZoomByRectangle(myZoomRect.Left, myZoomRect.Right, myZoomRect.Bottom,
     myZoomRect.Top)
2602| End If
2603| If moUnloadFormsAfterUse.Checked Then Unload frmViewPortOptions
2604|
2605|  Exit Sub
2606|
2607| ERRmvSpecifyViewRegion_Click:
2608| Select Case Err
2609|   Case 3021    'no current record
2610|    Resume Next
```

```
2611|    Case 3011 'The table does not yet exist
2612|      Call zDefineViewTable(MapDB)
2613|      Resume
2614|    Case Else
2615|      MsgBox "Error #" & Err & " occurred in mvSpecifyViewRegion_Click." & CRLF & Error
2616|      Err = 0
2617|      Exit Sub
2618|      Resume Next
2619|  End Select
2620|
2621|
2622|  End Sub
2623|
2624|  Sub mvStopDrawing_Click ()
2625|    cmdStopDrawing_Click
2626|  End Sub
2627|
2628|  Sub mvZoomByRectangle_Click ()
2629|   cmdStopDrawing_Click
2630|   PreviousTool = CurrentTool
2631|   CurrentTool = TOOL_ZoomByBox
2632|   LastCursor = SetCursorByName(Me.picMap, "ZoomByBox")
2633|   picMap.DrawWidth = 1
2634|   cmdStopDrawing_Click
2635|  End Sub
2636|
2637|  Sub mvZoomWayOut_Click ()
2638|  cmdStopDrawing_Click
2639|  Call ZoomWayOut
2640|  End Sub
2641|
2642|  Sub mwMapScale_Click ()
2643|  '   nDependents = nDependents + 1
2644|  '   ReDim Preserve FormsToUnloadFirst(1 To nDependents)
2645|  '   Dim Junk As Long
2646|  '   Set FormsToUnloadFirst(nDependents) = frmMapScale
2647|  '   Junk = SetParent(frmMapScale.hWnd, Me.hWnd)
2648|  '    frmMapScale.Show
2649|  End Sub
2650|
2651|  Sub myUnloadDependentForms ()
2652|
2653|  If nDependents = 0 Then Exit Sub
2654|  Dim nJunk As Long
2655|  For nJunk = 1 To nDependents
2656|    Unload FormsToUnloadFirst(nJunk): Set FormsToUnloadFirst(nJunk) = Nothing
2657|  Next nJunk
2658|
2659|
2660|
2661|
2662|
2663|  End Sub
2664|
```

```
Sub OLDmnuHelp_Click ()
   SendKeys "{F1}"
End Sub

Function OnePixelInWorldCoordinates () As Single
Dim BEFORE As Single, After As Single
Dim LLeft As Single, TTop As Single, WWidth As Single, HHeight As Single
LLeft = picMap.ScaleLeft
TTop = picMap.ScaleTop
WWidth = picMap.ScaleWidth
HHeight = picMap.ScaleHeight

BEFORE = picMap.CurrentX

picMap.ScaleMode = 3'pixel

picMap.CurrentX = picMap.CurrentX + 1

picMap.ScaleMode = 0

picMap.ScaleLeft = LLeft
picMap.ScaleTop = TTop
picMap.ScaleWidth = WWidth
picMap.ScaleHeight = HHeight
After = picMap.CurrentX
OnePixelInWorldCoordinates = After - BEFORE

End Function

Sub picMap_Click ()
If flgIgnoreNextClick Then : flgIgnoreNextClick = False: Exit Sub
'Don't beep at the user if you're drawing a rectangle.
If flgDrawRectangle Then : flgDrawRectangle = False: Exit Sub
If Me.picMap.ScaleMode = 3 Then : Beep: Exit Sub
  txtLastQSCoords = MousePt.X & "    " & MousePt.Y
'Moved the spatial querying into the Double click event, so that the
' drawing of lines and such can go easier.
'For analytical precision, might be best to run a spatial query from the
'   the command button and coordinate box.


End Sub

Sub picMap_DblClick ()
If lngSearchThemeID = 0 Then
  lblPercentDone = "No theme selected for spatial querying.  Use &Options menu"
  Beep
  Exit Sub
End If
MousePointer = 11
' Because of DOEVENTS, MouseX and MouseY might conceivably change during
'  the running or loading of a spatial query, so save them for output, as well
'  as the time, so if we're writing these to a file or table, they
' represent conditions at mouse_click, rather than some random later time.
```

```
2719  Dim QS_X As Single, QS_Y As Single, varTime As Variant, strMouseLoc  As String
2720
2721  varTime = Now
2722  QS_X = MouseX
2723  QS_Y = MouseY
2724  strMouseLoc = Format$(QS_X) & TabChar & Format$(QS_Y) & TabChar &
      Format$(varTime)
2725     On Error Resume Next
2726
2727  Dim nLine As Long
2728  If QSTheme.ThemeType = JOINTEDLINE Then
2729    For n = 1 To UBound(QSDataOut)
2730      nLine = QSDataOut(1).IDNum
2731      Call DrawLine(Lines(nLine), Lines(nLine).ZWidth, Lines(nLine).ZColor,
      Lines(nLine).ZStyle)
2732    Next n
2733  End If
2734
2735
2736  picMap.Circle (QS_X, QS_Y), 900, RGB(255, 128, 0)
2737  LastQS.X = QS_X
2738  LastQS.Y = QS_Y
2739
2740  ReDim Preserve QSDataOut(0 To 0)
2741  If lngSearchType = QS_RADIUSSEARCH Then
2742    picMap.Circle (QS_X, QS_Y), QSDataOut(0).Dist1, RGB(255, 0, 255)
2743  End If
2744
2745  dynSpatialQuery.Filter = ""
2746  Call SpatialSearch(QS_X, QS_Y, QSTheme, hGMemAllPoints, lngSearchType,
      dynSpatialQuery, QSDataOut(), lngQSHighlightColor, Me.picMap)
2747
2748
2749  If QSTheme.ThemeType = JOINTEDLINE Then
2750     nLine = QSDataOut(1).IDNum
2751     Call DrawLine(Lines(nLine), Lines(nLine).ZWidth, lngQSHighlightColor,
      Lines(nLine).ZStyle)
2752  End If
2753
2754  If QSTheme.ThemeType = JOINTEDLINE And lngSearchType = QS_LINEDIST Then
2755    pnlQSResults.Caption = "Distance to nearest line=" & Format(QSDataOut(1).Dist1,
      "0.00")
2756  End If
2757
2758  If lngSearchType = QS_CLOSESTPOINT Then
2759    pnlQSResults.Caption = "Distance to nearest point=" & Format(QSDataOut(1).Dist1,
      "0.00")
2760  End If
2761
2762  If lngSearchType = QS_RADIUSSEARCH Then
2763   pnlQSResults.Caption = UBound(QSDataOut) & " features found within " &
      QSDataOut(0).Dist1
2764  End If
2765
```

```
2766I If lngSearchType = QS_WITHINPOLY Then
2767I   pnlQSResults.Caption = UBound(QSDataOut) & " polygons enclose (" & QS_X & " , "
      & QS_Y & ")."
2768I   If UBound(QSDataOut) > 0 Then
2769I   Dim nPoints As Long, nStart As Long, iPoints As Integer
2770I     Dim nP2 As Long
2771I     For M = 1 To UBound(QSDataOut)
2772I
2773I     n = QSDataOut(M).IDNum
2774I   nPoints = Polys(n).nPoints: iPoints = nPoints: nP2 = 3 * nPoints
2775I   nStart = Polys(n).nStart
2776I   ReDim PolyIn(1 To nPoints) As RealPoint, GDIPoly(1 To nP2) As APIPoint
2777I   Call GetRlPntsFromGlobalMem(hGMemAllPoints, PolyIn(1), nStart, nPoints)
2778I   iPoints = SCScaleR4ToI2(ViewBounds, PixelRect, nPoints, PolyIn(1), nP2, GDIPoly(1))
2779I
2780I 'Draw  polygon interior with cross-hatch shading
2781I
2782I   Me.picMap.FillColor = lngQSHighlightColor
2783I   Me.picMap.FillStyle = 7
2784I   Me.picMap.DrawStyle = 5 'Turn off outlining
2785I   ReturnVal = Polygon(Me.picMap.hDC, GDIPoly(1), iPoints)
2786I     Next M
2787I   End If
2788I End If
2789I
2790I RestoreDrawingSettings
2791I
2792I ShowQSResults
2793I
2794I
2795I
2796I
2797I Set dynQSResults = dynSpatialQuery.CreateDynaset()
2798I flgUpdateGrdTxt = False
2799I  grdMapData.Rows = 2
2800I  grdMapData.FixedRows = 1
2801I  grdMapData.Row = 1
2802I  Call OverwriteGridFromDynaset(grdMapData, dynQSResults)
2803I  If MacroName <> "" Then 'GMD2 Macro stuff
2804I    RunMacro
2805I  End If
2806I 'Write data to a grid, for possible future insertion into a table.
2807I    Dim strQSOut As String
2808I    strQSOut = GetClipBoardTextFromDynaset(dynQSResults, 0)
2809I    strQSOut = ZAppendConstantTextToClipText(strQSOut, strMouseLoc)
2810I    Call SetGridFocusColumn(frmGridDataEdit.grdAttributes, 1)
2811I    Call FillGridFromClipBoardText(strQSOut, frmGridDataEdit.grdAttributes, 12&)
2812I
2813I flgUpdateGrdTxt = True
2814I
2815I    'Kluge 950609 GWPouch at KGS
2816I    ' Added code to allow E-Map to open and run a file every time
2817I    '  the user selects a feature with a spatial query.  The file
2818I    '  to open must be in a field called OnClickOpenFile, and should
```

```
2819|     ' be a filename with an extension recognized by file manager.
2820|     If idxOnClickOpenFile >= 0 Then 'idxOnClickOpenFile can be zero if it's the first field
2821|    Call OpenQSFiles(dynQSResults)
2822|     End If
2823|
2824|
2825|
2826|
2827|  MousePointer = 0
2828|
2829|  End Sub
2830|
2831|  Sub picMap_GotFocus ()
2832|   LastCursor = SetCursorByHCURSOR(Me.picMap, ToolPick(CurrentTool).hCursor)
2833|  End Sub
2834|
2835|  Sub picMap_KeyDown (KeyCode As Integer, Shift As Integer)
2836|  Static Mult As Integer, XShift As Integer, YShift As Integer
2837|  If KeyCode = KEY_ESCAPE Then
2838|   flgStopDrawing = True
2839|  End If
2840|  'Allow CTRL-Break and ESCAPE to cancel drawing.
2841|  'Maybe these should be set in an INI file.
2842|  If KeyCode = 17 And Shift = 2 Then flgStopDrawing = True
2843|  '
2844|
2845|  Mult = 1
2846|  XShift = 0
2847|  YShift = 0
2848|  If (Shift = 2) Then Mult = 5'for control key
2849|  If (Shift = 4) Then Mult = 20'for alt key
2850|  If KeyCode = KEY_Left Or KeyCode = KEY_NUMPAD4 Then XShift = -Mult
2851|  If KeyCode = KEY_RIGHT Or KeyCode = KEY_NUMPAD6 Then XShift = Mult
2852|  If KeyCode = KEY_UP Or KeyCode = KEY_NUMPAD8 Then YShift = -Mult
2853|  If KeyCode = KEY_DOWN Or KeyCode = KEY_NUMPAD2 Then YShift = Mult
2854|  If YShift Or XShift Then
2855|     Call ShiftCursor(XShift, YShift)
2856|  End If
2857|  'Trigger a click, and thus a query or line drawing, for return or q
2858|  If KeyCode = KEY_RETURN Or KeyCode = 81 Then Call picMap_DblClick
2859|  If KeyCode = 88 Then   'X marks the spot.  Record current location onto clipboard.
2860|     'Call cmdCopyCoordinates_Click
2861|     Clipboard.SetText txtCurrCoords
2862|     Me.picMap.SetFocus
2863|  End If
2864|  If KeyCode = 72 Then Call cmdHighlightPosition_Click 'For h, hightlight
2865|  'If KeyCode = 80 Then Call cmdPasteCoordinates_Click   'For p, paste the coordinates
       to clipboard.
2866|
2867|
2868|  If KeyCode = 32 Then
2869|     If CurrentTool = TOOL_DRAWLINE Or CurrentTool = TOOL_DRAWPOLYGON
      Then
2870|     'CurrentTool = PreviousTool
```

```
2871|     Dim strLineOut As String, TempString
2872|     For n = 1 To nPtsInLine - 1
2873|   Me!picMap.Line (UserLine(n).X, UserLine(n).Y)-(UserLine(n + 1).X, UserLine(n + 1).Y),
      RGB(255, 128, 0)
2874|    strLineOut = strLineOut & UserLine(n).X & "," & UserLine(n).Y & ";"
2875|     Next n
2876|     If CurrentTool = TOOL_DRAWLINE Then
2877|      nUserLines = nUserLines + 1
2878|     TempString = "A" & TabChar & "Name Goes Here" & TabChar & strLineOut &
      TabChar & Now & TabChar & nUserLines
2879|      frmUserLines.grdAttributes.AddItem TempString
2880|     End If
2881|     If CurrentTool = TOOL_DRAWPOLYGON Then
2882|      Me!picMap.Line (UserLine(nPtsInLine).X, UserLine(nPtsInLine).Y)-(UserLine(1).X,
      UserLine(1).Y), RGB(255, 128, 0)
2883|      nUserPolys = nUserPolys + 1
2884|      strLineOut = strLineOut & UserLine(1).X & "," & UserLine(1).Y & ";"
2885|      TempString = "A" & TabChar & "Name Goes Here" & TabChar & strLineOut &
      TabChar & Now & TabChar & nUserPolys
2886|      frmUserPolys.grdAttributes.AddItem TempString
2887|     End If
2888|     nPtsInLine = 0
2889|     ReDim UserLine(1)
2890|    End If 'Close if on CurrentTool
2891| End If
2892|
2893| 'If KeyCode = 32 Then Me.picMap.SetFocus 'Alternative to keeping the setfocus stuff in
      mouseMove, which
2894|             ' blew up due to memory leakage from repeated ' Me.picMap is
      ActiveForm.Control testing
2895|
2896| '
2897| End Sub
2898|
2899| Sub picMap_LostFocus ()
2900| M = SetCursorByHCURSOR(Me.picMap, OriginalCursorStyle)
2901| End Sub
2902|
2903| Sub picMap_MouseDown (Button As Integer, Shift As Integer, X As Single, Y As Single)
2904| If Me.picMap.ScaleMode = 3 Then : Beep: Exit Sub
2905| If Button <> 1 Or Shift <> 0 Then : Me.SetFocus : flgIgnoreNextClick = True: Exit Sub
2906|
2907| StartX = X: StartY = Y:
2908|
2909| Select Case CurrentTool
2910|  Case TOOL_ZoomByBox
2911|   ZoomRect.Left = X
2912|   ZoomRect.Top = Y
2913|   OldX = X: OldY = Y
2914|   flgDrawRectangle = True
2915|  Case TOOL_DRAWLINE, TOOL_DRAWPOLYGON
2916|   flgIgnoreNextClick = True
2917|   If nPtsInLine = 0 Then
2918|   nPtsInLine = 1
```

```
    ReDim UserLine(1)
    UserLine(1).X = X: UserLine(1).Y = Y
    OldX = X: OldY = Y
      Else
    'Me.picMap.DrawMode = 7
    'Me.picMap.Line (UserLine(nPtsInLine).X, UserLine(nPtsInLine).Y)-(X, Y), RGB(255,
255, 128)
    'OldX = X: OldY = Y
    Me.picMap.DrawMode = 7
    Me.picMap.Line (UserLine(nPtsInLine).X, UserLine(nPtsInLine).Y)-(OldX, OldY),
RGB(255, 255, 255) 'Erase old line
    Me.picMap.Line (UserLine(nPtsInLine).X, UserLine(nPtsInLine).Y)-(X, Y), RGB(255,
255, 255) 'draw new line
    OldX = X: OldY = Y

'       Call picMap_MouseMove(Button, Shift, X, Y)
    End If
End Select
End Sub

Sub picMap_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
If Me.picMap.ScaleMode = 3 Then Exit Sub  'No beep for just a move
MouseX = X: MouseY = Y
MousePt.X = X: MousePt.Y = Y
txtCurrCoords = MousePt.X & "  " & MousePt.Y
'grdPosition.Col = 0: grdPosition = Format$(X): grdPosition.Col = 1: grdPosition =
Format$(Y)

If QSRasterGridNumber Then Call ShowRasterValue(MouseX, MouseY,
QSRasterGridNumber)

Select Case CurrentTool
  Case TOOL_DRAWLINE, TOOL_DRAWPOLYGON
    If Button And 1 Then
    Me.picMap.DrawMode = 7
    Me.picMap.Line (UserLine(nPtsInLine).X, UserLine(nPtsInLine).Y)-(OldX, OldY),
RGB(255, 255, 255) 'Erase old line
    Me.picMap.Line (UserLine(nPtsInLine).X, UserLine(nPtsInLine).Y)-(X, Y), RGB(255,
255, 255) 'draw new line
    OldX = X: OldY = Y
    End If
End Select


 'pnl3DX = Format$(X)
 'pnl3DY = Format$(Y)
If flgDrawRectangle And Button Then
    'Undraw last rectangle
    Call ZQDrawBox(StartX, StartY, OldX, OldY, RGB(255, 255, 128))
    'Draw the new rectangle
    Call ZQDrawBox(StartX, StartY, X, Y, RGB(255, 255, 128))
    OldX = X
    OldY = Y
    Dim Temp As String
```

```
2966I     Temp = "Zooming by Rectangle." & CRLF & "Starting at X=" & StartX & "  Y=" &
      StartY & CRLF
2967I     Temp = Temp & "Going to X=" & X & "  Y=" & Y & CRLF
2968I     Temp = Temp & " X_Size=" & (X - StartX) & "  Y_Size= " & (Y - StartY)
2969I     pnl3dInstructions = Temp
2970I  End If
2971I
2972I  End Sub
2973I
2974I  Sub picMap_MouseUp (Button As Integer, Shift As Integer, X As Single, Y As Single)
2975I  If Me.picMap.ScaleMode = 3 Then : Beep: Exit Sub
2976I  'This should probably be replaced by a case-select later on, and should set a
      DON'T-RUN-SPATIAL-QUERY flag
2977I  Dim Temp As Single, LastFeature As Long
2978I  Select Case CurrentTool
2979I    Case TOOL_DRAWPOINT
2980I      nUserPoints = nUserPoints + 1
2981I      Dim TempString As String
2982I      TempString = "A" & TabChar & "Name Goes Here" & TabChar & X & TabChar & Y &
      TabChar & Now & TabChar & nUserPoints
2983I      frmUserPoints.grdAttributes.AddItem TempString
2984I      Call ZBeachBall(X, Y)
2985I      'ReDim strUserPointsOut(LastFeature)
2986I      'strUserPointsOut(LastFeature) = LastFeature & TabChar & Now & TabChar &
      Format$(X) & TabChar & Format$(Y) & CRLF
2987I    Case TOOL_DRAWLINE, TOOL_DRAWPOLYGON
2988I      If Button And 1 Then
2989I    Me!picMap.DrawMode = 7
2990I    Me!picMap.Line (UserLine(nPtsInLine).X, UserLine(nPtsInLine).Y)-(OldX, OldY),
      RGB(255, 255, 128)'Erase old line
2991I    Me!picMap.DrawMode = 13
2992I    Me!picMap.Line (UserLine(nPtsInLine).X, UserLine(nPtsInLine).Y)-(X, Y), RGB(0, 0,
      128)'draw new line
2993I    nPtsInLine = nPtsInLine + 1
2994I    ReDim Preserve UserLine(nPtsInLine)
2995I    UserLine(nPtsInLine).X = X: UserLine(nPtsInLine).Y = Y
2996I    'PointString = PointString & Format$(X) & TabChar & Format$(Y) & CRLF
2997I    'PointString = PointString & "n=" & nPtsInLine & ";X=" & Format$(X) & ";Y=" &
      Format$(Y)
2998I      End If
2999I    Case TOOL_ZoomByBox
3000I    ZoomRect.Right = X: ZoomRect.Bottom = Y
3001I      If ZoomRect.Bottom > ZoomRect.Top Then : Temp = ZoomRect.Bottom:
      ZoomRect.Bottom = ZoomRect.Top: ZoomRect.Top = Temp
3002I      If ZoomRect.Left > ZoomRect.Right Then : Temp = ZoomRect.Left: ZoomRect.Left =
      ZoomRect.Right: ZoomRect.Right = Temp
3003I    'Restore previous cursor
3004I    CurrentTool = PreviousTool
3005I    LastCursor = SetCursorByHCURSOR(Me.picMap, ToolPick(CurrentTool).hCursor)
3006I    pnl3dInstructions = ""
3007I    Call ZoomByRectangle(ZoomRect.Left, ZoomRect.Right, ZoomRect.Bottom,
      ZoomRect.Top)
3008I    Call cmdRefreshMap_Click
3009I
```

```
End Select

End Sub

Sub picMap_paint ()
'
If flgSkipRepainting Then Exit Sub
If flgStopDrawing Then
  flgStopDrawing = False
  Exit Sub
End If

   Me.picMap.DrawMode = 13

   'picMap.Cls
   'picMap.Line (ViewBounds.Left, ViewBounds.Top)-(ViewBounds.Right,
ViewBounds.Bottom), MapBaseColor, BF


   Call ShowThemes

End Sub

Sub picStatusBar_Resize ()
 "If (picStatusBar.Width > pnl3DMessage.Left) Then pnl3DMessage.Width =
picStatusBar.Width - pnl3DMessage.Left - .25
 'If (pnl3dGrid.Left > 4) Then grdMapData.Width = pnl3dGrid.Width - 2
End Sub

Sub pnl3dSide_Click ()
 'No resize event for this object.  Must use the form's resize event.
End Sub

Sub Print_QSResults ()
If moShowLastQuery.Checked = False Then Exit Sub
 Dim n As Long, nFeatures As Long
 Dim lb As Long, ub As Long
 On Error Resume Next
 lb = LBound(QSDataOut)
 ub = UBound(QSDataOut)
 Printer.ForeColor = lngQSHighlightColor
 Printer.FillColor = lngQSHighlightColor
 If LastQS.X >= ViewBounds.Left And LastQS.X <= ViewBounds.Right And LastQS.Y >=
ViewBounds.Bottom And LastQS.Y <= ViewBounds.Top Then Call
ZPrinterBeachBall(LastQS.X, LastQS.Y)
 RestorePrinterSettings
 If QSTheme.ThemeType = ONEPOINT Then
  Dim nPt As Long, nStart As Long, Pt As RealPoint, flgSkipIt As Integer

  Printer.ForeColor = lngQSHighlightColor
  For n = lb To ub
 flgSkipIt = False
 nPt = QSDataOut(n).IDNum
 nStart = Points(nPt).nStart
```

```
3060|    Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pt, nStart, lngOne)
3061|    If Pt.X < ViewBounds.Left Then flgSkipIt = True
3062|    If Pt.X > ViewBounds.Right Then flgSkipIt = True
3063|    If Pt.Y < ViewBounds.Bottom Then flgSkipIt = True
3064|    If Pt.Y > ViewBounds.Top Then flgSkipIt = True
3065|    SetFontInPrinter Points(nPt).ZFontCode
3066|    If flgSkipIt Then GoTo BotShowPoints5
3067|    Printer.FontSize = Points(nPt).ZSize
3068|    If flgSkipIt Then GoTo BotShowPoints5
3069|    Printer.FontBold = True
3070|    If flgSkipIt Then GoTo BotShowPoints5
3071|    Printer.CurrentX = Pt.X
3072|    Printer.CurrentY = Pt.Y
3073|    Printer.Print Chr$(Points(nPt).ZSymbol)
3074| BotShowPoints5:
3075|     Next
3076|   End If
3077|
3078|
3079|
3080|
3081|
3082|   If lngSearchType = QS_RADIUSSEARCH Then
3083|    'Printer.DrawStyle = 5
3084|    Printer.DrawWidth = 3
3085|    Printer.Circle (LastQS.X, LastQS.Y), QSDataOut(0).Dist1,
      lngQSHighlightColor'RGB(255, 0, 255)
3086|   End If
3087|
3088|   'Draw circles around the points found
3089|   For n = lb To ub
3090|     Printer.Circle (QSDataOut(n).X, QSDataOut(n).Y), sngQSHighlightRadius,
      lngQSHighlightColor
3091|   Next n
3092|
3093| 'Dim nLine As Long
3094| 'If QSTheme.ThemeType = JOINTEDLINE Then
3095| '    nLine = QSDataOut(1).IDNum
3096| '    Call DrawLine(Lines(nLine), Lines(nLine).ZWidth, lngQSHighlightColor,
      Lines(nLine).ZStyle)
3097| 'End If
3098|
3099| End Sub
3100|
3101| Sub PrintMap (ViewRect As RealRect, PrintRect As RealRect)
3102| On Error GoTo ERRPrintMap
3103| Dim PrintScale As RealRect, mm As Long, nTh As Long
3104|    Screen.MousePointer = 11
3105| 'ThemesToPrint is an array full of ThemesToPrint to be printed.
3106| 'ViewRect is the area, in world coordinates, to be printed
3107| 'PrintRect is the location, on the PRINTER, where the map of the area
3108| '         ViewRect should show up.
3109| 'The units of PrintRect are in MILLIMETERS.
3110|
```

```
3111  Dim PrinterHeight As Single, PrinterWidth As Single
3112  RestorePrinterSettings
3113  Printer.ScaleMode = 6'for millimeters
3114  PrinterHeight = Printer.ScaleHeight
3115  PrinterWidth = Printer.ScaleWidth
3116
3117  'These ranges leave some space on most printers, to accomodate the
3118  ' the rollers and such.  However, these will be used to define the
3119  ' the edges.
3120
3121  If frmPrinterOptions.optMapScale Then
3122    'if printing at a particular scale, add the scale factor in words
3123     Printer.CurrentX = 12
3124     Printer.CurrentY = Printer.ScaleHeight - 12
3125     Printer.FontName = "Arial": Printer.FontBold = False: Printer.FontItalic = False:
       Printer.FontSize = 8
3126     Printer.ForeColor = RGB(0, 0, 0)
3127     Printer.Print "Map printed at a scale of 1:" & frmPrinterOptions.txtPrintScale
3128  End If
3129
3130  'Figure out whether the width or height will be the limiting
3131  '  factor.
3132  Dim XScale As Single, YScale As Single, UsedScale As Single
3133  XScale = Abs((PrintRect.Right - PrintRect.Left) / (ViewRect.Right - ViewRect.Left))
3134  YScale = Abs((PrintRect.Top - PrintRect.Bottom) / (ViewRect.Top - ViewRect.Bottom))
3135  If XScale < YScale Then
3136     UsedScale = XScale
3137   Else
3138     UsedScale = YScale
3139  End If
3140
3141
3142
3143
3144
3145
3146     PrintScale.Left = ViewRect.Left - PrintRect.Left / UsedScale
3147     PrintScale.Top = ViewRect.Top + PrintRect.Top / UsedScale
3148     PrintScale.Bottom = PrintScale.Top - Printer.ScaleHeight / UsedScale
3149     PrintScale.Right = PrintScale.Left + Printer.ScaleWidth / UsedScale
3150
3151
3152
3153
3154     Printer.Scale (PrintScale.Left, PrintScale.Top)-(PrintScale.Right, PrintScale.Bottom)
3155
3156  SetPrinterViewRect ViewRect
3157
3158  If flgStopPrinting Then Exit Sub
3159
3160  For mm = 1 To nThemesShownHere
3161     If Not flgSkipRepainting Then DoEvents
3162     If flgStopDrawing Then GoTo ExitPrintMap
3163   If ThemeOrder(mm) > 0 Then
```

```
3164|     'If moLayerName(mm).Checked Then
3165|       nTh = Abs(ThemeOrder(mm))
3166|     Select Case Themes(nTh).ThemeType
3167|    Case JOINTEDLINE:  Call PrintLines5(Themes(nTh).FirstLoc, Themes(nTh).LastLoc,
      nTh)
3168|    Case ONEPOINT:     Call PrintPoints3(Themes(nTh).FirstLoc, Themes(nTh).LastLoc,
      nTh)
3169|    Case TILEBLOCK:    Call PrintTiles4(Themes(nTh).FirstLoc, Themes(nTh).LastLoc,
      nTh)
3170|    Case HATCHLINE:    Call PrintHatchLines3(Themes(nTh).FirstLoc,
      Themes(nTh).LastLoc, nTh)
3171|    Case SOLIDPOLYGON: Call PrintPolygons5(Themes(nTh).FirstLoc,
      Themes(nTh).LastLoc, nTh)
3172|    Case LABELTEXT:    Call PrintLabels1(Themes(nTh).FirstLoc, Themes(nTh).LastLoc,
      nTh)
3173|    Case RASTERGRID:   Call PrintRasterGrid(Themes(nTh).FirstLoc,
      Themes(nTh).LastLoc, nTh)
3174|    Case Else: Beep: MsgBox "Unidentified theme type. " & Themes(nTh).ThemeName
3175|     End Select
3176|  End If
3177|  RestorePrinterSettings
3178| Next mm
3179|
3180| ExitPrintMap:
3181|  RestorePrinterSettings
3182| 'Display the current spatial selection
3183| If moShowLastQuery.Checked Then Print_QSResults
3184|  'PrinterCircleQSResults QSDataOut(), lngQSHighlightColor, sngQSHighlightRadius,
      LastQS
3185|  ' If lngSearchType = QS_RADIUSSEARCH Then
3186|  '  Printer.Circle (LastQS.X, LastQS.Y), QSDataOut(0).Dist1, RGB(255, 0, 255)
3187|  ' End If
3188|  'End If
3189|
3190| RestorePrinterSettings
3191| Printer.Line (ViewRect.Left, ViewRect.Top)-(ViewRect.Right, ViewRect.Bottom), RGB(0,
      0, 0), B
3192| Printer.ForeColor = RGB(0, 0, 0)
3193|
3194|  RestorePrinterSettings
3195|
3196| If flgDrawGraticule Then
3197|    Call PrintGraticule(ViewRect, GraticuleColor, GraticuleThickness)
3198|   Else
3199|      Dim Dummy As Integer
3200|      'Print coordinates, whether the user wanted them or not.
3201|       Printer.FontSize = 8
3202|       Printer.FontName = "Arial"
3203|      'Print bounds of top
3204|       Dummy = SetTextAlign(Printer.hDC, TA_BOTTOM + TA_CENTER)
3205|       Printer.CurrentX = (ViewRect.Left + ViewRect.Right) / 2
3206|       Printer.CurrentY = ViewRect.Top
3207|       Printer.Print ViewRect.Top
3208|      'Print Bounds of bottom
```

```
3209|       Dummy = SetTextAlign(Printer.hDC, TA_TOP + TA_CENTER)
3210|       Printer.CurrentY = ViewRect.Bottom
3211|       Printer.CurrentX = (ViewRect.Left + ViewRect.Right) / 2
3212|       Printer.Print ViewRect.Bottom
3213|      'Print Bounds of left side
3214|       Dummy = SetTextAlign(Printer.hDC, TA_BASELINE + TA_RIGHT)
3215|       Printer.CurrentY = (ViewRect.Bottom + ViewRect.Top) / 2
3216|       Printer.CurrentX = (ViewRect.Left)
3217|       Printer.Print ViewRect.Left
3218|      'Print Bounds of left side
3219|       Dummy = SetTextAlign(Printer.hDC, TA_BASELINE + TA_LEFT)
3220|       Printer.CurrentY = (ViewRect.Bottom + ViewRect.Top) / 2
3221|       Printer.CurrentX = (ViewRect.Right)
3222|       Printer.Print ViewRect.Right
3223| End If
3224|
3225|  If iDrawScaleBar Then Call PrintScaleBar(ViewRect, iDrawScaleBar)
3226|
3227|
3228| Exit Sub
3229| ERRPrintMap:
3230| Printer.EndDoc
3231| Err = 0
3232| Exit Sub
3233| Resume
3234|
3235|
3236|
3237| End Sub
3238|
3239| Sub RestoreDrawingSettings ()
3240| 'Restore the original drawing tools
3241|   Me.picMap.DrawStyle = 0  'Solid lines
3242|   Me.picMap.DrawMode = 13  'CopyPen
3243|   Me.picMap.DrawWidth = 1  'Thin line
3244|   Me.picMap.FillStyle = 1  'Transparent fill
3245|    M = SetTextAlign(Me!picMap.hDC, 0)
3246| End Sub
3247|
3248| Sub RunMacro ()
3249|  Dim nL As Long
3250|  Dim QSClip As String
3251|  On Error GoTo ERRRunMacro
3252|  grdMapData.FixedCols = 0: grdMapData.FixedRows = 0
3253|
3254|  grdMapData.SelStartCol = 1: grdMapData.SelEndCol = grdMapData.Cols - 1
3255|  grdMapData.SelStartRow = 0: grdMapData.SelEndRow = grdMapData.Rows - 1
3256|  QSClip = grdMapData.Rows
3257|  For nL = 1 To grdMapData.Cols - 2
3258|    QSClip = QSClip & TabChar & Now
3259|  Next nL
3260|  QSClip = QSClip & CRLF & grdMapData.Clip
3261|  Clipboard.Clear
3262|  Clipboard.SetText QSClip
```

```
3263|  grdMapData.FixedRows = 1: grdMapData.FixedCols = 1
3264|  flgStopDrawing = True
3265|  AppActivate KeysToSend(1)
3266|  For nL = 2 To UBound(KeysToSend)
3267|    SendKeys KeysToSend(nL)
3268|  Next nL
3269|  flgStopDrawing = False
3270|
3271|
3272|  Exit Sub
3273|  ERRRunMacro:
3274|    Select Case Err
3275|   Case 5 'Could not find application
3276|   Beep
3277|   Beep
3278|   Dim Message As String
3279|   Message = "You chose to run a macro, stored in ".& UCase$(MacroName) & " ,
      everytime you run a spatial query, probably to graph the data." & CRLF
3280|   Message = Message & "E-Map cannot find the program-data file specified as " &
      KeysToSend(1) & CRLF
3281|   Message = Message & "Before proceeding, you must start the other application and
      possibly open the data file."
3282|   MsgBox Message
3283|   Resume
3284|     Case Else
3285|   MsgBox "Error #" & Err & " occurred in RunMacro." & CRLF & Error
3286|   Err = 0
3287|   Exit Sub
3288|   Resume
3289|    End Select
3290|
3291|
3292|  End Sub
3293|
3294|  Sub RunMacroForAllPoints (nTheme As Long)
3295|  Dim nP As Long
3296|  For nP = Themes(nTheme).FirstLoc To Themes(nTheme).LastLoc
3297|
3298|    dynSpatialQuery.Filter = "zqFeatureID = " & Points(nP).FeatureID
3299|    Set dynQSResults = dynSpatialQuery.CreateDynaset()
3300|    flgUpdateGrdTxt = False
3301|    grdMapData.Rows = 2
3302|    grdMapData.FixedRows = 1
3303|   grdMapData.Row = 1
3304|   Call OverwriteGridFromDynaset(grdMapData, dynQSResults)
3305|   If MacroName <> "" Then 'GMD2 Macro stuff
3306|      RunMacro
3307|   End If
3308|  Next
3309|  End Sub
3310|
3311|  Sub ShowHatchLines3 (FirstLine As Long, LastLine As Long, nTheme As Long) '
3312|  'GWPouch 940531  The ShowTYPE3 series of subroutines checks to see
3313|  '  if the desired feature lies inside, or close to the viewport
```

```
' before trying to draw it.  This should avoid many overflow errors
' and the like.
On Error GoTo ERRShowHatchLines3

Dim n As Long, nLines As Long, flgSkipIt As Integer
If (UBound(HatchLines) = 0) Then Exit Sub
nLines = LastLine - FirstLine + 1
pnl3dPercentDone.FloodPercent = 0
lblPercentDone = "Drawing hatch lines from " & Themes(nTheme).ThemeName

For n = FirstLine To LastLine
 DoEvents
 If flgStopDrawing Then Exit Sub
 flgSkipIt = False
' Check whether the polygon, or whatever, of which this is the hatching
' lies outside the current viewport.
 If HatchLines(n).MaxX < ViewBounds.Left Then flgSkipIt = True
 If HatchLines(n).MinX > ViewBounds.Right Then flgSkipIt = True
 If HatchLines(n).MinY > ViewBounds.Top Then flgSkipIt = True
 If HatchLines(n).MaxY < ViewBounds.Bottom Then flgSkipIt = True

   picMap.DrawWidth = HatchLines(n).ZWidth
   picMap.ForeColor = HatchLines(n).ZColor
   picMap.DrawStyle = HatchLines(n).ZStyle

   If flgSkipIt = False Then
 Call DrawGlobalHatchLine(hGMemAllPoints, HatchLines(n).nStart,
HatchLines(n).nPoints)
   End If

   Me.pnl3dPercentDone.FloodPercent = (n - FirstLine) / nLines * 100
Next n
RestoreDrawingSettings
pnl3dPercentDone.FloodPercent = 100
lblPercentDone = "Done drawing hatch lines from " & Themes(nTheme).ThemeName
Exit Sub

ERRShowHatchLines3:
flgSkipIt = True
Resume Next



End Sub

Sub ShowLabels1 (FirstLabel As Long, LastLabel As Long, nTheme As Long)
Dim n As Long, Pt  As RealPoint
If (UBound(Labels) = 0) Then Exit Sub
Dim flgSkipIt As Integer, ViewBoundsPlus As RealRect
'ViewBoundsPlus will contain a rectangle, ten percent larger on each side,
' than the current ViewPort.  This will allow Labels near the edge to
' be partially shown, even when they don't quite lie in the rectangle.
Call MagnifyRectangle(ViewBounds, ViewBoundsPlus, .1)

```

```
3367|
3368| On Error GoTo ERRShowLabels1
3369|
3370| DummyShort = SetTextAlign(Me.picMap.hDC, TA_BASELINE)
3371|
3372| nLabels = LastLabel - FirstLabel + 1
3373|
3374| Me!pnl3dPercentDone.FloodPercent = 0
3375| Me!lblPercentDone = "Drawing text labels from " & Themes(nTheme).ThemeName
3376|    SetFontInPIC -1, Me.picMap
3377| Dim nStart As Long
3378| For n = FirstLabel To LastLabel
3379|    flgSkipIt = False
3380|   DoEvents
3381|   If flgStopDrawing Then Exit Sub
3382|    nStart = Labels(n).nStart
3383|    Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pt, nStart, lngOne)
3384| 'Check whether the point falls in the viewbounds.
3385|
3386|    If Pt.X < ViewBoundsPlus.Left Then flgSkipIt = True
3387|    If Pt.X > ViewBoundsPlus.Right Then flgSkipIt = True
3388|    If Pt.Y < ViewBoundsPlus.Bottom Then flgSkipIt = True
3389|    If Pt.Y > ViewBoundsPlus.Top Then flgSkipIt = True
3390|    Me.picMap.FontSize = Labels(n).FontSize
3391|    SetFontInPIC Labels(n).FontCode, Me.picMap
3392|    'SetFontCodeInPICxxx Labels(N).FontCode, Me.picMap  'New 940701
3393|    If flgSkipIt Then GoTo BotShowLabels
3394|
3395|    If flgSkipIt Then GoTo BotShowLabels
3396|    Me.picMap.ForeColor = Labels(n).ZColor
3397|    DummyShort = SetTextAlign(Me.picMap.hDC, Labels(n).Alignment)
3398|    Me.picMap.FontTransparent = Not Labels(n).Opaque
3399|    If flgSkipIt Then GoTo BotShowLabels
3400|    Me.picMap.CurrentX = Pt.X
3401|    '              If flgSkipIt Then GoTo BotShowPoints2
3402|    Me.picMap.CurrentY = Pt.Y
3403|    '              If flgSkipIt Then GoTo BotShowPoints2
3404|    Me.picMap.Print Trim$(Labels(n).Text)
3405|    '              If flgSkipIt Then GoTo BotShowPoints2
3406| BotShowLabels:
3407|    Me!pnl3dPercentDone.FloodPercent = (n - FirstLabel) / nLabels * 100
3408| Next n
3409| RestoreDrawingSettings
3410| Me!lblPercentDone = "Done Drawing": Me!pnl3dPercentDone.FloodPercent = 100
3411|  Me!pnl3dPercentDone.FloodPercent = 0: Me!lblPercentDone = ""
3412| Exit Sub
3413| ERRShowLabels1:
3414| If Err = 6 Then
3415|    flgSkipIt = True
3416|    Resume Next
3417| End If
3418| MsgBox "Error #" & Err & " occurred in ShowLabels1." & CRLF & Error
3419| Err = 0
3420| Exit Sub
```

```
3421|  Resume Next
3422|
3423|
3424|  End Sub
3425|
3426|  Sub ShowLines5 (FirstLine As Long, LastLine As Long, nTheme As Long)
3427|  '940621  This comes full circle.  This is now a descendant of the
3428|  ' ShowPolygon5, which is fast.
3429|  'GWPouch 940613
3430|  'Displays solid color polygons, I hope.
3431|  'Derived from ShowLines3.
3432|
3433|  'This will check to see whether a line overlaps the view rectangle
3434|  '  before plotting it.
3435|  Dim n As Long, nLines As Long, nStart As Long, nPoints As Long, nP2 As Long
3436|  Dim flgSkipToBottom As Integer, iPoints As Integer
3437|  On Error GoTo ERRShowLines5
3438|
3439|  If (UBound(Lines) = 0) Then Exit Sub
3440|
3441|  nLines = LastLine - FirstLine + 1
3442|  Me!pnl3dPercentDone.FloodPercent = 0
3443|  Me!lblPercentDone = "Drawing lines from " & Themes(nTheme).ThemeName
3444|
3445|  'SaveScaling
3446|    SaveScaling Me.picMap, OriginalScale
3447|    GetPixelBounds Me.picMap, PixelRect
3448|    Me.picMap.ScaleMode = 3 'Put into pixel mode
3449|
3450|
3451|  For n = FirstLine To LastLine
3452|    flgSkipToBottom = False
3453|    DoEvents
3454|    If flgStopDrawing Then GoTo LeaveShowLines5 'Graceful exit that restores scaling
3455|    'Test whether the line should be plotted
3456|          'XXXX LOGIC PROBLEM.  YOU CAN HAVE A LINE WHOSE BOUNDS ARE
       ABOVE
3457|          ' AND BELOW THE CURRENT VIEW WINDOW THAT STILL NEEDS TO BE
       DRAWN.
3458|    If Lines(n).MaxX < ViewBounds.Left Then flgSkipToBottom = True
3459|    If Lines(n).MinX > ViewBounds.Right Then flgSkipToBottom = True
3460|    If Lines(n).MinY > ViewBounds.Top Then flgSkipToBottom = True
3461|    If Lines(n).MaxY < ViewBounds.Bottom Then flgSkipToBottom = True
3462|    nPoints = Lines(n).nPoints: iPoints = nPoints: nP2 = 3 * nPoints
3463|    nStart = Lines(n).nStart
3464|    If nPoints < 2 Then flgSkipToBottom = True
3465|  If flgSkipToBottom = False Then
3466|    ReDim LineIn(1 To nPoints) As RealPoint, GDILine(1 To nP2) As APIPoint
3467|    Call GetRlPntsFromGlobalMem(hGMemAllPoints, LineIn(1), nStart, nPoints)
3468|    iPoints = SCScaleR4To2(ViewBounds, PixelRect, nPoints, LineIn(1), nP2, GDILine(1))
3469|
3470|
3471|  'Draw line
3472|
```

```
3473|    Me.picMap.DrawWidth = Lines(n).ZWidth
3474|    Me.picMap.ForeColor = Lines(n).ZColor
3475|    Me.picMap.DrawStyle = Lines(n).ZStyle
3476|    If flgSkipToBottom = False Then
3477|       ReturnVal = Polyline(Me.picMap.hDC, GDILine(1), iPoints)
3478|       '"If ReturnVal = 0 Then Beep: dDebug.Print "Problem showing line #" & n & " in
      frmMap.picMap"
3479|    End If
3480| End If
3481|
3482|    Me.pnl3dPercentDone.FloodPercent = (n - FirstLine) / nLines * 100
3483| Next n
3484|
3485| LeaveShowLines5:
3486|  Me!lblPercentDone = "Done Drawing": Me!pnl3dPercentDone.FloodPercent = 100
3487|  Me!pnl3dPercentDone.FloodPercent = 0: Me!lblPercentDone = ""
3488|  RestoreScaling Me.picMap, OriginalScale
3489|  RestoreDrawingSettings
3490|  Me.picMap.FillStyle = 1 'Turn off shape filling
3491|  Me.picMap.DrawStyle = 0  'Turn on outlining
3492| Exit Sub
3493|
3494|
3495| ERRShowLines5:
3496|
3497| flgSkipToBottom = True
3498| Resume Next
3499|
3500|
3501|
3502|
3503|
3504| End Sub
3505|
3506| Sub ShowPoints3 (FirstPoint As Long, LastPoint As Long, nTheme As Long)
3507| Dim n As Long, Pt  As RealPoint
3508| If (UBound(Points) = 0) Then Exit Sub
3509| Dim flgSkipIt As Integer, ViewBoundsPlus As RealRect
3510| 'ViewBoundsPlus will contain a rectangle, ten percent larger on each side,
3511| '  than the current ViewPort.  This will allow points near the edge to
3512| '  be partially shown, even when they don't quite lie in the rectangle.
3513| Call MagnifyRectangle(ViewBounds, ViewBoundsPlus, .1)
3514| On Error GoTo ERRShowPoints3
3515|
3516| DummyShort = SetTextAlign(Me.picMap.hDC, TA_BASELINE)
3517| """Me.picMap.FontName = frmPointOptionsSimple.cmbFontList 'New 940701
3518| nPoints = LastPoint - FirstPoint + 1
3519|
3520| Me!pnl3dPercentDone.FloodPercent = 0
3521| Me!lblPercentDone = "Drawing symbols from " & Themes(nTheme).ThemeName
3522|  SetFontInPIC -1, Me.picMap
3523| Dim nStart As Long
3524| For n = FirstPoint To LastPoint
3525|    flgSkipIt = False
```

```
3526|   DoEvents
3527|   If flgStopDrawing Then Exit Sub
3528|    nStart = Points(n).nStart
3529|    Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pt, nStart, lngOne)
3530| 'Check whether the point falls in the viewbounds.
3531|
3532|    If Pt.X < ViewBoundsPlus.Left Then flgSkipIt = True
3533|    If Pt.X > ViewBoundsPlus.Right Then flgSkipIt = True
3534|    If Pt.Y < ViewBoundsPlus.Bottom Then flgSkipIt = True
3535|    If Pt.Y > ViewBoundsPlus.Top Then flgSkipIt = True
3536|    'SetFontCodeInPICxxx Points(N).ZFontCode, Me.picMap  'New 940701
3537|    picMap.FontSize = Points(n).ZSize
3538|    SetFontInPIC Points(n).ZFontCode, Me.picMap
3539|    If flgSkipIt Then GoTo BotShowPoints3
3540|
3541|    If flgSkipIt Then GoTo BotShowPoints3
3542|    picMap.ForeColor = Points(n).ZColor
3543|    'Me.Print "#" "This line slows program down, but does
3544|    '          allow use of color on the Zenith 486 in 401.
3545|
3546|    If flgSkipIt Then GoTo BotShowPoints3
3547|    picMap.CurrentX = Pt.X
3548|    '                If flgSkipIt Then GoTo BotShowPoints2
3549|    picMap.CurrentY = Pt.Y
3550|    '                If flgSkipIt Then GoTo BotShowPoints2
3551|    Me.picMap.Print Chr$(Points(n).ZSymbol)
3552|    '                If flgSkipIt Then GoTo BotShowPoints2
3553| BotShowPoints3:
3554|    Me.pnl3dPercentDone.FloodPercent = (n - FirstPoint) / nPoints * 100
3555| Next n
3556| RestoreDrawingSettings
3557| Me!pnl3dPercentDone.FloodPercent = 100
3558| Me!lblPercentDone = "Done drawing symbols from " & Themes(nTheme).ThemeName
3559| Me!pnl3dPercentDone.FloodPercent = 0: Me!lblPercentDone = ""
3560| Exit Sub
3561|
3562| ERRShowPoints3:
3563|  Select Case Err
3564|    Case 6
3565|     flgSkipIt = True
3566|     Resume Next
3567|    Case Else
3568|     Beep
3569|     MsgBox "Error #" & Err & " occurred in ShowPoints." & CRLF & Error
3570|     Err = 0
3571|     Exit Sub
3572|     Resume
3573|  End Select
3574|
3575|
3576|
3577| End Sub
3578|
3579| Sub ShowPolygons5 (FirstPoly As Long, LastPoly As Long, nTheme As Long)
```

```
3580| 'GWPouch 940613
3581| 'Displays solid color polygons, I hope.
3582| 'Derived from ShowLines3.
3583|
3584| 'This will check to see whether a polygon overlaps the view rectangle
3585| '  before plotting it.
3586| Dim n As Long, nPolys As Long, nStart As Long, nPoints As Long, nP2  As Long
3587| Dim flgSkipToBottom As Integer, iPoints As Integer
3588| On Error GoTo ERRShowPolygons5
3589|
3590| If (UBound(Polys) = 0) Then Exit Sub
3591|
3592| nPolys = LastPoly - FirstPoly + 1
3593| Me!pnl3dPercentDone.FloodPercent = 0
3594| Me!lblPercentDone = "Drawing polygons from " & Themes(nTheme).ThemeName
3595|
3596| 'SaveScaling
3597|  SaveScaling Me.picMap, OriginalScale
3598|  GetPixelBounds Me.picMap, PixelRect
3599|
3600|  Me.picMap.ScaleMode = 3 'Put into pixel mode
3601|  SetPolyFillMode Me.picMap.hDC, ALTERNATE 'Set fill mode to allow islands
3602|
3603| For n = FirstPoly To LastPoly
3604|  flgSkipToBottom = False
3605|  DoEvents
3606|  If flgStopDrawing Then GoTo LeaveShowPolygons5 'Graceful exit that restores scaling
3607|  'Test whether the polygon should be plotted
3608|        'XXXX LOGIC PROBLEM.  YOU CAN HAVE A LINE WHOSE BOUNDS ARE
      ABOVE
3609|        ' AND BELOW THE CURRENT VIEW WINDOW THAT STILL NEEDS TO BE
      DRAWN.
3610|  If Polys(n).MaxX < ViewBounds.Left Then flgSkipToBottom = True
3611|  If Polys(n).MinX > ViewBounds.Right Then flgSkipToBottom = True
3612|  If Polys(n).MinY > ViewBounds.Top Then flgSkipToBottom = True
3613|  If Polys(n).MaxY < ViewBounds.Bottom Then flgSkipToBottom = True
3614|  nPoints = Polys(n).nPoints: iPoints = nPoints: nP2 = 3 * nPoints
3615|  nStart = Polys(n).nStart
3616|  If nPoints < 3 Then flgSkipToBottom = True
3617| If flgSkipToBottom = False Then
3618|  ReDim PolyIn(1 To nPoints) As RealPoint, GDIPoly(1 To nP2) As APIPoint
3619|  Call GetRlPntsFromGlobalMem(hGMemAllPoints, PolyIn(1), nStart, nPoints)
3620|  'Call ScaleR4To2LOCAL(ViewBounds, PixelRect, nPoints, PolyIn(1), GDIPoly(1))
3621|  iPoints = SCScaleR4To2(ViewBounds, PixelRect, nPoints, PolyIn(1), nP2, GDIPoly(1))
3622|
3623| 'Draw optional polygon interior
3624|   If Polys(n).FillStyle <> 1 Then
3625|   Me.picMap.FillColor = Polys(n).FillColor
3626|   Me.picMap.FillStyle = Polys(n).FillStyle
3627|   Me.picMap.DrawStyle = 5 'Turn off outlining
3628|   If flgSkipToBottom = False Then
3629|     ReturnVal = Polygon(Me.picMap.hDC, GDIPoly(1), iPoints)
3630|     ""If ReturnVal = 0 Then Beep: dDebug.Print "Problem showing polygon #" & n & " in
      frmMap.picMap"
```

```
3631|    End If
3632|    End If
3633|
3634|  'Draw optional outline
3635|    If Polys(n).OutlineWidth > 0 Then
3636|    Me.picMap.DrawWidth = Polys(n).OutlineWidth
3637|    Me.picMap.ForeColor = Polys(n).OutlineColor
3638|    Me.picMap.DrawStyle = 6   'Inside solid outlining
3639|    If flgSkipToBottom = False Then
3640|       ReturnVal = Polyline(Me.picMap.hDC, GDIPoly(1), iPoints)
3641|       "'If ReturnVal = 0 Then Beep: dDebug.Print "Problem showing polygon #" & n & " in
      frmMap.picMap"
3642|    End If
3643|    End If
3644|
3645|  End If
3646|
3647|    Me.pnl3dPercentDone.FloodPercent = (n - FirstPoly) / nPolys * 100
3648|  Next n
3649|
3650|  LeaveShowPolygons5:
3651|   RestoreDrawingSettings
3652|   RestoreScaling Me.picMap, OriginalScale
3653|   Me.picMap.FillStyle = 1 'Turn off shape filling
3654|   Me.picMap.DrawStyle = 0  'Turn on outlining
3655|   Me!lblPercentDone = "Done Drawing": Me!pnl3dPercentDone.FloodPercent = 100
3656|   Me!pnl3dPercentDone.FloodPercent = 0: Me!lblPercentDone = ""
3657|  Exit Sub
3658|
3659|
3660|  ERRShowPolygons5:
3661|
3662|  flgSkipToBottom = True
3663|  Resume Next
3664|
3665|  End Sub
3666|
3667|  Sub ShowQSResults ()
3668|  If moShowLastQuery.Checked = False Then Exit Sub
3669|    Dim n As Long, nFeatures As Long
3670|    Dim lb As Long, ub As Long
3671|    On Error Resume Next
3672|    lb = LBound(QSDataOut)
3673|    ub = UBound(QSDataOut)
3674|    If LastQS.X >= ViewBounds.Left And LastQS.X <= ViewBounds.Right And LastQS.Y >=
      ViewBounds.Bottom And LastQS.Y <= ViewBounds.Top Then Call
      ZBeachBall(LastQS.X, LastQS.Y)
3675|
3676|    If QSTheme.ThemeType = ONEPOINT Then
3677|      Dim nPt As Long, nStart As Long, Pt As RealPoint, flgSkipIt As Integer
3678|      Dim tstrFontName As String
3679|
3680|      nPt = SetTextAlign(Me.picMap.hDC, TA_BASELINE)
3681|      Me.picMap.ForeColor = lngQSHighlightColor
```

```
3682|     For n = lb To ub
3683|    flgSkipIt = False
3684|    nPt = QSDataOut(n).IDNum
3685|    nStart = Points(nPt).nStart
3686|    Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pt, nStart, lngOne)
3687|    If Pt.X < ViewBounds.Left Then flgSkipIt = True
3688|    If Pt.X > ViewBounds.Right Then flgSkipIt = True
3689|    If Pt.Y < ViewBounds.Bottom Then flgSkipIt = True
3690|    If Pt.Y > ViewBounds.Top Then flgSkipIt = True
3691|    SetFontInPIC Points(nPt).ZFontCode, Me.picMap
3692|    tstrFontName = Me!picMap.FontName
3693|    If flgSkipIt Then GoTo BotShowPoints4
3694|    Me!picMap.FontSize = Points(nPt).ZSize
3695|    Me!picMap.FontName = tstrFontName
3696|    If flgSkipIt Then GoTo BotShowPoints4
3697|    Me.picMap.FontBold = True
3698|    Me!picMap.FontName = tstrFontName
3699|    If flgSkipIt Then GoTo BotShowPoints4
3700|    Me!picMap.CurrentX = Pt.X
3701|    Me!picMap.CurrentY = Pt.Y
3702|    Me!picMap.Print Chr$(Points(nPt).ZSymbol)
3703| BotShowPoints4:
3704|     Next
3705|   End If
3706|
3707|
3708|   'Draw circles around the points found
3709|   For n = lb To ub
3710|     Me.picMap.Circle (QSDataOut(n).X, QSDataOut(n).Y), sngQSHighlightRadius,
      lngQSHighlightColor
3711|   Next n
3712|
3713|
3714|
3715|   If lngSearchType = QS_RADIUSSEARCH Then
3716|    picMap.Circle (LastQS.X, LastQS.Y), QSDataOut(0).Dist1, lngQSHighlightColor'
      RGB(255, 0, 255)
3717|   End If
3718|
3719| Dim nLine As Long
3720| If QSTheme.ThemeType = JOINTEDLINE Then
3721|     nLine = QSDataOut(1).IDNum
3722|     Call DrawLine(Lines(nLine), Lines(nLine).ZWidth, lngQSHighlightColor,
      Lines(nLine).ZStyle)
3723| End If
3724|
3725|
3726|
3727| End Sub
3728|
3729| Sub ShowRasterGrid (myIndex As Long, Junk As Long, nTheme As Long)
3730| On Error GoTo ERRShowRasterGrid
3731| Dim nR As Long, nC As Long, PtX As Single, PtY As Single, ColorOut  As Long
3732| If RasterGrids(myIndex).DataType <> DB_LONG Then Exit Sub
```

```
3733| Me.picMap.DrawWidth = RasterGrids(myIndex).DeltaX / OnePixelInWorldCoordinates()
    + 1
3734| Dim nRows As Long
3735| Dim nFirstRow As Long
3736| nFirstRow = GridRowNumFromYCoord(ViewBounds.Bottom, RasterGrids(myIndex))
3737| nRows = GridRowNumFromYCoord(ViewBounds.Top, RasterGrids(myIndex)) -
    nFirstRow + 1
3738| Me.lblPercentDone = "Drawing rasters from " & Themes(nTheme).ThemeName
3739| For nR = GridRowNumFromYCoord(ViewBounds.Bottom, RasterGrids(myIndex)) To
    GridRowNumFromYCoord(ViewBounds.Top, RasterGrids(myIndex))
3740|   PtY = RasterGrids(myIndex).Y_Origin + nR * RasterGrids(myIndex).DeltaY
3741|   Me!pnl3dPercentDone.FloodPercent = 100 * (nR - nFirstRow) / nRows
3742|   For nC = GridColNumFromXCoord(ViewBounds.Left, RasterGrids(myIndex)) To
    GridColNumFromXCoord(ViewBounds.Right, RasterGrids(myIndex))
3743|     DoEvents
3744|     If flgStopDrawing Then GoTo STOPShowRasterGrid
3745|     PtX = RasterGrids(myIndex).X_Origin + nC * RasterGrids(myIndex).DeltaX
3746|     ColorOut = GetIJCellINT4FromGlobalMem(nC, nR, RasterGrids(myIndex),
    RasterGrids(myIndex).hGMem)
3747|     ' ColorOut = GetXYCellINT4FromGlobalMem(PtX, PtY, RasterGrids(myIndex),
    RasterGrids(myIndex).hGMem)
3748|     If ColorOut <> 0 Then Me.picMap.PSet (PtX, PtY), ColorOut
3749|   Next nC
3750| Next nR
3751| STOPShowRasterGrid:
3752| Me!lblPercentDone = "Done Drawing": Me!pnl3dPercentDone.FloodPercent = 100
3753| Me.picMap.DrawWidth = 1
3754| Exit Sub
3755| ERRShowRasterGrid:
3756| Select Case Err
3757|  Case Else
3758|  MsgBox "Erro #" & Err & " in ShowRasterGrid." & CRLF
3759|  Exit Sub
3760|  Resume
3761| End Select
3762|
3763| End Sub
3764|
3765| Sub ShowRasterValue (X As Single, Y As Single, RGNumber As Long)
3766| Dim CellValue As Variant
3767|   CellValue = GetXYCellValueGenericFromGlobalMem(X, Y, RasterGrids(RGNumber),
    RasterGrids(RGNumber).hGMem)
3768|   'Show CellValue somewhere
3769|   Me!pnlQSResults = CellValue & " <= " & RasterGrids(RGNumber).LayerName
3770| End Sub
3771|
3772| Sub ShowThemes ()
3773|   If Me.WindowState = 1 Then Exit Sub   'Leave if iconized
3774| Dim nTh As Long, mm As Long
3775|
3776| If nThemesShownHere = 0 Then
3777|    'zzShowDisclaimer
3778|    Exit Sub
3779| End If
```

```
3780| If flgSkipRepainting Then Exit Sub
3781| If flgStopDrawing Then Exit Sub
3782|
3783| If flgLoadDIB Then
3784|   flgSkipRepainting = True
3785|   picMap.Picture = LoadPicture(FrmTempName & ".DIB")
3786|   flgLoadDIB = False
3787|   flgSkipRepainting = False
3788| End If
3789|
3790| picMap.Cls
3791| 'Make sure the scale is right before trying anything else
3792| Me.picMap.Scale (ViewBounds.Left, ViewBounds.Top)-(ViewBounds.Right,
      ViewBounds.Bottom)
3793| picMap.Line (ViewBounds.Left, ViewBounds.Top)-(ViewBounds.Right,
      ViewBounds.Bottom), MapBaseColor, BF
3794|
3795|  'Me!lblScaleWords = "Scale 1:" & Format$(ScaleFactor * 1000, "#,###")
3796|  Me!lblScaleWords = "Scale 1:" & MapScaleInWords(Me!picMap.ScaleWidth,
      Me!picMap.Width)
3797|
3798| For mm = 1 To nThemesShownHere
3799|    If Not flgSkipRepainting Then DoEvents
3800|    If flgStopDrawing Then GoTo ExitShowThemes
3801|  If ThemeOrder(mm) > 0 Then
3802|    'If moLayerName(mm).Checked Then
3803|      nTh = Abs(ThemeOrder(mm))
3804|    Select Case Themes(nTh).ThemeType
3805|   Case JOINTEDLINE:  Call ShowLines5(Themes(nTh).FirstLoc, Themes(nTh).LastLoc,
      nTh)
3806|   Case ONEPOINT:    Call ShowPoints3(Themes(nTh).FirstLoc, Themes(nTh).LastLoc,
      nTh)
3807|   Case TILEBLOCK:   Call ShowTiles4(Themes(nTh).FirstLoc, Themes(nTh).LastLoc,
      nTh)
3808|   Case HATCHLINE:   Call ShowHatchLines3(Themes(nTh).FirstLoc,
      Themes(nTh).LastLoc, nTh)
3809|   Case SOLIDPOLYGON: Call ShowPolygons5(Themes(nTh).FirstLoc,
      Themes(nTh).LastLoc, nTh)
3810|   Case LABELTEXT:   Call ShowLabels1(Themes(nTh).FirstLoc,
      Themes(nTh).LastLoc, nTh)
3811|   Case RASTERGRID:  Call ShowRasterGrid(Themes(nTh).FirstLoc,
      Themes(nTh).LastLoc, nTh)
3812|   Case Else: Beep: MsgBox "Unidentified theme type. " & Themes(nTh).ThemeName
3813|    End Select
3814|  End If
3815| Next mm
3816|
3817| ExitShowThemes:
3818| RestoreDrawingSettings
3819| ShowQSResults 'Display the latest query results
3820|  If flgDrawGraticule Then
3821|     Call DrawGraticule(Me!picMap, GraticuleSize, GraticuleColor, GraticuleThickness)
3822|     Me!mvGraticule.Caption = "Map Graticule " & Format$(GraticuleSize, "#,###.#") & " "
      & CoordinateUnits()
```

```
3823|   End If
3824|
3825|   Call RestoreDrawingSettings
3826|   If iDrawScaleBar Then Call DrawScaleBar(Me!picMap, iDrawScaleBar)
3827|
3828|
3829|  Exit Sub
3830|
3831|  End Sub
3832|
3833|  Sub ShowTiles3 (FirstTile As Long, LastTile As Long, nTheme As Long)
3834|
3835|  ReDim OldTiles(20) As TileDeclPlain
3836|  Dim TileSizeX As Single, TileSizeY As Single
3837|
3838|  Dim LLeft As Single, RRight As Single, TTop As Single, BBottom As Single
3839|  Dim flgSkipPlotting As Integer
3840|
3841|  Dim n As Long, Pt As RealPoint, L As Long, One As Long
3842|  pnl3dPercentDone.FloodPercent = 0
3843|  lblPercentDone = "Drawing OldTiles from " & Themes(nTheme).ThemeName
3844|  nTiles = LastTile - FirstTile + 1
3845|
3846|  For n = FirstTile To LastTile
3847|   DoEvents
3848|   If flgStopDrawing Then Exit Sub
3849|    L = OldTiles(n).nStart
3850|    Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pt, L, lngOne)
3851|
3852|  'Before trying to plot the tile, check to see whether it falls in
3853|  '  the current viewing rectangle.
3854|    LLeft = Pt.X - TileSizeX
3855|    If LLeft > ViewBounds.Right Then GoTo BottomTilesLoop3
3856|    RRight = Pt.X + TileSizeX
3857|    If RRight < ViewBounds.Left Then GoTo BottomTilesLoop3
3858|    TTop = Pt.Y + TileSizeY
3859|    If TTop < ViewBounds.Bottom Then GoTo BottomTilesLoop3
3860|    BBottom = Pt.Y - TileSizeY
3861|    If BBottom > ViewBounds.Top Then GoTo BottomTilesLoop3
3862|    'Note that goto s are used rather than a flag variable because this
3863|    '  will often let us skip some computations.
3864|    picMap.Line (Pt.X - TileSizeX, Pt.Y - TileSizeY)-(Pt.X + TileSizeX, Pt.Y + TileSizeY),
       OldTiles(n).ZColor, BF
3865|  BottomTilesLoop3:
3866|    Me.pnl3dPercentDone.FloodPercent = (n - FirstTile) / nTiles * 100
3867|  Next n
3868|  RestoreDrawingSettings
3869|  pnl3dPercentDone.FloodPercent = 100
3870|  lblPercentDone = "Done drawing OldTiles from " & Themes(nTheme).ThemeName
3871|
3872|
3873|  End Sub
3874|
3875|  Sub ShowTiles4 (FirstTile As Long, LastTile As Long, nTheme As Long)
```

```
3876I 'GWPouch 940720 Converted to use GDI Rectangle, and full outline tiles.
3877I
3878I Dim flgSkipPlotting As Integer
3879I ReDim Pts(1 To 2) As RealPoint, GDIPts(1 To 2) As APIPoint
3880I Dim n As Long, L  As Long, One As Long, Two  As Long, dummyInt As Integer
3881I Dim GDIBounds As APIRECT
3882I One = 1&: Two = 2&
3883I
3884I Me.pnl3dPercentDone.FloodPercent = 0
3885I Me.lblPercentDone = "Drawing tiles from " & Themes(nTheme).ThemeName
3886I nTiles = LastTile - FirstTile + 1
3887I
3888I  SaveScaling Me.picMap, OriginalScale
3889I  GetPixelBounds Me.picMap, PixelRect
3890I  Me.picMap.ScaleMode = 3 'Put into pixel mode
3891I
3892I For n = FirstTile To LastTile
3893I  flgSkipPlotting = False
3894I  DoEvents
3895I  If flgStopDrawing Then GoTo STOPShowTiles4
3896I   L = Tiles(n).nStart
3897I   Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pts(1), L, Two)
3898I
3899I 'Before trying to plot the tile, check to see whether it falls in
3900I '  the current viewing rectangle.
3901I
3902I   If Pts(1).X > ViewBounds.Right Then flgSkipPlotting = True
3903I   If Pts(2).X < ViewBounds.Left Then flgSkipPlotting = True
3904I   If Pts(1).Y < ViewBounds.Bottom Then flgSkipPlotting = True
3905I   If Pts(2).Y > ViewBounds.Top Then flgSkipPlotting = True
3906I   Me.picMap.FillColor = Tiles(n).InteriorColor
3907I   Me.picMap.ForeColor = Tiles(n).OutlineColor
3908I   Me.picMap.FillStyle = Tiles(n).InteriorFillStyle
3909I   Me.picMap.DrawStyle = Tiles(n).OutlineStyle
3910I   If Tiles(n).OutlineWidth > 0 Then
3911I   Me.picMap.DrawWidth = Tiles(n).OutlineWidth
3912I     Else
3913I   Me.picMap.DrawStyle = 5
3914I   End If
3915I   'Note that goto s are used rather than a flag variable because this
3916I   '  will often let us skip some computations.
3917I   'Me.picMap.Line (Pt.X - TileSizeX, Pt.Y - TileSizeY)-(Pt.X + TileSizeX, Pt.Y +
      TileSizeY), Tiles(n).ZColor, BF
3918I   If Not flgSkipPlotting Then
3919I   Call ScaleR4To12LOCAL(ViewBounds, PixelRect, Two, Pts(1), GDIPts(1))
3920I   dummyInt = Rectangle(Me.picMap.hDC, GDIPts(1).X, GDIPts(1).Y, GDIPts(2).X,
      GDIPts(2).Y)
3921I
3922I   End If
3923I
3924I   Me.pnl3dPercentDone.FloodPercent = (n - FirstTile) / nTiles * 100
3925I Next n
3926I
3927I STOPShowTiles4:
```

```
3928|
3929|  RestoreDrawingSettings
3930|  RestoreScaling Me.picMap, OriginalScale
3931|  Me!lblPercentDone = "Done Drawing": Me!pnl3dPercentDone.FloodPercent = 100
3932|  Me!pnl3dPercentDone.FloodPercent = 0: Me!lblPercentDone = ""
3933|  Exit Sub
3934|
3935|  ERRPrintTiles4:
3936|  flgSkipPlotting = True
3937|  Resume Next
3938|
3939|
3940|
3941|  End Sub
3942|
3943|  Sub WMF_QSResults ()
3944|    Dim n As Long, nFeatures As Long
3945|    Dim lb As Long, ub As Long
3946|    On Error Resume Next
3947|    lb = LBound(QSDataOut)
3948|    ub = UBound(QSDataOut)
3949|
3950|    If LastQS.X >= ViewBounds.Left And LastQS.X <= ViewBounds.Right And LastQS.Y >=
       ViewBounds.Bottom And LastQS.Y <= ViewBounds.Top Then
3951|     Call ZWMFBeachBall(LastQS.X, LastQS.Y, lngQSHighlightColor)
3952|        WMFDrawHollowCircle QSDataOut(n).X, QSDataOut(n).Y, sngQSHighlightRadius,
       Not lngQSHighlightColor
3953|     End If
3954|
3955|    If QSTheme.ThemeType = ONEPOINT Then
3956|      Dim nPt As Long, nStart As Long, Pt As RealPoint, flgSkipIt As Integer
3957|       Dim iX As Integer, jY As Integer, strOut As String
3958|      SetTextColorInWMF lngQSHighlightColor
3959|      For n = lb To ub
3960|     flgSkipIt = False
3961|     nPt = QSDataOut(n).IDNum
3962|     nStart = Points(nPt).nStart
3963|     Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pt, nStart, lngOne)
3964|     If Pt.X < ViewBounds.Left Then flgSkipIt = True
3965|     If Pt.X > ViewBounds.Right Then flgSkipIt = True
3966|     If Pt.Y < ViewBounds.Bottom Then flgSkipIt = True
3967|     If Pt.Y > ViewBounds.Top Then flgSkipIt = True
3968|     If flgSkipIt = False Then Call WMFPlotSymbol(Pt.X, Pt.Y, Points(nPt).ZFontCode Or
       FONT_BOLD, Points(nPt).ZSize, Points(nPt).ZSymbol)
3969|  BotShowPoints6:
3970|      Next
3971|    End If
3972|
3973|
3974|
3975|
3976|
3977|    If lngSearchType = QS_RADIUSSEARCH Then
3978|     WMFDrawHollowCircle LastQS.X, LastQS.Y, QSDataOut(0).Dist1,
```

```
        lngQSHighlightColor
3979|   End If
3980|
3981|   'Draw circles around the points found
3982|   For n = lb To ub
3983|     WMFDrawHollowCircle QSDataOut(n).X, QSDataOut(n).Y, sngQSHighlightRadius,
        lngQSHighlightColor
3984|   Next n
3985|
3986|  End Sub
3987|
3988|  Sub WriteMetaFileOut (FileOut As String)
3989|  MousePointer = 11
3990|  'Prepare metafile to be HI_METRIC, centered on zero.
3991|   ViewPortRect.Left = -50# * picMap.Width
3992|   ViewPortRect.Right = 50# * picMap.Width
3993|   ViewPortRect.Top = 50# * picMap.Height
3994|   ViewPortRect.Bottom = -50# * picMap.Height
3995|
3996|   Call StartWMF(ViewBounds, ViewPortRect, FileOut)
3997|
3998|  Dim mm As Long, nTh As Long, hMFReturned As Integer
3999|  For mm = 1 To nThemesShownHere
4000|   If ThemeOrder(mm) > 0 Then
4001|      'If moLayerName(mm).Checked Then
4002|        nTh = Abs(ThemeOrder(mm))
4003|     Select Case Themes(nTh).ThemeType
4004|     Case JOINTEDLINE:  Call WMFLines5(Themes(nTh).FirstLoc, Themes(nTh).LastLoc,
        nTh)
4005|     Case ONEPOINT:     Call WMFPoints3(Themes(nTh).FirstLoc, Themes(nTh).LastLoc,
        nTh)
4006|     Case TILEBLOCK:    Call WMFTiles4(Themes(nTh).FirstLoc, Themes(nTh).LastLoc,
        nTh)
4007|     Case HATCHLINE:    'Call ShowHatchLines3(Themes(nTh).FirstLoc,
        Themes(nTh).LastLoc, nTh)
4008|     Case SOLIDPOLYGON: Call WMFPolygons5(Themes(nTh).FirstLoc,
        Themes(nTh).LastLoc, nTh)
4009|     Case LABELTEXT:    Call WMFLabels1(Themes(nTh).FirstLoc,
        Themes(nTh).LastLoc, nTh)
4010|     Case RASTERGRID:   Call WMFRasterGrid(Themes(nTh).FirstLoc,
        Themes(nTh).LastLoc, nTh)
4011|     Case Else: Beep: MsgBox "Unidentified theme type. " & Themes(nTh).ThemeName
4012|      End Select
4013|   End If
4014|  Next mm
4015|
4016|  If flgDrawGraticule Then Call WMFGraticule(GraticuleColor, GraticuleThickness)
4017|
4018|  If iDrawScaleBar Then Call WMFScaleBar(iDrawScaleBar)
4019|
4020|   If moShowLastQuery.Checked Then WMF_QSResults
4021|   Call StopWMF(hMFReturned)
4022|   If FileOut <> "" Then
4023|     'If written to a disk file, make it a placeable metafile
```

```
4024|     Call WritePlaceableWMFHeader_HIMETRIC(FileOut)
4025|    Else
4026|     'Otherwise, send it to the clipboard.
4027|     Call CopyWMFToClipboard(hMFReturned)
4028|   End If
4029|
4030|   MousePointer = 0
4031|  End Sub
4032|
4033|  Sub XScroll_Change ()
4034|  If flgDontMove Then Exit Sub
4035|  NewScrollX = XScroll
4036|  Dim DeltaX As Single
4037|  DeltaX = (NewScrollX - LastScrollX) / 50#
4038|  'The above line is a
       Throw_the_goat's_liver_over_your_shoulder_'cause_it_worked_last_time_solution.
4039|  'Call ZZShiftX(DeltaX * CurrentSize.X)
4040|  Call ZZShiftX(DeltaX * DataRange.X)
4041|  If moAutoDrawOnViewShift.Checked Then Call ShowThemes
4042|
4043|  ' Discarded 940513 GWPouch
4044|  'Exit Sub
4045|  '  If NewScrollX > LastScrollX Then
4046|  '    LastScrollX = NewScrollX
4047|  '    mnuVRight_Click
4048|  '  End If
4049|  '
4050|  '  If NewScrollX < LastScrollX Then
4051|  '    LastScrollX = NewScrollX
4052|  '    mnuVLeft_Click
4053|  '  End If
4054|
4055|
4056|  End Sub
4057|
4058|  Sub XScroll_DragDrop (Source As Control, X As Single, Y As Single)
4059|
4060|  Dim ShiftLeft As Single
4061|  'ViewBounds.Left = DataBounds.Left + NewScrollX / 200# * DataRange.X
4062|  'Call ZZoom(ViewBounds.Left, ViewBounds.Top, 1#)
4063|  Dim ShiftX As Single
4064|  ShiftX = (NewScrollX - 100) / 200# * CurrentSize.X
4065|
4066|  Call ZZShiftX(ShiftX)
4067|
4068|  LastScrollX = NewScrollX
4069|
4070|
4071|
4072|
4073|
4074|
4075|
4076|  End Sub
```

```
4077|
4078| Sub YScroll_Change ()
4079| If flgDontMove Then Exit Sub
4080| NewScrollY = YScroll
4081| Dim DeltaY As Single
4082| DeltaY = (NewScrollY - LastScrollY) / 50#
4083| 'The above line is a
     Throw_the_goat's_liver_over_your_shoulder_'cause_it_worked_last_time_solution.
4084|
4085| 'Call ZZShiftY(DeltaY * CurrentSize.Y)
4086| Call ZZShiftY(DeltaY * DataRange.Y)
4087| If moAutoDrawOnViewShift.Checked Then Call ShowThemes
4088|
4089| Exit Sub
4090|
4091|
4092|
4093|
4094|
4095|
4096|
4097|
4098|
4099| NewScrollY = YScroll
4100|
4101| If flgDontMove Then
4102|  'do nothing
4103|  Else
4104|  If NewScrollY > LastScrollY Then
4105|    LastScrollY = NewScrollY
4106|    mnuVUp_Click
4107|  End If
4108|
4109|  If NewScrollY < LastScrollY Then
4110|    LastScrollY = NewScrollY
4111|    mnuVDown_Click
4112|  End If
4113|
4114| End If
4115|
4116|
4117| End Sub
4118|
4119| Sub ZAskUserForThemesVisible ()
4120|  frmThemesVisible.chkThemeVisible(0).Visible = False
4121|  frmThemesVisible.picHolder.Height = 7.5 * (nThemesShownHere + 3)
4122|  frmThemesVisible.VScroll.Max = frmThemesVisible.picHolder.Height
4123|  'This does not list themes which were discarded using options-theme-order
4124|    For n = 1 To nThemesShownHere
4125|   M = Abs(ThemeOrder(n))
4126|   Load frmThemesVisible.chkThemeVisible(n)
4127|   frmThemesVisible.chkThemeVisible(n).Top = 7.5 * n
4128|   If ThemeOrder(n) > 0 Then
4129|      frmThemesVisible.chkThemeVisible(n).Value = 1
```

```
4130|       Else
4131|        frmThemesVisible.chkThemeVisible(n).Value = 0
4132|     End If
4133|     frmThemesVisible.chkThemeVisible(n).Visible = True
4134|     frmThemesVisible.chkThemeVisible(n).Caption = Themes(M).ThemeName
4135|     'frmThemeOrder.lstThemesToShow.AddItem Themes(M).ThemeName & Format(M, "
      000")
4136|     Next n
4137|
4138|   frmThemesVisible.Show 1
4139|
4140|   If frmThemesVisible.cmdOK.FontItalic Then
4141|     For n = nThemesShownHere To 1 Step -1
4142|     If frmThemesVisible.chkThemeVisible(n).Value = 1 Then
4143|        ThemeOrder(n) = Abs(ThemeOrder(n))
4144|       Else
4145|        ThemeOrder(n) = -Abs(ThemeOrder(n))
4146|     End If
4147|     Unload frmThemesVisible.chkThemeVisible(n)
4148|     Next n
4149|     ShowThemes
4150|   End If
4151|
4152|   Unload frmThemesVisible
4153|   End Sub
4154|
4155|   Sub ZBeachBall (X As Single, Y As Single)
4156|       On Error Resume Next
4157|     Dim DDrawMode As Integer, DDrawStyle As Integer, PhilColor As Long, PhilStyle As
      Integer
4158|     'Save current drawing setup
4159|     PhilColor = picMap.FillColor
4160|     PhilStyle = picMap.FillStyle
4161|     DDrawMode = picMap.DrawMode
4162|     DDrawStyle = picMap.DrawStyle
4163|      'Set for inverse, solid color drawing
4164|     Me!picMap.DrawStyle = 0 'Was 6 for inside solid, made it 0 for solid
4165|     'Me!picMap.DrawMode = 7
4166|     Me!picMap.FillStyle = 0
4167|     Me!picMap.FillColor = lngQSHighlightColor' RGB(255, 255, 255)
4168|       'Make a beachball
4169|       'lngQSHighlightColor
4170|     'Me!picMap.Circle (X, Y), CurrentSize.X * .075, RGB(255, 255, 255), -.00000000001,
      -1.570796326795
4171|     'Me!picMap.Circle (X, Y), CurrentSize.X * .075, RGB(255, 255, 255), -3.14159265359,
      -4.712388980385
4172|     Me!picMap.Circle (X, Y), CurrentSize.X * .075, lngQSHighlightColor, -.00000000001,
      -1.570796326795
4173|     Me!picMap.Circle (X, Y), CurrentSize.X * .075, lngQSHighlightColor, -3.14159265359,
      -4.712388980385
4174|
4175|       'restore previous drawing setup
4176|     Me!picMap.FillColor = PhilColor
4177|     Me!picMap.FillStyle = PhilStyle
```

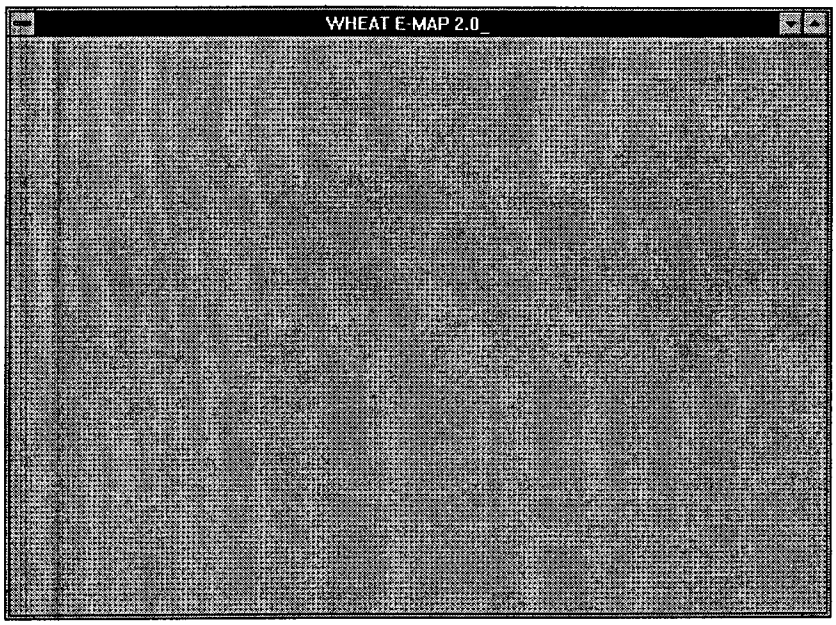
```
4178|    Me!picMap.DrawMode = DDrawMode
4179|    Me!picMap.DrawStyle = DDrawStyle
4180|
4181|  End Sub
4182|
4183|  Sub ZLoadThemeToMenu (NewTheme As typTheme)
4184|  'Add it to the theme show order
4185|     nThemesShownHere = nThemesShownHere + 1
4186|     ReDim Preserve ThemeOrder(nThemesShownHere)
4187|     ReDim Preserve ThemeVisible(nThemesShownHere)
4188|     ThemeOrder(nThemesShownHere) = NewTheme.myIndex
4189|
4190|
4191|
4192|
4193|
4194|
4195|
4196|  'Add it the theme menu
4197|     'Load moLayerName(nThemesShownHere)
4198|     'moLayerName(nThemesShownHere).Visible = True
4199|     'moLayerName(nThemesShownHere).Caption = "&" &
      Format$(nThemesShownHere) & " " & NewTheme.ThemeName
4200|     'moLayerName(nThemesShownHere).Checked = True
4201|     'moLayerName(nThemesShownHere).Enabled = True
4202|     '
4203|  End Sub
4204|
4205|  Sub zMyAddDataChangeBounds (x1new As Single, y1new As Single, x2new As Single,
      y2new As Single)
4206|  'This subroutine changes DataBounds in response to the addition
4207|  ' of new data.  Whatever subroutine reads the new data calls
4208|  Call AddDataChangeBounds(x1new, y1new, x2new, y2new)
4209|  'It also resets the DataAspect ratio, which is the ratio
4210|  ' of the range in x to the range in y
4211|  'Finally, it re-adjusts the MaxScaleFactor to fit all the data
4212|  '  onto the Map when it is zoomed out all the way.
4213|  Dim a As Single, B As Single
4214|  a = DataRange.X / (Me.picMap.Width - 1)
4215|  B = DataRange.Y / (Me.picMap.Height - 1)
4216|
4217|  If a > B Then
4218|     MaxScaleFactor = a
4219|    Else
4220|     MaxScaleFactor = B
4221|  End If
4222|
4223|  End Sub
4224|
4225|  Sub ZoomByRectangle (XMin As Single, XMax As Single, YMin As Single, YMax As
      Single)
4226|  Dim ZoomFactor As Single
4227|  Dim Z2 As Single
4228|     If XMax = XMin Then Exit Sub
```

```
4229|    If YMax = YMin Then Exit Sub
4230|
4231|   ZoomFactor = (XMax - XMin) / (CurrentSize.X * 2#)
4232|   Z2 = (YMax - YMin) / (CurrentSize.Y * 2#)
4233| If ZoomFactor < Z2 Then ZoomFactor = Z2
4234| Call ZZoom(XMin, YMax, ZoomFactor)
4235|
4236|  'xxxxxxx
4237| End Sub
4238|
4239| Sub ZoomWayOut ()
4240| On Error GoTo ERRZoomWayOut
4241| Dim ZoomFactor As Single, NewLeft As Single, NewTop As Single
4242|  ZoomFactor = MaxScaleFactor / ScaleFactor
4243|  NewLeft = DataBounds.Left
4244|  NewTop = DataBounds.Top
4245| Call ZZoom(NewLeft, NewTop, ZoomFactor)
4246| Call cmdRefreshMap_Click
4247| Exit Sub
4248|
4249| ERRZoomWayOut:
4250| Err = 0
4251| Call AdjustMapScaleView(DataBounds)
4252| Call cmdRefreshMap_Click
4253| Exit Sub
4254| Resume
4255|
4256|
4257| End Sub
4258|
4259| Sub ZQDrawBox (X1 As Single, Y1 As Single, X2 As Single, Y2 As Single, Color As
      Long)
4260| 'This subroutine is used for drawing/undrawing a zoom or selection box on picMap.
4261|    Dim OldDrawMode As Integer
4262|    OldDrawMode = picMap.DrawMode
4263|    picMap.DrawMode = 7'Inverse
4264|    picMap.Line (X1, Y1)-(X2, Y2), Color, B
4265|    picMap.DrawMode = OldDrawMode
4266| End Sub
4267|
4268| Sub ZUnloadThemeList ()
4269|   For n = nThemesShownHere To 1 Step -1
4270|     'Unload moLayerName(n)
4271|   Next n
4272|   nThemesShownHere = 0
4273| End Sub
4274|
4275| Sub ZZAdjustMapScale ()
4276| On Error GoTo ERRAdjustMapScale
4277|
4278|  ViewBounds.Right = ViewBounds.Left + picMap.Width * ScaleFactor
4279|  ViewBounds.Bottom = ViewBounds.Top - picMap.Height * ScaleFactor
4280|  CurrentSize.Y = .5 * (ViewBounds.Top - ViewBounds.Bottom)
4281|  CurrentSize.X = .5 * (ViewBounds.Right - ViewBounds.Left)
```

```
4282  Me!picMap.Scale (ViewBounds.Left, ViewBounds.Top)-(ViewBounds.Right,
      ViewBounds.Bottom)
4283
4284
4285
4286
4287  'Adjust Scroll Bar positions.
4288  flgDontMove = True 'Turn off scrolling by ScrollBars
4289  DummyReal = -100 * (DataBounds.Left - ViewBounds.Left) / DataRange.X
4290
4291  If DummyReal > 100 Then DummyReal = 100
4292  If DummyReal < -100 Then DummyReal = -100
4293  XScroll = DummyReal + 100
4294  LastScrollX = XScroll
4295  DummyReal = -100 * (DataBounds.Top - ViewBounds.Top) / DataRange.Y
4296  If DummyReal > 100 Then DummyReal = 100
4297  If DummyReal < -100 Then DummyReal = -100
4298  YScroll = DummyReal + 100
4299  LastScrollY = YScroll
4300  flgDontMove = False
4301  ' Re-enable scrolling with scroll bars
4302
4303  "'Find out how big a pixel is in world coordinates, so we
4304  "' can move a specified distance.
4305  "Dim Before As Single, After As Single
4306  "Before = MouseX
4307  "Call ShiftCursor(1, 0)
4308  "After = MouseX
4309  "OnePixelInWorldCoordinates = After - Before
4310  "'This works for both X and Y due the incredible lengths gone
4311  "' to to keep the display isotropic.
4312  Exit Sub
4313  ERRAdjustMapScale:
4314  Err = 0
4315  Call AdjustMapScaleView(DataBounds)
4316  Call cmdRefreshMap_Click
4317  Exit Sub
4318  Resume
4319
4320  End Sub
4321
4322  Sub zzFile_Click ()
4323    'Call cmdStopDrawing_Click
4324  End Sub
4325
4326  Sub ZZoom (NewLeft As Single, NewTop As Single, ZoomFactor As Single)
4327  'Call ZZoom(NewLeft, NewTop, ZoomFactor)
4328  ScaleFactor = ScaleFactor * ZoomFactor
4329  ViewBounds.Left = NewLeft
4330  ViewBounds.Top = NewTop
4331  ZZAdjustMapScale
4332  End Sub
4333
4334  Sub ZZShiftX (XShift As Single)
```

```
 4335|  ViewBounds.Left = ViewBounds.Left + XShift
 4336|
 4337|  ZZAdjustMapScale
 4338|
 4339|
 4340|  End Sub
 4341|
 4342|  Sub ZZShiftY (YShift As Single)
 4343|   ViewBounds.Top = ViewBounds.Top + YShift
 4344|   ZZAdjustMapScale
 4345|  End Sub
 4346|
 4347|  Sub zzShowDisclaimer ()
 4348|  Me.picMap.Cls
 4349|  Me!picMap.CurrentX = Me!picMap.ScaleLeft
 4350|  Me!picMap.CurrentY = Me!picMap.ScaleTop
 4351|  Me!picMap.FontName = "MS Sans Serif"
 4352|  Dim OldAlignment As Long
 4353|  'OldAlignment = SetTextAlign(Me.picMap.hDC, 1&)
 4354|  Me!picMap.FontSize = 8.5
 4355|   Dim strStartMessage As String
 4356|      strStartMessage = ""
 4357|      strStartMessage = strProgName & strVersion & " is the Electronic MAP part of the
       WHEAT, the " & CRLF
 4358|      strStartMessage = strStartMessage & "Windows-based, Hydrogeologic Exploration
       and Analysis Toolkit." & CRLF
 4359|      strStartMessage = strStartMessage & "This set of programs was developed by Greg
       Pouch " & CRLF
 4360|      strStartMessage = strStartMessage & "at the Kanas Geological Survey, Geohydrology
       Section," & CRLF
 4361|      strStartMessage = strStartMessage & "under contract to the "
 4362|      strStartMessage = strStartMessage & "Equus Beds Groundwater Management
       District, " & CRLF & "Halstead, Kansas.  (KGS OFR 94-51)" & CRLF & CRLF
 4363|      strStartMessage = strStartMessage & "E-MAP is provided free of charge and without
       any warranty" & CRLF & "of any kind.  "
 4364|      strStartMessage = strStartMessage & "Free technical support for using or adding
       features " & CRLF & "to this program or for designing databases is not available." &
       CRLF
 4365|
 4366|      strStartMessage = strStartMessage & CRLF & "DISCLAIMER" & CRLF & "Neither the
       Equus Beds Groundwater Management District nor" & CRLF
 4367|      strStartMessage = strStartMessage & "the Kansas Geological Survey guarantees this
       computer program" & CRLF
 4368|      strStartMessage = strStartMessage & "to be free from errors or inaccuracies and
       disclaims any" & CRLF
 4369|      strStartMessage = strStartMessage & "responsibility or liability for interpretations
       based on data " & CRLF
 4370|      strStartMessage = strStartMessage & "processed using this software or decisions
       based thereon." & CRLF & CRLF
 4371|                  '& CRLF
 4372|
 4373|    Dim SectionName As String, Buffer As String, KeyString As String, DefaultValue As
       String
 4374|     SectionName = "Disclaimer"
```

```
4375|    Buffer = Space$(1024)
4376|
4377|    Dim n As Long, nBytesBack As Long
4378|    Do
4379|      n = n + 1
4380|      KeyString = "Disclaimer" & n
4381|      DefaultValue = ""
4382|      nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
       1024, INIFile)
4383|      strStartMessage = strStartMessage & CRLF & Left$(Buffer, nBytesBack)
4384|    Loop While nBytesBack <> 0
4385|    "dDebug.Print strStartMessage
4386|    Me!picMap.Print strStartMessage
4387|
4388| End Sub
4389|
```


WHEAT E-MAP 2.0_

```
VERSION 2.00
Begin MDIForm MDI_WHEAT
  Caption       = "WHEAT E-MAP 2.0_"
  ClientHeight  = 6720
  ClientLeft    = 60
  ClientTop     = 390
  ClientWidth   = 9450
  Height        = 7125
  Icon          = MDIWHT01.FRX:0000
  Left          = 0
  LinkTopic     = "MDIForm1"
  ScrollBars    = 0  'False
  Top           = 45
  Width         = 9570
  Begin PictureBox picToolBar
    Align       = 1  'Align Top
    BackColor   = &H00C0C0C0&
    BorderStyle = 0  'None
    Height      = 15
    Left        = 0
    ScaleHeight = 15
    ScaleWidth  = 9450
    TabIndex    = 0
    Top         = 0
    Width       = 9450
  End
End
Option Explicit

Sub MDIForm_Load ()
 EMapStartup
 frmMap.Top = 0
 frmMap.Left = 0
 frmMap.Width = .95 * Me.Width
 frmMap.Height = .9 * Me.Height
 frmStatus.Show
 frmMap.Show
 frmMap.SetFocus


End Sub

Sub MDIForm_QueryUnload (Cancel As Integer, UnloadMode As Integer)

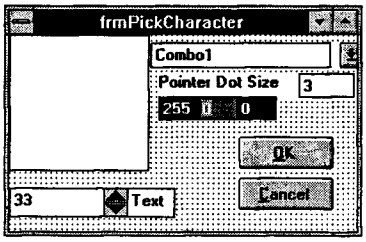
 ShutDownAndFreeMem

 Do While (GlobalUnlock(hGMemAllPoints)): Loop
 Do While (GlobalFree(hGMemAllPoints)): Loop


End Sub

Sub MDIForm_Unload (Cancel As Integer)
 ShutDownAndFreeMem
```

```
55|  Do While (GlobalUnlock(hGMemAllPoints)): Loop
56|  Do While (GlobalFree(hGMemAllPoints)): Loop
57|
58|  End Sub
59|
```

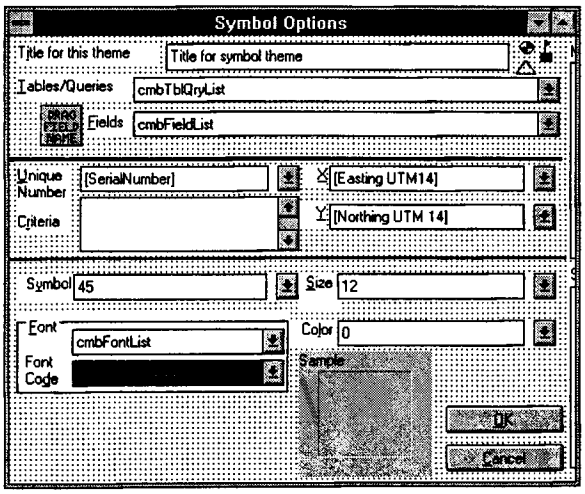

frmPickCharacter
Combo1
Pointer Dot Size
3
255
0
OK
Cancel
33
Text

```
VERSION 2.00
Begin Form frmPickCharacter
  Caption        = "frmPickCharacter"
  ClientHeight   = 2310
  ClientLeft     = 1020
  ClientTop      = 1905
  ClientWidth    = 4005
  ControlBox     = 0  'False
  Height         = 2715
  Left           = 960
  LinkTopic      = "Form1"
  ScaleHeight    = 2310
  ScaleWidth     = 4005
  Top            = 1560
  Width          = 4125
  Begin CommandButton cmdCancel
    Caption      = "&Cancel"
    Height       = 375
    Left         = 2640
    TabIndex     = 10
    Top          = 1680
    Width        = 1095
  End
  Begin CommandButton cmdOK
    Caption      = "&OK"
    Height       = 375
    Left         = 2640
    TabIndex     = 9
    Top          = 1200
    Width        = 1095
  End
  Begin TextBox txtCurrentSymbol
    Height       = 345
    Left         = 1350
    TabIndex     = 8
    Text         = "Text1"
    Top          = 1800
    Width        = 555
  End
  Begin TextBox txtBlue
    Alignment    = 1  'Right Justify
    BackColor    = &H00C00000&
    ForeColor    = &H00FFFFFF&
    Height       = 285
    Left         = 2610
    TabIndex     = 7
    Text         = "0"
    Top          = 720
    Width        = 465
  End
  Begin TextBox txtGreen
    Alignment    = 1  'Right Justify
    BackColor    = &H0000C000&
    ForeColor    = &H00FFFFFF&
```

```
   Height      = 285
   Left        = 2160
   TabIndex    = 6
   Text        = "0"
   Top         = 720
   Width       = 465
End
Begin TextBox txtRed
   Alignment   = 1 'Right Justify
   BackColor   = &H000000FF&
   ForeColor   = &H00FFFFFF&
   Height      = 285
   Left        = 1710
   TabIndex    = 5
   Text        = "255"
   Top         = 720
   Width       = 465
End
Begin TextBox txtDotSize
   Height      = 285
   Left        = 3330
   TabIndex    = 4
   Text        = "3"
   Top         = 450
   Width       = 645
End
Begin ComboBox cmbFontList
   Height      = 300
   Left        = 1620
   Sorted      = -1 'True
   TabIndex    = 2
   Text        = "Combo1"
   Top         = 90
   Width       = 2445
End
Begin SpinButton SpinChar
   Height      = 345
   Left        = 1080
   Top         = 1800
   Width       = 285
End
Begin TextBox txtCharNumber
   Height      = 345
   Left        = 0
   TabIndex    = 1
   Text        = "33"
   Top         = 1800
   Width       = 1125
End
Begin PictureBox picShowCharacter
   FontBold    = -1 'True
   FontItalic  = 0 'False
   FontName    = "MS Sans Serif"
   FontSize    = 48
```

```
109|      FontStrikethru = 0  'False
110|      FontUnderline  = 0  'False
111|      Height        = 1600
112|      Left          = 0
113|      ScaleHeight    = 1.095
114|      ScaleMode      = 0 'User
115|      ScaleWidth     = 1.094
116|      TabIndex       = 0
117|      Top           = 0
118|      Width         = 1600
119|    End
120|    Begin Label Label1
121|      Caption       = "Pointer Dot Size"
122|      Height        = 285
123|      Left          = 1710
124|      TabIndex      = 3
125|      Top           = 450
126|      Width         = 1635
127|    End
128| End
129| Option Explicit
130| Dim intSymbolCode As Integer
131| Dim DotSize As Integer
132|
133| Sub cmdCancel_Click ()
134| Me.Hide
135| End Sub
136|
137| Sub cmdOK_Click ()
138| Me.Hide
139| cmdOK.FontItalic = True
140| End Sub
141|
142| Sub Form_Paint ()
143|    cmdOK.FontItalic = False
144|    Dim strTemp As String
145|    strTemp = txtCharNumber
146|    txtCharNumber = "33"
147|    txtCharNumber = strTemp
148| End Sub
149|
150| Sub SpinChar_SpinDown ()
151|  intSymbolCode = Printable(intSymbolCode - 1)
152|  txtCharNumber = intSymbolCode
153|  txtCharNumber.SetFocus
154| End Sub
155|
156| Sub SpinChar_SpinUp ()
157|  intSymbolCode = Printable(intSymbolCode + 1)
158|  txtCharNumber = intSymbolCode
159|  txtCharNumber.SetFocus
160|
161| End Sub
162|
```

```
163| Sub txtCharNumber_Change ()
164|   picShowCharacter.Cls
165|     Dim DummyInt As Integer
166|   DummyInt = SetTextAlign(picShowCharacter.hDC, TA_BASELINE)
167|  picShowCharacter.CurrentX = .5
168|  picShowCharacter.CurrentY = .5
169|
170|   intSymbolCode = Val(txtCharNumber)
171|   If (intSymbolCode < 32) Then intSymbolCode = 32
172|   If (intSymbolCode > 255) Then intSymbolCode = 255
173|
174|   picShowCharacter.Print Chr$(intSymbolCode)
175|   If DotSize <> 0 Then
176|     picShowCharacter.DrawWidth = DotSize
177|    ' Red=Limit0255(val(txtRed))
178|     picShowCharacter.PSet (.5, .5), RGB(255, 0, 0)
179|   End If
180|   txtCurrentSymbol = Chr$(intSymbolCode)
181|
182| End Sub
183|
184| Sub txtDotSize_Change ()
185| DotSize = Val(txtDotSize)
186| If DotSize < 0 Then DotSize = 0
187| End Sub
188|
```


Symbol Options
Title for this theme
Title for symbol theme
Tables/Queries
cmbTblQryList
Fields
cmbFieldList
Unique Number
[SerialNumber]
[Easting UTM14]
Criteria
[Northing UTM 14]
Symbol
45
Size
12
Font
cmbFontList
Color
0
Font Code
Sample
OK
Cancel

```
VERSION 2.00
Begin Form frmPointOptionsSimple
  Caption       = "Symbol Options"
  ClientHeight  = 5205
  ClientLeft    = 135
  ClientTop     = 390
  ClientWidth   = 6540
  ControlBox    = 0  'False
  Height        = 5610
  Icon          = PNTOPT10.FRX:0000
  Left          = 75
  LinkTopic     = "Form1"
  MinButton     = 0  'False
  ScaleHeight   = 5205
  ScaleWidth    = 6540
  Top           = 45
  Width         = 6660
  Begin SSPanel zpnlOuter
    Alignment     = 0  'Left Justify - TOP
    AutoSize      = 3  'AutoSize Child To Panel
    BackColor     = &H00C0C0C0&
    BevelInner    = 1  'Inset
    BevelOuter    = 0  'None
    BorderWidth   = 15
    Font3D        = 0  'None
    ForeColor     = &H00FFFFFF&
    Height        = 1455
    Left          = 3330
    TabIndex      = 31
    Top           = 3690
    Width         = 1545
    Begin SSPanel zpnlSampleText
      Alignment     = 0  'Left Justify - TOP
      AutoSize      = 3  'AutoSize Child To Panel
      BackColor     = &H00C0C0C0&
      BevelOuter    = 0  'None
      BorderWidth   = 1
      Caption       = "Sample"
      Font3D        = 0  'None
      FontBold      = 0  'False
      FontItalic    = 0  'False
      FontName      = "MS Sans Serif"
      FontSize      = 8.25
      FontStrikethru = 0  'False
      FontUnderline = 0  'False
      ForeColor     = &H00000000&
      Height        = 220
      Left          = 0
      TabIndex      = 30
      Top           = 0
      Width         = 735
    End
    Begin PictureBox picSample
      BackColor     = &H00C0C0C0&
```

```
55|       BorderStyle    = 0  'None
56|       Height         = 980
57|       Left           = 240
58|       ScaleHeight    = 975
59|       ScaleWidth     = 1065
60|       TabIndex       = 32
61|       Top            = 240
62|       Width          = 1060
63|     End
64|   End
65|   Begin ComboBox cmb_zqFeatureID
66|     FontBold       = 0  'False
67|     FontItalic     = 0  'False
68|     FontName       = "MS Sans Serif"
69|     FontSize       = 8.25
70|     FontStrikethru = 0  'False
71|     FontUnderline  = 0  'False
72|     Height         = 300
73|     HelpContextID  = 1500
74|     Left           = 810
75|     TabIndex       = 8
76|     Text           = "[SerialNumber]"
77|     Top            = 1530
78|     Width          = 2535
79|   End
80|   Begin ComboBox cmb_zqY
81|     FontBold       = 0  'False
82|     FontItalic     = 0  'False
83|     FontName       = "MS Sans Serif"
84|     FontSize       = 8.25
85|     FontStrikethru = 0  'False
86|     FontUnderline  = 0  'False
87|     Height         = 300
88|     HelpContextID  = 1100
89|     Left           = 3690
90|     TabIndex       = 11
91|     Text           = "[Northing UTM 14]"
92|     Top            = 1980
93|     Width          = 2625
94|   End
95|   Begin ComboBox cmb_zqX
96|     FontBold       = 0  'False
97|     FontItalic     = 0  'False
98|     FontName       = "MS Sans Serif"
99|     FontSize       = 8.25
100|    FontStrikethru = 0  'False
101|    FontUnderline  = 0  'False
102|    Height         = 300
103|    HelpContextID  = 1000
104|    Left           = 3690
105|    TabIndex       = 9
106|    Text           = "[Easting UTM14]"
107|    Top            = 1530
108|    Width          = 2625
```

```
End
Begin ComboBox cmb_zqColor
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 300
  HelpContextID = 2000
  Left          = 3780
  TabIndex      = 19
  Text          = "0"
  Top           = 3330
  Width         = 2535
End
Begin ComboBox ZSymbol
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 300
  HelpContextID = 2610
  Left          = 735
  TabIndex      = 15
  Text          = "45"
  Top           = 2790
  Width         = 2595
End
Begin ComboBox cmb_zqFontSize
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 300
  HelpContextID = 2500
  Left          = 3780
  TabIndex      = 17
  Text          = "12"
  Top           = 2790
  Width         = 2535
End
Begin TextBox txtThemeName
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 315
```

```
163|     Left          =  1800
164|     TabIndex      =  1
165|     Text          =  "Title for symbol theme"
166|     Top           =  90
167|     Width         =  3975
168|   End
169|   Begin TextBox txtCriteria
170|     FontBold      =  0  'False
171|     FontItalic    =  0  'False
172|     FontName      =  "MS Sans Serif"
173|     FontSize      =  8.25
174|     FontStrikethru =  0  'False
175|     FontUnderline =  0  'False
176|     Height        =  645
177|     HelpContextID =  1600
178|     Left          =  810
179|     MultiLine     =  -1  'True
180|     ScrollBars    =  2  'Vertical
181|     TabIndex      =  13
182|     Top           =  1890
183|     Width         =  2535
184|   End
185|   Begin TextBox txtSpatialQuerySQL
186|     FontBold      =  0  'False
187|     FontItalic    =  0  'False
188|     FontName      =  "MS Sans Serif"
189|     FontSize      =  8.25
190|     FontStrikethru =  0  'False
191|     FontUnderline =  0  'False
192|     Height        =  2115
193|     Left          =  6480
194|     MultiLine     =  -1  'True
195|     ScrollBars    =  2  'Vertical
196|     TabIndex      =  29
197|     Text          =  "SQL Statement to produce data displayed on DataBase Form"
198|     Top           =  2940
199|     Width         =  2715
200|   End
201|   Begin TextBox txtMapSQL
202|     FontBold      =  0  'False
203|     FontItalic    =  0  'False
204|     FontName      =  "MS Sans Serif"
205|     FontSize      =  8.25
206|     FontStrikethru =  0  'False
207|     FontUnderline =  0  'False
208|     Height        =  2325
209|     Left          =  6480
210|     MultiLine     =  -1  'True
211|     ScrollBars    =  2  'Vertical
212|     TabIndex      =  27
213|     Text          =  "SQL statement to produce map data"
214|     Top           =  300
215|     Width         =  2715
216|   End
```

```
Begin ComboBox cmbTblQryList
  DragIcon      = PNTOPT10.FRX:0302
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 300
  Left          = 1425
  Style         = 2  'Dropdown List
  TabIndex      = 3
  Top           = 480
  Width         = 4935
End
Begin CommandButton cmdCancel
  Caption       = "&Cancel"
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 315
  Left          = 5040
  TabIndex      = 25
  Top           = 4770
  Width         = 1365
End
Begin CommandButton cmdOK
  Caption       = "&OK"
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 315
  Left          = 5040
  TabIndex      = 24
  Top           = 4320
  Width         = 1365
End
Begin ComboBox cmbFieldList
  DragIcon      = PNTOPT10.FRX:0604
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 300
  Left          = 1425
  Sorted        = -1  'True
```

```
271|     Style         = 2 'Dropdown List
272|     TabIndex      = 5
273|     Top           = 885
274|     Width         = 4935
275|   End
276|   Begin Frame zFrameFont
277|     Caption       = "&Font"
278|     FontBold      = 0 'False
279|     FontItalic    = 0 'False
280|     FontName      = "MS Sans Serif"
281|     FontSize      = 8.25
282|     FontStrikethru = 0 'False
283|     FontUnderline = 0 'False
284|     Height        = 915
285|     HelpContextID = 2400
286|     Left          = 90
287|     TabIndex      = 20
288|     Top           = 3240
289|     Width         = 3120
290|     Begin ComboBox cmb_zqFontCode
291|       BackColor   = &H000000FF&
292|       FontBold    = 0 'False
293|       FontItalic  = 0 'False
294|       FontName    = "MS Sans Serif"
295|       FontSize    = 8.25
296|       FontStrikethru = 0 'False
297|       FontUnderline = 0 'False
298|       ForeColor   = &H00000000&
299|       Height      = 300
300|       HelpContextID = 2400
301|       Left        = 630
302|       Style       = 2 'Dropdown List
303|       TabIndex    = 23
304|       Top         = 540
305|       Width       = 2445
306|     End
307|     Begin ComboBox cmbFontList
308|       FontBold    = 0 'False
309|       FontItalic  = 0 'False
310|       FontName    = "MS Sans Serif"
311|       FontSize    = 8.25
312|       FontStrikethru = 0 'False
313|       FontUnderline = 0 'False
314|       Height      = 300
315|       HelpContextID = 2400
316|       Left        = 630
317|       Style       = 2 'Dropdown List
318|       TabIndex    = 21
319|       Top         = 180
320|       Width       = 2445
321|     End
322|     Begin Label zzzlblFontCodeNum
323|       Caption     = "Font Co&de"
324|       FontBold    = 0 'False
```

```
325|       FontItalic    = 0  'False
326|       FontName      = "MS Sans Serif"
327|       FontSize      = 8.25
328|       FontStrikethru = 0  'False
329|       FontUnderline = 0  'False
330|       Height        = 400
331|       Left          = 90
332|       TabIndex      = 22
333|       Top           = 450
334|       Width         = 555
335|     End
336|   End
337|   Begin Label ACTION
338|     BackColor     = &H0000FFFF&
339|     ForeColor     = &H000000FF&
340|     Height        = 195
341|     Left          = 960
342|     TabIndex      = 33
343|     Top           = 2040
344|     Visible       = 0  'False
345|     Width         = 1995
346|   End
347|   Begin Image imgFieldName
348|     Height        = 480
349|     Left          = 360
350|     Picture       = PNTOPT10.FRX:0906
351|     Top           = 810
352|     Width         = 480
353|   End
354|   Begin Line zzLine2
355|     BorderWidth   = 2
356|     X1            = 6420
357|     X2            = 0
358|     Y1            = 2610
359|     Y2            = 2610
360|   End
361|   Begin Line zzzLine1
362|     BorderWidth   = 3
363|     X1            = 0
364|     X2            = 6360
365|     Y1            = 1440
366|     Y2            = 1440
367|   End
368|   Begin Image zIMGExample
369|     Height        = 480
370|     Left          = 5850
371|     Picture       = PNTOPT10.FRX:0C08
372|     Top           = 15
373|     Width         = 480
374|   End
375|   Begin Label lblThemeName
376|     Caption       = "T&itle for this theme"
377|     FontBold      = 0  'False
378|     FontItalic    = 0  'False
```

```
379|     FontName      = "MS Sans Serif"
380|     FontSize      = 8.25
381|     FontStrikethru = 0  'False
382|     FontUnderline  = 0  'False
383|     Height        = 225
384|     Left          = 90
385|     TabIndex      = 0
386|     Top           = 90
387|     Width         = 1650
388|   End
389|   Begin Label zzzlblColor
390|     Alignment     = 2  'Center
391|     Caption       = "Co&lor"
392|     FontBold      = 0  'False
393|     FontItalic    = 0  'False
394|     FontName      = "MS Sans Serif"
395|     FontSize      = 8.25
396|     FontStrikethru = 0  'False
397|     FontUnderline  = 0  'False
398|     Height        = 285
399|     Left          = 3240
400|     TabIndex      = 18
401|     Top           = 3330
402|     Width         = 615
403|   End
404|   Begin Label zzzlblZSymbol
405|     Alignment     = 2  'Center
406|     Caption       = "S&ymbol  "
407|     FontBold      = 0  'False
408|     FontItalic    = 0  'False
409|     FontName      = "MS Sans Serif"
410|     FontSize      = 8.25
411|     FontStrikethru = 0  'False
412|     FontUnderline  = 0  'False
413|     Height        = 315
414|     Left          = 90
415|     TabIndex      = 14
416|     Top           = 2790
417|     Width         = 855
418|   End
419|   Begin Label zzzzlblCriteia
420|     Caption       = "C&riteria"
421|     FontBold      = 0  'False
422|     FontItalic    = 0  'False
423|     FontName      = "MS Sans Serif"
424|     FontSize      = 8.25
425|     FontStrikethru = 0  'False
426|     FontUnderline  = 0  'False
427|     Height        = 255
428|     Left          = 90
429|     TabIndex      = 12
430|     Top           = 2070
431|     Width         = 615
432|   End
```

```
433|   Begin Label zzzzlblDBFormSQL
434|     Caption       = "Spatial Query SQL"
435|     FontBold      = 0  'False
436|     FontItalic    = 0  'False
437|     FontName      = "MS Sans Serif"
438|     FontSize      = 8.25
439|     FontStrikethru = 0  'False
440|     FontUnderline = 0  'False
441|     Height        = 465
442|     Left          = 6480
443|     TabIndex      = 28
444|     Top           = 2700
445|     Width         = 2595
446|   End
447|   Begin Label zzzzlblMapTileSQL
448|     Caption       = "Map Symbol SQL"
449|     FontBold      = 0  'False
450|     FontItalic    = 0  'False
451|     FontName      = "MS Sans Serif"
452|     FontSize      = 8.25
453|     FontStrikethru = 0  'False
454|     FontUnderline = 0  'False
455|     Height        = 465
456|     Left          = 6480
457|     TabIndex      = 26
458|     Top           = 75
459|     Width         = 2595
460|   End
461|   Begin Label zzzlblFields
462|     Caption       = "&Fields"
463|     FontBold      = 0  'False
464|     FontItalic    = 0  'False
465|     FontName      = "MS Sans Serif"
466|     FontSize      = 8.25
467|     FontStrikethru = 0  'False
468|     FontUnderline = 0  'False
469|     Height        = 285
470|     Left          = 900
471|     TabIndex      = 4
472|     Top           = 900
473|     Width         = 555
474|   End
475|   Begin Label zzzlblTables
476|     Caption       = "&Tables/Queries"
477|     DragIcon      = PNTOPT10.FRX:0F0A
478|     DragMode      = 1  'Automatic
479|     FontBold      = 0  'False
480|     FontItalic    = 0  'False
481|     FontName      = "MS Sans Serif"
482|     FontSize      = 8.25
483|     FontStrikethru = 0  'False
484|     FontUnderline = 0  'False
485|     Height        = 285
486|     Left          = 90
```

```
   TabIndex      = 2
   Top           = 480
   Width         = 1365
End
Begin Label zzzlblName
   Caption       = "&Unique Number"
   FontBold      = 0  'False
   FontItalic    = 0  'False
   FontName      = "MS Sans Serif"
   FontSize      = 8.25
   FontStrikethru = 0  'False
   FontUnderline = 0  'False
   Height        = 525
   Left          = 90
   TabIndex      = 6
   Top           = 1530
   Width         = 615
End
Begin Label zzzlblZSize
   Alignment     = 2  'Center
   Caption       = "&Size"
   FontBold      = 0  'False
   FontItalic    = 0  'False
   FontName      = "MS Sans Serif"
   FontSize      = 8.25
   FontStrikethru = 0  'False
   FontUnderline = 0  'False
   Height        = 405
   Left          = 3330
   TabIndex      = 16
   Top           = 2790
   Width         = 495
End
Begin Label zzzlblY
   Alignment     = 1  'Right Justify
   Caption       = "&Y"
   FontBold      = 0  'False
   FontItalic    = 0  'False
   FontName      = "MS Sans Serif"
   FontSize      = 8.25
   FontStrikethru = 0  'False
   FontUnderline = 0  'False
   Height        = 255
   Left          = 3330
   TabIndex      = 10
   Top           = 1980
   Width         = 315
End
Begin Label zzzlblX
   Alignment     = 1  'Right Justify
   Caption       = "&X"
   FontBold      = 0  'False
   FontItalic    = 0  'False
   FontName      = "MS Sans Serif"
```

```
541|      FontSize      =  8.25
542|      FontStrikethru =  0  'False
543|      FontUnderline  =  0  'False
544|      Height        =  255
545|      Left          =  3420
546|      TabIndex      =  7
547|      Top           =  1530
548|      Width         =  240
549|    End
550| End
551| Option Explicit
552| Option Compare Text
553|
554| Dim flgValidSyntax As Integer
555| Dim ListOfFieldsForDBForm As String
556|
557| Dim FlagActivateDragDrop As Integer
558| Dim tblPointMap As Table
559| Dim FieldsInCurrentQT() As String, FieldListWBrackets() As String
560| Dim FieldDetails() As FieldDescription
561| Dim nStoredMaps As Long
562|
563| 'Const SizeLookFor = "SIZE HEIGHT WIDTH ", SizeShowAlways = "10 12 18 "
564|
565| Dim flgIgnoreMapSQLChange  As Integer
566|
567| Sub ACTION_Change ()
568|   Dim tmp As String
569|   tmp = UCase$(Me!ACTION.Caption)
570|   Select Case tmp
571|    Case "Decode MapSQL"
572|       tmp = Me!txtMapSQL
573|       Call DecodeMapSQL(tmp)
574|    Case "wait" 'do nothing
575|    Case "refresh"
576|       Call Form_Load
577|    Case Else
578|   End Select
579| End Sub
580|
581| Sub cmb_zqColor_Change ()
582|  SetMapSQL
583| End Sub
584|
585| Sub cmb_zqColor_Click ()
586|  SetMapSQL
587| End Sub
588|
589| Sub cmb_zqColor_DragDrop (Source As Control, X As Single, Y As Single)
590|  cmb_zqColor = Mid$(cmb_zqColor, 1, cmb_zqColor.SelStart) & " " & "[" & Source & "]"
     & " " & Mid$(cmb_zqColor, cmb_zqColor.SelStart + cmb_zqColor.SelLength + 1,
     32768)
591| 'Drops the selected text at the current insertion point
592|
```

```
End Sub

Sub cmb_zqFeatureID_Change ()
   SetMapSQL
   Call DBFormSQLAddField("")
End Sub

Sub cmb_zqFeatureID_Click ()
 SetMapSQL
 Call DBFormSQLAddField("")
End Sub

Sub cmb_zqFeatureID_DragDrop (Source As Control, X As Single, Y As Single)
cmb_zqFeatureID.AddItem "[" & Source & "]"
cmb_zqFeatureID.ListIndex = cmb_zqFeatureID.ListCount - 1
'Drops the selected text over the entire field, for use on
' NAME only, because we'll need this for matching and it
' NAME should never be a calculated field, at least not easily.

End Sub

Sub cmb_zqFontCode_Click ()
 SetMapSQL
End Sub

Sub cmb_zqFontCode_DragDrop (Source As Control, X As Single, Y As Single)
 cmb_zqFontCode.AddItem "[" & Source & "]"
 cmb_zqFontCode.ListIndex = cmb_zqFontCode.ListCount - 1
End Sub

Sub cmb_zqFontSize_Change ()
 SetMapSQL
End Sub

Sub cmb_zqFontSize_Click ()
 SetMapSQL
End Sub

Sub cmb_zqFontSize_DragDrop (Source As Control, X As Single, Y As Single)
 'cmb_zqFontSize = Mid$(cmb_zqFontSize, 1, cmb_zqFontSize.SelStart) & " " &
AddBracketsAsNeeded(CStr(Source)) & " " & Mid$(cmb_zqFontSize,
cmb_zqFontSize.SelStart + cmb_zqFontSize.SelLength + 1, 32768)
 cmb_zqFontSize = Mid$(cmb_zqFontSize, 1, cmb_zqFontSize.SelStart) & " " & "[" &
Source & "]" & " " & Mid$(cmb_zqFontSize, cmb_zqFontSize.SelStart +
cmb_zqFontSize.SelLength + 1, 32768)
'Drops the selected text at the current insertion point
End Sub

Sub cmb_zqFontSize_KeyDown (KeyCode As Integer, Shift As Integer)
   If KeyCode = 13 Then
    cmb_zqFontSize.AddItem cmb_zqFontSize.Text
    cmb_zqFontSize.ListIndex = cmb_zqFontSize.ListCount - 1
    KeyCode = 0
   End If
```

```
643| End Sub
644|
645| Sub cmb_zqX_Change ()
646|   SetMapSQL
647| End Sub
648|
649| Sub cmb_zqX_Click ()
650|  SetMapSQL
651| End Sub
652|
653| Sub cmb_zqX_DragDrop (Source As Control, X As Single, Y As Single)
654| 'X = Mid$(X, 1, X.SelStart) & " " & AddBracketsAsNeeded(CStr(Source)) & " " & Mid$(X,
    X.SelStart + X.SelLength + 1, 32768)
655|    cmb_zqX = Mid$(Me!cmb_zqX, 1, Me!cmb_zqX.SelStart) & " [" & CStr(Source) & "] " &
    Mid$(Me!cmb_zqX, Me!cmb_zqX.SelStart + Me!cmb_zqX.SelLength + 1, 32768)
656| 'Drops the selected text at the current insertion point
657| End Sub
658|
659| Sub cmb_zqY_Change ()
660|  SetMapSQL
661| End Sub
662|
663| Sub cmb_zqY_Click ()
664|  SetMapSQL
665| End Sub
666|
667| Sub cmb_zqY_DragDrop (Source As Control, X As Single, Y As Single)
668| 'Y = Mid$(Y, 1, Y.SelStart) & " " & AddBracketsAsNeeded(CStr(Source)) & " " & Mid$(Y,
    Y.SelStart + Y.SelLength + 1, 32768)
669| Me.cmb_zqY = Mid$(Me.cmb_zqY, 1, Me.cmb_zqY.SelStart) & " [" & CStr(Source) & "] "
    & Mid$(Me.cmb_zqY, Me.cmb_zqY.SelStart + Me.cmb_zqY.SelLength + 1, 32768)
670| 'Drops the selected text at the current insertion point
671| End Sub
672|
673| Sub cmbFieldList_KeyDown (KeyCode As Integer, Shift As Integer)
674|   If KeyCode = KEY_INSERT And Shift = 2 Then 'CTRL-Insert?
675|     clipboard.Clear
676|     clipboard.SetText ("[" & cmbFieldList & "]")
677|   End If
678| End Sub
679|
680| Sub cmbFontList_Click ()
681| Dim Temp As String
682| Temp = cmbFontList
683|   cmb_zqFontCode.AddItem Format$(FontIDFromName(Temp))
684|   cmb_zqFontCode.ListIndex = cmb_zqFontCode.ListCount - 1
685| End Sub
686|
687| Sub cmbTblQryList_Click ()
688| If Me!ACTION.Caption = "wait" Then Exit Sub
689|
690| On Error GoTo ERRcmbTblQryList_Click
691|
692|  If Not flgIgnoreMapSQLChange Then
```

```
693|    'For usual case, set the SpatialQuerySQL and the theme name
694|    txtSpatialQuerySQL = "SELECT * FROM " &
       AddBracketsAsNeeded(CStr(cmbTblQryList))
695|    ListOfFieldsForDBForm = "*"
696|    txtThemeName = "SYMBOLS: " & cmbTblQryList & "  " & Now
697|   End If
698|
699|
700|  Dim TableName As String
701|   TableName = cmbTblQryList
702|   Set snpFieldList = MapDB.CreateSnapshot(TableName)
703|   nFields = snpFieldList.Fields.Count
704|   cmbFieldList.Clear
705|   ReDim FieldsInCurrentQT(1 To nFields), FieldListWBrackets(1 To nFields)
706|   ', FieldDetails(1 To nFields)
707|
708|
709|   For i = 1 To nFields
710|     cmbFieldList.AddItem snpFieldList.Fields(i - 1).Name
711|     FieldsInCurrentQT(i) = snpFieldList.Fields(i - 1).Name
712|     FieldListWBrackets(i) = "[" & snpFieldList.Fields(i - 1).Name & "]"
713|   Next i
714|   snpFieldList.Close
715|   cmbFieldList.ListIndex = 0
716|   txtCriteria = ""
717|   FlagActivateDragDrop = 0
718|
719|  cmb_zqFontSize.Clear
720|  FillCMBFromSGSTLST FieldListWBrackets(), SizeLookFor, SizeShowAlways,
       cmb_zqFontSize
721|  cmb_zqFontSize.ListIndex = 0
722|
723|  ZSymbol.Clear
724|  FillCMBFromSGSTLST FieldListWBrackets(), SymbolCodeLookFor,
       SymbolCodeShowAlways, ZSymbol
725|  ZSymbol.ListIndex = 0
726|
727|  cmb_zqColor.Clear
728|  FillCMBFromSGSTLST FieldListWBrackets(), ColorLookFor, ColorShowAlways,
       cmb_zqColor
729|  cmb_zqColor.ListIndex = 0
730|
731|  cmb_zqFontCode.Clear
732|  FillCMBFromSGSTLST FieldListWBrackets(), FontCodeLookFor,
       FontCodeShowAlways, cmb_zqFontCode
733|  cmb_zqFontCode.ListIndex = 0
734|
735|  cmb_zqX.Clear
736|  FillCMBFromSGSTLST FieldListWBrackets(), XLookFor, XShowAlways, cmb_zqX
737|  cmb_zqX.ListIndex = 0
738|
739|  cmb_zqY.Clear
740|  FillCMBFromSGSTLST FieldListWBrackets(), YLookFor, YShowAlways, cmb_zqY
741|  cmb_zqY.ListIndex = 0
```

```
742|
743| cmb_zqFeatureID.Clear
744| FillCMBFromSGSTLST FieldListWBrackets(), FeatureNameLookFor,
     FeatureNameShowAlways, cmb_zqFeatureID
745| cmb_zqFeatureID.ListIndex = 0
746|
747|  frmPointOptionsSimple.cmbFieldList.ListIndex = 0
748| Exit Sub
749| ERRcmbTblQryList_Click:
750|   MsgBox "Error #" & Err & " occurred in cmbTblQryList_Click." & CRLF & Error
751|   Err = 0
752|   Exit Sub
753|   Resume Next
754|
755|
756| End Sub
757|
758| Sub cmdCancel_Click ()
759|  Me.Hide
760| End Sub
761|
762| Sub cmdOK_Click ()
763| If Me!cmbTblQryList = "" Then
764|    MsgBox "A query or table must be selected before trying to display its results." & CRLF
     & "Or you can hit CANCEL to not add a point theme right now and get out of here."
765|    Me!cmbTblQryList.SetFocus
766|    SendKeys "%{DOWN}"
767|    Exit Sub
768| End If
769| If Me!cmbFontList = "" Then
770|    MsgBox "A font must be selected before using it as point symbols." & CRLF & "Or you
     can hit CANCEL to not add a point theme right now and get out of here."
771|    Me!cmbFontList.SetFocus
772|    SendKeys "%{DOWN}"
773|    Exit Sub
774| End If
775|
776| 'Call DBFormSQLAddField("")
777|  Me!cmdOK.SetFocus
778| Me.Hide
779|   Me!cmdOK.FontItalic = True
780|
781| End Sub
782|
783| Sub DBFormSQLAddField (NewField As String)
784| If flgIgnoreMapSQLChange Then Exit Sub
785| If NewField <> "" Then
786|  If ListOfFieldsForDBForm = "*" Then
787|    ListOfFieldsForDBForm = "[" & NewField & "]"
788|   Else
789|    ListOfFieldsForDBForm = ListOfFieldsForDBForm & ", [" & NewField & "]"
790|  End If
791| End If
792| txtSpatialQuerySQL = "SELECT " & ListOfFieldsForDBForm & ", " & cmb_zqFeatureID &
```

```
      " AS zqFeatureID FROM " & "[" & cmbTblQryList & "]"
 793| End Sub
 794|
 795| Sub DecodeMapSQL (strSQLIn As String)
 796|   Dim i As Long, h As Integer, strTemp As String
 797|
 798|   strSQLIn = Trim$(strSQLIn)
 799|   If strSQLIn = "" Then Exit Sub
 800|
 801|
 802| 'If the MapSQL was changed by another form or by the user,
 803| '   set the comboboxes to reflect the changes as best possible.
 804|   Dim strParameters  As String, strFields As String, strFrom As String, strWhere As
      String, strGroupBy As String, strHaving As String, strOrderBy As String
 805|
 806|
 807|
 808|   If SplitSQLStatement(strSQLIn, strParameters, strFields, strFrom, strWhere,
      strGroupBy, strHaving, strOrderBy) = False Then Exit Sub
 809|
 810|   If strParameters <> "" Then
 811|     MsgBox "WHEAT programs do not run parameter queries.  Please choose another
      map SQL string."
 812|     Exit Sub
 813|   End If
 814|   flgIgnoreMapSQLChange = True
 815|   Dim strProblems As String
 816|
 817|   strFrom = Strip2Brackets(strFrom)
 818| 'Set the source table or query
 819|   i = IndexOfStringInCOMBOBOX(strFrom, Me!cmbTblQryList)
 820|   Me!cmbTblQryList.ListIndex = i
 821|    DoEvents 'Let the click event occur for the QT List
 822|
 823| 'Get field assignments
 824|   Dim strRawName() As String, strAsName() As String, nFields As Long, nF As Long
 825|   Call PullFieldNamesFromString(strFields, strRawName(), strAsName(), nFields)
 826|   'For i = 1 To nFields
 827|   '   dDebug.Print strRawName(i), , strAsName(i)
 828|   'Next i
 829|
 830|    'Set X Coordinate field
 831|   For nF = 1 To nFields
 832|    If strAsName(nF) = "zqX" Then
 833|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqX)
 834|      If i <> -1 Then
 835|        Me.cmb_zqX.ListIndex = i
 836|       Else
 837|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
      for X-coordinates, but this routine cannot find it.  You will have to select this one
      yourself." & CRLF
 838|      End If
 839|      Exit For
 840|    End If
```

```
841|   Next nF
842|
843|    'Set Y Coordinate field
844|   For nF = 1 To nFields
845|    If strAsName(nF) = "zqY" Then
846|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqY)
847|      If i <> -1 Then
848|        Me.cmb_zqY.ListIndex = i
849|       Else
850|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
    for Y-coordinates, but this routine cannot find it.  You will have to select this one
    yourself." & CRLF
851|      End If
852|      Exit For
853|    End If
854|   Next nF
855|
856|    'Set FeatureID field
857|   For nF = 1 To nFields
858|    If strAsName(nF) = "zqFeatureID" Then
859|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFeatureID)
860|      If i <> -1 Then
861|        Me.cmb_zqFeatureID.ListIndex = i
862|       Else
863|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
    for feature ID, but this routine cannot find it.  You will have to select this one yourself." &
    CRLF
864|      End If
865|      Exit For
866|    End If
867|   Next nF
868|
869|    'Font size
870|   For nF = 1 To nFields
871|    If strAsName(nF) = "zqFontSize" Then
872|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFontSize)
873|      If i <> -1 Then
874|        Me.cmb_zqFontSize.ListIndex = i
875|       Else
876|        i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFontSize)
877|        If i <> -1 Then
878|          Me.cmb_zqFontSize.ListIndex = i
879|         Else
880|          strProblems = strProblems & "This theme originally used " & strRawName(nF) &
    " for the font size, but this routine cannot find it.  You will have to select this one yourself."
    & CRLF
881|        End If
882|      End If
883|      Exit For
884|    End If
885|   Next nF
886|
887|      'The color
888|   For nF = 1 To nFields
```

```
 889|    If strAsName(nF) = "zqColor" Then
 890|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqColor)
 891|      If i <> -1 Then
 892|        Me.cmb_zqColor.ListIndex = i
 893|       Else
 894|        i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqColor)
 895|        If i <> -1 Then
 896|          Me.cmb_zqColor.ListIndex = i
 897|         Else
 898|
 899|           strProblems = strProblems & "This theme originally used " & strRawName(nF) &
      " for the color, but this routine cannot find it.  You will have to select this one yourself." &
      CRLF
 900|       End If
 901|      End If
 902|      Exit For
 903|    End If
 904|   Next nF
 905|
 906|     'The font code
 907|   For nF = 1 To nFields
 908|    If strAsName(nF) = "zqFontCode" Then
 909|      'If the fontcode is stored in a field, this will find it.
 910|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFontCode)
 911|      If i <> -1 Then
 912|        Me.cmb_zqFontCode.ListIndex = i
 913|       Else
 914|        'if the font code was not in a field found, it should probably be a
 915|        ' a constant.
 916|        h = Val(strRawName(nF))
 917|        i = IndexOfStringInCOMBOBOX(FontNameFromID(h), Me!cmbFontList)
 918|        If i <> -1 Then
 919|          Me!cmbFontList.ListIndex = i
 920|         Else
 921|          Me!cmbFontList.ListIndex = 0
 922|        End If
 923|      End If
 924|      Exit For
 925|    End If
 926|   Next nF
 927|
 928|
 929|    txtCriteria = strWhere
 930|
 931|
 932|   flgIgnoreMapSQLChange = False
 933|    If strProblems <> "" Then
 934|      Beep
 935|      MsgBox strProblems
 936|    End If
 937|
 938|  End Sub
 939|
 940|  Sub Form_Load ()
```

```
941|
942|    FlagActivateDragDrop = 0
943|    Me!cmbFontList.Clear
944|    Call LoadComboBoxWithFontNames(Me!cmbFontList)
945|
946|    Me!cmbTblQryList.Clear
947|    For i = 1 To nTblQrys
948|     Me!cmbTblQryList.AddItem strTblQryList(i)
949|    Next i
950|
951| End Sub
952|
953| Sub Form_Paint ()
954|   If cmbTblQryList.ListIndex = -1 Then cmbTblQryList.ListIndex = 0
955|   If cmbFontList.ListIndex = -1 Then cmbFontList.ListIndex = 1
956|   cmdOK.FontItalic = False
957| End Sub
958|
959| Sub imgFieldName_Click ()
960|   cmbFieldList.Drag
961| End Sub
962|
963| Sub PickCharacterCodeWithForm ()
964|  If cmbFontList = "" Then
965|     MsgBox "Select a font before trying to pick a symbol"
966|     frmPointOptionsSimple.cmbFontList.SetFocus
967|     SendKeys "%{DOWN}"
968|     Exit Sub
969|  End If
970|     frmPickCharacter.cmbFontList.Visible = False
971|     frmPickCharacter.txtCharNumber = Val(ZSymbol)
972|     frmPickCharacter.picShowCharacter.FontName = cmbFontList.Text
973|     frmPickCharacter.Show 1
974|     If frmPickCharacter.cmdOK.FontItalic Then Call
      AddAndHighlightItem(CStr(frmPickCharacter.txtCharNumber.Text), ZSymbol)
975| End Sub
976|
977| Sub picSample_Paint ()
978|  ShowSample
979| End Sub
980|
981| Sub SetMapSQL ()
982|
983| If flgIgnoreMapSQLChange Then Exit Sub
984| Dim Temp As String
985| Dim TempSQL As String
986|  ShowSample
987| TempSQL = "SELECT DISTINCTROW "
988| ' Assemble a new MapSQL statement.
989| TempSQL = TempSQL & AddVarNameAsNeeded(GetGoodValueField(cmb_zqX.Text),
      "zqX") & " , "
990| TempSQL = TempSQL & AddVarNameAsNeeded(GetGoodValueField(cmb_zqY.Text),
      "zqY") & " , "
991| TempSQL = TempSQL &
```

```
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqFontSize.Text), "ZqSize") & " , "
 992| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqColor.Text), "ZqColor") & " , "
 993| TempSQL = TempSQL & AddVarNameAsNeeded(GetGoodValueField(ZSymbol.Text),
      "ZqSymbol") & " , "
 994| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqFontCode.Text), "ZqFontCode") &
      " , "
 995| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqFeatureID.Text), "zqFeatureID") &
      " "
 996|
 997|
 998|
 999| TempSQL = TempSQL & " FROM [" & cmbTblQryList & "] "
1000|
1001|
1002|       'Do not add any criteria of your own.
1003| If (txtCriteria = "") Then
1004|    TempSQL = TempSQL
1005|   Else
1006|    TempSQL = TempSQL & "  WHERE (" & txtCriteria & ") "
1007| End If
1008|
1009|
1010| 'Build the Order By clause
1011| Dim strOrderBy As String    'changed 951201
1012|   strOrderBy = ""
1013|
1014|   Temp = GetFieldForOrderBy(cmb_zqFontCode.Text)
1015|   If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
1016|
1017|   Temp = GetFieldForOrderBy(cmb_zqColor.Text)
1018|   If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
1019|
1020|
1021|   Temp = GetFieldForOrderBy(cmb_zqFontSize.Text)
1022|   If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
1023|
1024|   If strOrderBy <> "" Then
1025|     strOrderBy = " ORDER BY " & Left$(strOrderBy, Len(strOrderBy) - 2)
1026|   End If
1027|
1028|   TempSQL = TempSQL & strOrderBy & " ; "
1029|
1030|
1031| ' Now put the SQL statement into the appropriate text box.
1032| flgIgnoreMapSQLChange = True
1033|   txtMapSQL = TempSQL
1034| flgIgnoreMapSQLChange = False
1035|
1036| End Sub
1037|
1038| Sub ShowSample ()
```

```
Dim OldAlignment As Integer
OldAlignment = SetTextAlign(Me.picSample.hDC, TA_BASELINE)
'Display for user
  Dim XPoint As Single, YPoint As Single
  XPoint = .5 * picSample.ScaleWidth
  YPoint = .5 * picSample.ScaleHeight
  picSample.Cls
  picSample.Line (0, YPoint)-(2 * XPoint, YPoint), RGB(255, 0, 0)
  picSample.Line (XPoint, 0)-(XPoint, 2 * YPoint), RGB(255, 0, 0)
  picSample.CurrentX = XPoint
  picSample.CurrentY = YPoint
  Dim Holder As Single
  Holder = Val(cmb_zqFontSize.Text)
  If Holder > 0 Then
     picSample.FontSize = Holder
    Else
     picSample.FontSize = 18
  End If
  OldAlignment = Val(cmb_zqFontCode.Text)
  Call SetFontInPIC(OldAlignment, Me.picSample)
  OldAlignment = Val(ZSymbol.Text)
  picSample.ForeColor = Val(cmb_zqColor.Text)
  picSample.Print Chr$(Printable(OldAlignment))

End Sub

Sub txtCriteria_Change ()
 SetMapSQL
End Sub

Sub txtCriteria_DragDrop (Source As Control, X As Single, Y As Single)
'txtCriteria = Mid$(txtCriteria, 1, txtCriteria.SelStart) & " " &
    AddBracketsAsNeeded(CStr(Source)) & " " & Mid$(txtCriteria, txtCriteria.SelStart +
    txtCriteria.SelLength + 1, 32768)
txtCriteria = Mid$(txtCriteria, 1, txtCriteria.SelStart) & " " & "[" & Source & "]" & " " &
    Mid$(txtCriteria, txtCriteria.SelStart + txtCriteria.SelLength + 1, 32768)
'Drops the selected text at the current insertion point
End Sub

Sub txtMapSQL_KeyPress (KeyAscii As Integer)

If flgIgnoreMapSQLChange Then Exit Sub
If KeyAscii <> 13 Then Exit Sub
 KeyAscii = 0'Swallow the return key
 Dim strSQL As String
 strSQL = txtMapSQL
 Call DecodeMapSQL(strSQL)



End Sub

Sub txtSpatialQuerySQL_DragDrop (Source As Control, X As Single, Y As Single)
  Call DBFormSQLAddField(CStr(Source))
```

```
End Sub

Sub txtSpatialQuerySQL_KeyPress (KeyAscii As Integer)
Select Case KeyAscii
   Case 13 'Return/ENTER
   KeyAscii = 0
   'Maybe decode the QS SQL
End Select

End Sub

Sub ZSymbol_Change ()
 SetMapSQL
End Sub

Sub ZSymbol_Click ()
 SetMapSQL
End Sub

Sub ZSymbol_DragDrop (Source As Control, X As Single, Y As Single)
 'ZSymbol = Mid$(ZSymbol, 1, ZSymbol.SelStart) & " " &
AddBracketsAsNeeded(CStr(Source)) & " " & Mid$(ZSymbol, ZSymbol.SelStart +
ZSymbol.SelLength + 1, 32768)
  ZSymbol = Mid$(ZSymbol, 1, ZSymbol.SelStart) & " " & "[" & Source & "]" & " " &
Mid$(ZSymbol, ZSymbol.SelStart + ZSymbol.SelLength + 1, 32768)
'Drops the selected text at the current insertion point
End Sub

Sub zzzlblColor_Click ()
  Dim ColorWas As Long, NewColor As Long
  ColorWas = Val(cmb_zqColor)
  NewColor = GetColorWithCmnDialog(ColorWas)
  If NewColor <> ColorWas Then AddAndHighlightItem Format$(NewColor),
cmb_zqColor
End Sub

Sub zzzlblColor_DragDrop (Source As Control, X As Single, Y As Single)
    'cmb_zqColor = "[" & Source & "]"
 cmb_zqColor.AddItem "[" & Source & "]"
 cmb_zqColor.ListIndex = cmb_zqColor.ListCount - 1

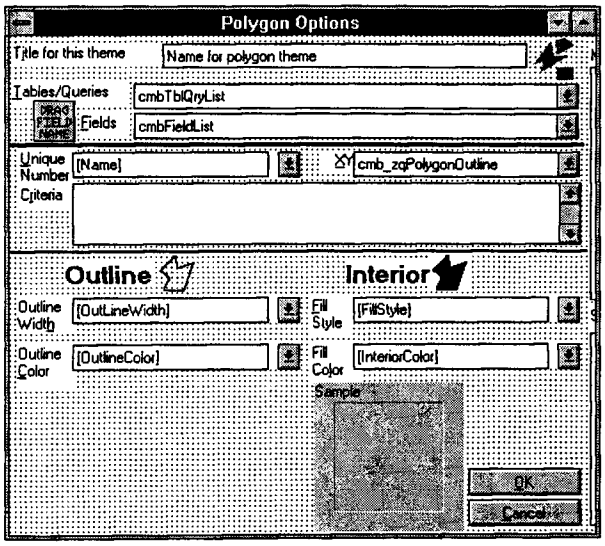
End Sub

Sub zzzlblFields_Click ()
 cmbFieldList.Drag
End Sub

Sub zzzlblFontCodeNum_DragDrop (Source As Control, X As Single, Y As Single)
 cmb_zqFontCode.AddItem "[" & Source & "]"
 cmb_zqFontCode.ListIndex = cmb_zqFontCode.ListCount - 1
End Sub

Sub zzzlblName_DragDrop (Source As Control, X As Single, Y As Single)
'Sets all of the following Z-field to whatever was dropped on it
```

```
1140|  cmb_zqFeatureID.AddItem "[" & Source & "]"
1141|  cmb_zqFeatureID.ListIndex = cmb_zqFeatureID.ListCount - 1
1142|
1143|  End Sub
1144|
1145|  Sub zzzlblX_DragDrop (Source As Control, X As Single, Y As Single)
1146|  'X = "[" & CStr(Source) & "]"'Sets all of the following X-field to whatever was dropped on
       it
1147|   Me!cmb_zqX.AddItem "[" & Source & "]"
1148|   Me!cmb_zqX.ListIndex = Me!cmb_zqX.ListCount - 1
1149|
1150|  End Sub
1151|
1152|  Sub zzzlblY_DragDrop (Source As Control, X As Single, Y As Single)
1153|  'Y = "[" & CStr(Source) & "]"'Sets all of the following Y-field to whatever was dropped on it
1154|   Me!cmb_zqY.AddItem "[" & Source & "]"
1155|   Me!cmb_zqY.ListIndex = Me!cmb_zqY.ListCount - 1
1156|
1157|  End Sub
1158|
1159|  Sub zzzlblZSize_DragDrop (Source As Control, X As Single, Y As Single)
1160|
1161|   cmb_zqFontSize.AddItem "[" & Source & "]"
1162|   cmb_zqFontSize.ListIndex = cmb_zqFontSize.ListCount - 1
1163|  'Add the item to the list at the end, and set the list to
1164|  '  show the just-added item.
1165|       'cmb_zqFontSize = "[" & Source & "]"  'Sets all of the following Z-field to whatever
       was dropped on it
1166|  End Sub
1167|
1168|  Sub zzzlblZSymbol_Click ()
1169|    PickCharacterCodeWithForm
1170|  End Sub
1171|
1172|  Sub zzzlblZSymbol_DragDrop (Source As Control, X As Single, Y As Single)
1173|   'ZSymbol = AddBracketsAsNeeded(CStr(Source))'Sets all of the following Z-field to
       whatever was dropped on it
1174|   'ZSymbol = "[" & Source & "]" 'Sets all of the following Z-field to whatever was dropped
       on it
1175|   ZSymbol.AddItem "[" & Source & "]"
1176|   ZSymbol.ListIndex = ZSymbol.ListCount - 1
1177|
1178|  End Sub
1179|
1180|  Sub zzzzlblCriteia_DragDrop (Source As Control, X As Single, Y As Single)
1181|  'txtCriteria = AddBracketsAsNeeded(CStr(Source))
1182|  txtCriteria = "[" & Source & "]"
1183|  End Sub
1184|
1185|  Sub zzzzlblDBFormSQL_DragDrop (Source As Control, X As Single, Y As Single)
1186|  Call DBFormSQLAddField(CStr(Source))
1187|  End Sub
1188|
```


Polygon Options
Title for this theme
Name for polygon theme
Tables/Queries
cmbTblQryList
Fields
cmbFieldList
Unique Number
[Name]
cmb_zqPolygonOutline
Criteria
Outline
Interior
Outline Width
[OutLineWidth]
Fill Style
[FillStyle]
Outline Color
[OutlineColor]
Fill Color
[InteriorColor]
Sample
OK
Cancel

```
VERSION 2.00
Begin Form frmPolyOptions
  Caption        = "Polygon Options"
  ClientHeight   = 5805
  ClientLeft     = 105
  ClientTop      = 420
  ClientWidth    = 6765
  ControlBox     = 0  'False
  Height         = 6210
  Icon           = POLYOPT4.FRX:0000
  Left           = 45
  LinkTopic      = "Form1"
  MinButton      = 0  'False
  ScaleHeight    = 5805
  ScaleWidth     = 6765
  Top            = 75
  Width          = 6885
  Begin ComboBox cmb_zqOutlineColor
    FontBold       = 0  'False
    FontItalic     = 0  'False
    FontName       = "MS Sans Serif"
    FontSize       = 8.25
    FontStrikethru = 0  'False
    FontUnderline  = 0  'False
    Height         = 300
    HelpContextID  = 2000
    Left           = 720
    TabIndex       = 15
    Text           = "[OutlineColor]"
    Top            = 3600
    Width          = 2625
  End
  Begin ComboBox cmb_zqFillStyle
    FontBold       = 0  'False
    FontItalic     = 0  'False
    FontName       = "MS Sans Serif"
    FontSize       = 8.25
    FontStrikethru = 0  'False
    FontUnderline  = 0  'False
    Height         = 300
    HelpContextID  = 2300
    Left           = 3960
    TabIndex       = 17
    Text           = "[FillStyle]"
    Top            = 3060
    Width          = 2625
  End
  Begin ComboBox cmb_zqFillColor
    FontBold       = 0  'False
    FontItalic     = 0  'False
    FontName       = "MS Sans Serif"
    FontSize       = 8.25
    FontStrikethru = 0  'False
    FontUnderline  = 0  'False
```

```
55|     Height      = 300
56|     HelpContextID  = 2000
57|     Left        = 3960
58|     TabIndex     = 19
59|     Text        = "[InteriorColor]"
60|     Top         = 3600
61|     Width       = 2625
62|   End
63|   Begin ComboBox cmb_zqFeatureID
64|     FontBold     = 0  'False
65|     FontItalic   = 0  'False
66|     FontName     = "MS Sans Serif"
67|     FontSize     = 8.25
68|     FontStrikethru = 0  'False
69|     FontUnderline  = 0  'False
70|     Height       = 300
71|     HelpContextID  = 1500
72|     Left        = 720
73|     TabIndex     = 7
74|     Text        = "[Name]"
75|     Top         = 1350
76|     Width       = 2625
77|   End
78|   Begin SSPanel zpnlOuter
79|     Alignment    = 0  'Left Justify - TOP
80|     AutoSize     = 3  'AutoSize Child To Panel
81|     BackColor    = &H00C0C0C0&
82|     BevelInner   = 1  'Inset
83|     BevelOuter   = 0  'None
84|     BorderWidth  = 15
85|     Font3D       = 0  'None
86|     ForeColor    = &H00FFFFFF&
87|     Height       = 1725
88|     Left        = 3510
89|     TabIndex     = 27
90|     Top         = 4050
91|     Width       = 1725
92|     Begin PictureBox picSample
93|       BackColor    = &H00C0C0C0&
94|       BorderStyle  = 0  'None
95|       Height       = 1250
96|       Left        = 240
97|       ScaleHeight  = 83
98|       ScaleMode    = 3  'Pixel
99|       ScaleWidth   = 83
100|      TabIndex     = 29
101|      Top         = 240
102|      Width       = 1250
103|    End
104|    Begin SSPanel zpnlSampleWord
105|      Alignment    = 0  'Left Justify - TOP
106|      AutoSize     = 3  'AutoSize Child To Panel
107|      BackColor    = &H00C0C0C0&
108|      BevelOuter   = 0  'None
```

```
109|       BorderWidth    = 1
110|       Caption        = "Sample"
111|       Font3D         = 0 'None
112|       FontBold       = 0 'False
113|       FontItalic     = 0 'False
114|       FontName       = "MS Sans Serif"
115|       FontSize       = 8.25
116|       FontStrikethru = 0 'False
117|       FontUnderline  = 0 'False
118|       ForeColor      = &H00000000&
119|       Height         = 220
120|       Left           = 0
121|       TabIndex       = 28
122|       Top            = 0
123|       Width          = 735
124|     End
125|   End
126|   Begin ComboBox cmb_zqOutlineWidth
127|     FontBold       = 0 'False
128|     FontItalic     = 0 'False
129|     FontName       = "MS Sans Serif"
130|     FontSize       = 8.25
131|     FontStrikethru = 0 'False
132|     FontUnderline  = 0 'False
133|     Height         = 300
134|     HelpContextID  = 2100
135|     Left           = 720
136|     TabIndex       = 13
137|     Text           = "[OutLineWidth]"
138|     Top            = 3060
139|     Width          = 2625
140|   End
141|   Begin ComboBox cmb_zqPolygonOutline
142|     FontBold       = 0 'False
143|     FontItalic     = 0 'False
144|     FontName       = "MS Sans Serif"
145|     FontSize       = 8.25
146|     FontStrikethru = 0 'False
147|     FontUnderline  = 0 'False
148|     Height         = 300
149|     HelpContextID  = 1200
150|     Left           = 3960
151|     Style          = 2 'Dropdown List
152|     TabIndex       = 9
153|     Top            = 1350
154|     Width          = 2625
155|   End
156|   Begin TextBox txtThemeName
157|     FontBold       = 0 'False
158|     FontItalic     = 0 'False
159|     FontName       = "MS Sans Serif"
160|     FontSize       = 8.25
161|     FontStrikethru = 0 'False
162|     FontUnderline  = 0 'False
```

```
163|     Height      = 285
164|     Left        = 1755
165|     TabIndex    = 1
166|     Text        = "Name for polygon theme"
167|     Top         = 90
168|     Width       = 4215
169|   End
170|   Begin TextBox txtCriteria
171|     FontBold      = 0  'False
172|     FontItalic    = 0  'False
173|     FontName      = "MS Sans Serif"
174|     FontSize      = 8.25
175|     FontStrikethru = 0  'False
176|     FontUnderline = 0  'False
177|     Height        = 735
178|     HelpContextID = 5
179|     Left          = 720
180|     MultiLine     = -1 'True
181|     ScrollBars    = 2  'Vertical
182|     TabIndex      = 11
183|     Top           = 1710
184|     Width         = 5880
185|   End
186|   Begin TextBox txtSpatialQuerySQL
187|     FontBold      = 0  'False
188|     FontItalic    = 0  'False
189|     FontName      = "MS Sans Serif"
190|     FontSize      = 8.25
191|     FontStrikethru = 0  'False
192|     FontUnderline = 0  'False
193|     Height        = 2235
194|     Left          = 6705
195|     MultiLine     = -1 'True
196|     ScrollBars    = 2  'Vertical
197|     TabIndex      = 25
198|     Text          = "SQL Statement to produce data displayed on DataBase Form"
199|     Top           = 3480
200|     Width         = 2700
201|   End
202|   Begin TextBox txtMapSQL
203|     FontBold      = 0  'False
204|     FontItalic    = 0  'False
205|     FontName      = "MS Sans Serif"
206|     FontSize      = 8.25
207|     FontStrikethru = 0  'False
208|     FontUnderline = 0  'False
209|     Height        = 2745
210|     Left          = 6705
211|     MultiLine     = -1 'True
212|     ScrollBars    = 2  'Vertical
213|     TabIndex      = 23
214|     Text          = "SQL statement to produce map data"
215|     Top           = 360
216|     Width         = 2700
```

```
End
Begin ComboBox cmbTblQryList
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 300
  Left          = 1425
  Style         = 2  'Dropdown List
  TabIndex      = 3
  Top           = 555
  Width         = 5145
End
Begin CommandButton cmdCancel
  Caption       = "&Cancel"
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 315
  Left          = 5300
  TabIndex      = 21
  Top           = 5400
  Width         = 1300
End
Begin CommandButton cmdOK
  Caption       = "&OK"
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 315
  Left          = 5300
  TabIndex      = 20
  Top           = 5040
  Width         = 1300
End
Begin ComboBox cmbFieldList
  DragIcon      = POLYOPT4.FRX:0302
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 300
  Left          = 1425
  Sorted        = -1  'True
```

```
271|     Style         = 2 'Dropdown List
272|     TabIndex      = 5
273|     Top           = 900
274|     Width         = 5145
275|   End
276|   Begin Label ACTION
277|     BackColor     = &H0000FFFF&
278|     ForeColor     = &H000000FF&
279|     Height        = 195
280|     Left          = 1260
281|     TabIndex      = 31
282|     Top           = 2040
283|     Visible       = 0 'False
284|     Width         = 1995
285|   End
286|   Begin Image imgFieldName
287|     Height        = 480
288|     Left          = 270
289|     Picture       = POLYOPT4.FRX:0604
290|     Top           = 750
291|     Width         = 480
292|   End
293|   Begin Label zzlblInterior
294|     Caption       = "Interior"
295|     FontBold      = -1 'True
296|     FontItalic    = 0 'False
297|     FontName      = "MS Sans Serif"
298|     FontSize      = 13.5
299|     FontStrikethru = 0 'False
300|     FontUnderline = 0 'False
301|     Height        = 285
302|     Left          = 3870
303|     TabIndex      = 26
304|     Top           = 2610
305|     Width         = 1005
306|   End
307|   Begin Image zzimgOutline
308|     Height        = 480
309|     Left          = 1710
310|     Picture       = POLYOPT4.FRX:0906
311|     Top           = 2520
312|     Width         = 480
313|   End
314|   Begin Label zzlblOutline
315|     Caption       = "Outline"
316|     FontBold      = -1 'True
317|     FontItalic    = 0 'False
318|     FontName      = "MS Sans Serif"
319|     FontSize      = 13.5
320|     FontStrikethru = 0 'False
321|     FontUnderline = 0 'False
322|     Height        = 285
323|     Left          = 630
324|     TabIndex      = 30
```

```
325|     Top          = 2610
326|     Width        = 1005
327|   End
328|   Begin Label zlblOutlineColor
329|     Caption      = "Outline &Color"
330|     FontBold     = 0  'False
331|     FontItalic   = 0  'False
332|     FontName     = "MS Sans Serif"
333|     FontSize     = 8.25
334|     FontStrikethru = 0  'False
335|     FontUnderline  = 0  'False
336|     Height       = 495
337|     Left         = 90
338|     TabIndex     = 14
339|     Top          = 3600
340|     Width        = 645
341|   End
342|   Begin Label zlblFillStyle
343|     Caption      = "&Fill Style"
344|     FontBold     = 0  'False
345|     FontItalic   = 0  'False
346|     FontName     = "MS Sans Serif"
347|     FontSize     = 8.25
348|     FontStrikethru = 0  'False
349|     FontUnderline  = 0  'False
350|     Height       = 465
351|     Left         = 3480
352|     TabIndex     = 16
353|     Top          = 3060
354|     Width        = 465
355|   End
356|   Begin Label zlblFillColor
357|     Caption      = "Fill Co&lor"
358|     FontBold     = 0  'False
359|     FontItalic   = 0  'False
360|     FontName     = "MS Sans Serif"
361|     FontSize     = 8.25
362|     FontStrikethru = 0  'False
363|     FontUnderline  = 0  'False
364|     Height       = 465
365|     Left         = 3480
366|     TabIndex     = 18
367|     Top          = 3600
368|     Width        = 555
369|   End
370|   Begin Label zzzlblZSize
371|     Caption      = "Outline Widt&h"
372|     FontBold     = 0  'False
373|     FontItalic   = 0  'False
374|     FontName     = "MS Sans Serif"
375|     FontSize     = 8.25
376|     FontStrikethru = 0  'False
377|     FontUnderline  = 0  'False
378|     Height       = 405
```

```
379|     Left        = 90
380|     TabIndex     = 12
381|     Top         = 3060
382|     Width       = 675
383|   End
384|   Begin Line Line2
385|     BorderWidth   = 2
386|     X1          = 6660
387|     X2          = 0
388|     Y1          = 2520
389|     Y2          = 2520
390|   End
391|   Begin Line Line1
392|     BorderWidth   = 3
393|     X1          = 6600
394|     X2          = 0
395|     Y1          = 1260
396|     Y2          = 1260
397|   End
398|   Begin Image zIMGExample
399|     Height      = 480
400|     Left        = 6030
401|     Picture     = POLYOPT4.FRX:0C08
402|     Top         = 30
403|     Width       = 480
404|   End
405|   Begin Label lblThemeName
406|     Caption     = "T&itle for this theme"
407|     FontBold    = 0  'False
408|     FontItalic  = 0  'False
409|     FontName    = "MS Sans Serif"
410|     FontSize    = 8.25
411|     FontStrikethru = 0  'False
412|     FontUnderline  = 0  'False
413|     Height      = 225
414|     Left        = 30
415|     TabIndex    = 0
416|     Top         = 90
417|     Width       = 1830
418|   End
419|   Begin Label zzzzlblCriteia
420|     Caption     = "C&riteria"
421|     FontBold    = 0  'False
422|     FontItalic  = 0  'False
423|     FontName    = "MS Sans Serif"
424|     FontSize    = 8.25
425|     FontStrikethru = 0  'False
426|     FontUnderline  = 0  'False
427|     Height      = 255
428|     Left        = 120
429|     TabIndex    = 10
430|     Top         = 1740
431|     Width       = 615
432|   End
```

```
433|   Begin Label zzzzlblDBFormSQL
434|     Caption       = "Spatial Query SQL"
435|     FontBold      = 0  'False
436|     FontItalic    = 0  'False
437|     FontName      = "MS Sans Serif"
438|     FontSize      = 8.25
439|     FontStrikethru = 0  'False
440|     FontUnderline = 0  'False
441|     Height        = 255
442|     Left          = 6705
443|     TabIndex      = 24
444|     Top           = 3180
445|     Width         = 2700
446|   End
447|   Begin Label zzzzlblMapTileSQL
448|     Caption       = "Map Polygon SQL"
449|     FontBold      = 0  'False
450|     FontItalic    = 0  'False
451|     FontName      = "MS Sans Serif"
452|     FontSize      = 8.25
453|     FontStrikethru = 0  'False
454|     FontUnderline = 0  'False
455|     Height        = 255
456|     Left          = 6705
457|     TabIndex      = 22
458|     Top           = 120
459|     Width         = 2700
460|   End
461|   Begin Label zzzlblFields
462|     Caption       = "&Fields"
463|     FontBold      = 0  'False
464|     FontItalic    = 0  'False
465|     FontName      = "MS Sans Serif"
466|     FontSize      = 8.25
467|     FontStrikethru = 0  'False
468|     FontUnderline = 0  'False
469|     Height        = 285
470|     Left          = 810
471|     TabIndex      = 4
472|     Top           = 900
473|     Width         = 555
474|   End
475|   Begin Label zzzlblTables
476|     Caption       = "&Tables/Queries"
477|     FontBold      = 0  'False
478|     FontItalic    = 0  'False
479|     FontName      = "MS Sans Serif"
480|     FontSize      = 8.25
481|     FontStrikethru = 0  'False
482|     FontUnderline = 0  'False
483|     Height        = 285
484|     Left          = 30
485|     TabIndex      = 2
486|     Top           = 540
```

```
487|     Width         =  1335
488|   End
489|   Begin Label zzzlblName
490|     Caption       =  "&Unique Number"
491|     FontBold      =  0  'False
492|     FontItalic    =  0  'False
493|     FontName      =  "MS Sans Serif"
494|     FontSize      =  8.25
495|     FontStrikethru =  0  'False
496|     FontUnderline =  0  'False
497|     Height        =  405
498|     Left          =  120
499|     TabIndex      =  6
500|     Top           =  1320
501|     Width         =  615
502|   End
503|   Begin Label zzzlblX
504|     Alignment     =  1  'Right Justify
505|     Caption       =  "&XY"
506|     FontBold      =  0  'False
507|     FontItalic    =  0  'False
508|     FontName      =  "MS Sans Serif"
509|     FontSize      =  8.25
510|     FontStrikethru =  0  'False
511|     FontUnderline =  0  'False
512|     Height        =  255
513|     Left          =  3600
514|     TabIndex      =  8
515|     Top           =  1350
516|     Width         =  360
517|   End
518|   Begin Image zzimgInterior
519|     Height        =  480
520|     Left          =  4860
521|     Picture       =  POLYOPT4.FRX:0F0A
522|     Top           =  2520
523|     Width         =  480
524|   End
525| End
526| Option Explicit
527| Option Compare Text
528| Dim flgValidSyntax As Integer, ListOfFieldsForDBForm As String
529| Dim FlagActivateDragDrop As Integer
530| Dim tblPointMap As Table
531|
532| Dim FieldsInCurrentQT() As String, FieldListWBrackets() As String
533| Dim nStoredMaps As Long, ReturnVal As Long
534|
535| Dim intOutlineWidth As Integer, lngOutlineColor As Long
536|
537| Dim flgIgnoreMapSQLChange As Long
538|
539| Sub ACTION_Change ()
540|   Dim tmp As String
```

```
541|   tmp = UCase$(Me!ACTION.Caption)
542|   Select Case tmp
543|    Case "Decode MapSQL"
544|       tmp = Me!txtMapSQL
545|       Call DecodeMapSQL(tmp)
546|    Case "wait" 'do nothing
547|    Case "refresh"
548|       Call Form_Load
549|    Case Else
550|   End Select
551| End Sub
552|
553| Sub cmb_zqFeatureID_Change ()
554|  SetMapSQL
555|  Call DBFormSQLAddField("")
556| End Sub
557|
558| Sub cmb_zqFeatureID_Click ()
559|   SetMapSQL
560|   Call DBFormSQLAddField("")
561| End Sub
562|
563| Sub cmb_zqFeatureID_DragDrop (Source As Control, X As Single, Y As Single)
564|    cmb_zqFeatureID = "[" & Source & "]"
565| 'Drops the selected text over the entire field, for use on
566| ' NAME only, because we'll need this for matching and it
567| ' NAME should never be a calculated field, at least not easily.
568|
569| End Sub
570|
571| Sub cmb_zqFillColor_Change ()
572|     SetMapSQL
573| End Sub
574|
575| Sub cmb_zqFillColor_Click ()
576|    SetMapSQL
577| End Sub
578|
579| Sub cmb_zqFillColor_DragDrop (Source As Control, X As Single, Y As Single)
580|  cmb_zqFillColor.AddItem Mid$(cmb_zqFillColor, 1, cmb_zqFillColor.SelStart) & " " & "["
     & Source & "]" & " " & Mid$(cmb_zqFillColor, cmb_zqFillColor.SelStart +
     cmb_zqFillColor.SelLength + 1, 32768)
581|  cmb_zqFillColor.ListIndex = cmb_zqFillColor.ListCount - 1
582| End Sub
583|
584| Sub cmb_zqFillStyle_Change ()
585|    SetMapSQL
586| End Sub
587|
588| Sub cmb_zqFillStyle_Click ()
589|    SetMapSQL
590| End Sub
591|
592| Sub cmb_zqFillStyle_DragDrop (Source As Control, X As Single, Y As Single)
```

```
593|  cmb_zqFillStyle.AddItem Mid$(cmb_zqFillStyle, 1, cmb_zqFillStyle.SelStart) & " " & "["
      & Source & "]" & " " & Mid$(cmb_zqFillStyle, cmb_zqFillStyle.SelStart +
      cmb_zqFillStyle.SelLength + 1, 32768)
594|  cmb_zqFillStyle.ListIndex = cmb_zqFillStyle.ListCount - 1
595|
596|  End Sub
597|
598|  Sub cmb_zqOutlineColor_Change ()
599|   SetMapSQL
600|  End Sub
601|
602|  Sub cmb_zqOutlineColor_Click ()
603|     SetMapSQL
604|  End Sub
605|
606|  Sub cmb_zqOutlineColor_DragDrop (Source As Control, X As Single, Y As Single)
607|  cmb_zqOutlineColor.AddItem Mid$(cmb_zqOutlineColor, 1,
      cmb_zqOutlineColor.SelStart) & " " & "[" & Source & "]" & " " &
      Mid$(cmb_zqOutlineColor, cmb_zqOutlineColor.SelStart +
      cmb_zqOutlineColor.SelLength + 1, 32768)
608|  cmb_zqOutlineColor.ListIndex = cmb_zqOutlineColor.ListCount - 1
609|  End Sub
610|
611|  Sub cmb_zqOutlineWidth_Change ()
612|  SetMapSQL
613|  End Sub
614|
615|  Sub cmb_zqOutlineWidth_Click ()
616|    SetMapSQL
617|  End Sub
618|
619|  Sub cmb_zqOutlineWidth_DragDrop (Source As Control, X As Single, Y As Single)
620|  cmb_zqOutlineWidth.AddItem Mid$(cmb_zqOutlineWidth, 1,
      cmb_zqOutlineWidth.SelStart) & " " & "[" & Source & "]" & " " &
      Mid$(cmb_zqOutlineWidth, cmb_zqOutlineWidth.SelStart +
      cmb_zqOutlineWidth.SelLength + 1, 32768)
621|  cmb_zqOutlineWidth.ListIndex = cmb_zqOutlineWidth.ListCount - 1
622|  End Sub
623|
624|  Sub cmb_zqPolygonOutline_Change ()
625|  SetMapSQL
626|  End Sub
627|
628|  Sub cmb_zqPolygonOutline_Click ()
629|   SetMapSQL
630|  End Sub
631|
632|  Sub cmbFieldList_KeyDown (KeyCode As Integer, Shift As Integer)
633|    If KeyCode = KEY_INSERT And Shift = 2 Then 'CTRL-Insert?
634|      clipboard.Clear
635|      clipboard.SetText ("[" & cmbFieldList & "]")
636|    End If
637|  End Sub
638|
```

```
639| Sub cmbTblQryList_Click ()
640|
641|  If Me!ACTION.Caption = "wait" Then Exit Sub
642|  If Not flgIgnoreMapSQLChange Then
643|    'For usual case, set the SpatialQuerySQL and the theme name
644|    txtSpatialQuerySQL = "SELECT * FROM " &
     AddBracketsAsNeeded(CStr(cmbTblQryList))
645|    ListOfFieldsForDBForm = "*"
646|    txtThemeName = "POLYGONS: " & cmbTblQryList & "  " & Now
647|  End If
648|
649|  Dim TableName As String
650|  On Error GoTo ERRcmbTblQryList_Click
651|  TableName = cmbTblQryList
652|  Set snpFieldList = MapDB.CreateSnapshot(TableName)
653|  nFields = snpFieldList.Fields.Count
654|  cmbFieldList.Clear
655|  cmb_zqPolygonOutline.Clear
656|  ReDim FieldsInCurrentQT(1 To nFields), FieldListWBrackets(1 To nFields)
657|
658|  For i = 1 To nFields
659|    If snpFieldList.Fields(i - 1).Type = DB_LONGBINARY Then
660|       'Put XYBLOBs (stored in OLE/BinaryLargeObjectFields) in the
661|       ' outline list box
662|       cmb_zqPolygonOutline.AddItem "[" & snpFieldList.Fields(i - 1).Name & "]"
663|      Else
664|       'Put everything else in the general field list.
665|       cmbFieldList.AddItem snpFieldList.Fields(i - 1).Name
666|       FieldListWBrackets(i) = "[" & snpFieldList.Fields(i - 1).Name & "]"
667|    End If
668|    FieldsInCurrentQT(i) = snpFieldList.Fields(i - 1).Name
669|  Next i
670|  snpFieldList.Close
671|  cmbFieldList.ListIndex = 0
672|  txtCriteria = ""
673|
674|   If cmb_zqPolygonOutline.ListCount = 0 Then
675|    Beep
676|    MsgBox "This field contains no line data." & CRLF & "Please choose a new query or
     table that does."
677|    cmbTblQryList.SetFocus
678|    SendKeys "%{DOWN}"
679|    Exit Sub
680|   Else
681|    cmb_zqPolygonOutline.ListIndex = 0
682|   End If
683|
684|  cmb_zqFeatureID.Clear
685|  FillCMBFromSGSTLST FieldListWBrackets(), FeatureNameLookFor,
     FeatureNameShowAlways, cmb_zqFeatureID
686|  cmb_zqFeatureID.ListIndex = 0
687|
688|  cmb_zqFillColor.Clear
689|  FillCMBFromSGSTLST FieldListWBrackets(), ColorLookFor, ColorShowAlways,
```

```
      cmb_zqFillColor
 690  cmb_zqFillColor.ListIndex = 0
 691
 692  cmb_zqOutlineColor.Clear
 693  FillCMBFromSGSTLST FieldListWBrackets(), ColorLookFor, ColorShowAlways,
      cmb_zqOutlineColor
 694  cmb_zqOutlineColor.ListIndex = 0
 695
 696  cmb_zqOutlineWidth.Clear
 697  FillCMBFromSGSTLST FieldListWBrackets(), LineWidthLookFor, "0<=>No_Outline " &
      LineWidthShowAlways, cmb_zqOutlineWidth
 698  cmb_zqOutlineWidth.ListIndex = 0
 699
 700  cmb_zqFillStyle.Clear
 701  FillCMBFromSGSTLST FieldListWBrackets(), FillStyleLookFor, FillStyleShowAlways,
      cmb_zqFillStyle
 702  cmb_zqFillStyle.ListIndex = 0
 703
 704
 705  FlagActivateDragDrop = 0
 706  Exit Sub
 707  ERRcmbTblQryList_Click:
 708   MsgBox "Error #" & Err & " occurred in cmbTblQryList_Click." & CRLF & Error
 709   Err = 0
 710   Exit Sub
 711   Resume Next
 712
 713  End Sub
 714
 715  Sub cmdCancel_Click ()
 716  Me.Hide
 717  End Sub
 718
 719  Sub cmdOK_Click ()
 720  If cmbTblQryList = "" Then
 721     MsgBox "A query or table must be selected before trying to display its results." & CRLF
      & "Or you can hit CANCEL to not add a point theme right now and get out of here."
 722     cmbTblQryList.SetFocus
 723     SendKeys "%{DOWN}"
 724     Exit Sub
 725  End If
 726
 727
 728
 729         'SetMapSQL
 730     '   Call DBFormSQLAddField("")
 731
 732    Me!cmdOK.SetFocus
 733  Me.Hide
 734    Me!cmdOK.FontItalic = True
 735
 736
 737  End Sub
 738
```

```
739  Sub DBFormSQLAddField (NewField As String)
740  If flgIgnoreMapSQLChange Then Exit Sub
741  If NewField <> "" Then
742   If ListOfFieldsForDBForm = "*" Then
743     ListOfFieldsForDBForm = "[" & NewField & "]"
744    Else
745     ListOfFieldsForDBForm = ListOfFieldsForDBForm & ", [" & NewField & "]"
746   End If
747  End If
748  txtSpatialQuerySQL = "SELECT " & ListOfFieldsForDBForm & ", " & cmb_zqFeatureID &
     " AS zqFeatureID FROM " & "[" & cmbTblQryList & "]"
749  End Sub
750
751  Sub DecodeMapSQL (strSQLIn As String)
752   Dim i As Long, h As Integer, strTemp As String
753   strSQLIn = Trim$(strSQLIn)
754   If strSQLIn = "" Then Exit Sub
755
756
757
758  'If the MapSQL was changed by another form or by the user,
759  '   set the comboboxes to reflect the changes as best possible.
760   Dim strParameters  As String, strFields As String, strFrom As String, strWhere As
     String, strGroupBy As String, strHaving As String, strOrderBy As String
761
762
763
764   If SplitSQLStatement(strSQLIn, strParameters, strFields, strFrom, strWhere,
     strGroupBy, strHaving, strOrderBy) = False Then Exit Sub
765
766   If strParameters <> "" Then
767    MsgBox "WHEAT programs do not run parameter queries.  Please choose another
     map SQL string."
768    Exit Sub
769   End If
770   flgIgnoreMapSQLChange = True
771   Dim strProblems As String
772
773   strFrom = Strip2Brackets(strFrom)
774  'Set the source table or query
775   i = IndexOfStringInCOMBOBOX(strFrom, Me!cmbTblQryList)
776   Me!cmbTblQryList.ListIndex = i
777    DoEvents 'Let the click event occur for the QT List
778
779  'Get field assignments
780   Dim strRawName() As String, strAsName() As String, nFields As Long, nF As Long
781   Call PullFieldNamesFromString(strFields, strRawName(), strAsName(), nFields)
782   'For i = 1 To nFields
783   '   dDebug.Print strRawName(i), , strAsName(i)
784   'Next i
785
786    'Set XY Coordinate field
787   For nF = 1 To nFields
788    If strAsName(nF) = "zqPolygonOutline" Then
```

```
789|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqPolygonOutline)
790|      If i <> -1 Then
791|        Me.cmb_zqPolygonOutline.ListIndex = i
792|       Else
793|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
      for the XY-coordinate chain, but this routine cannot find it.  You will have to select this
      one yourself." & CRLF
794|      End If
795|      Exit For
796|    End If
797|   Next nF
798|
799|    'Set FeatureID field
800|   For nF = 1 To nFields
801|    If strAsName(nF) = "zqFeatureID" Then
802|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFeatureID)
803|      If i <> -1 Then
804|        Me.cmb_zqFeatureID.ListIndex = i
805|       Else
806|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
      for feature ID, but this routine cannot find it.  You will have to select this one yourself." &
      CRLF
807|      End If
808|      Exit For
809|    End If
810|   Next nF
811|
812|    'Set line width field
813|   For nF = 1 To nFields
814|    If strAsName(nF) = "zqOutlineWidth" Then
815|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqOutlineWidth)
816|      If i <> -1 Then
817|        Me.cmb_zqOutlineWidth.ListIndex = i
818|       Else
819|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
      for line width, but this routine cannot find it.  You will have to select this one yourself." &
      CRLF
820|      End If
821|      Exit For
822|    End If
823|   Next nF
824|
825|
826|     'outline color
827|   For nF = 1 To nFields
828|    If strAsName(nF) = "zqOutlineColor" Then
829|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqOutlineColor)
830|      If i <> -1 Then
831|        Me.cmb_zqOutlineColor.ListIndex = i
832|       Else
833|        i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqOutlineColor)
834|        If i <> -1 Then
835|          Me.cmb_zqOutlineColor.ListIndex = i
836|         Else
```

```
837|            strProblems = strProblems & "This theme originally used " & strRawName(nF) &
    " for the color, but this routine cannot find it.  You will have to select this one yourself." &
    CRLF
838|          End If
839|        End If
840|        Exit For
841|      End If
842|    Next nF
843|
844|     'fill color
845|    For nF = 1 To nFields
846|     If strAsName(nF) = "zqFillColor" Then
847|       i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFillColor)
848|       If i <> -1 Then
849|         Me.cmb_zqFillColor.ListIndex = i
850|        Else
851|         i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFillColor)
852|         If i <> -1 Then
853|           Me.cmb_zqFillColor.ListIndex = i
854|          Else
855|            strProblems = strProblems & "This theme originally used " & strRawName(nF) &
    " for the color, but this routine cannot find it.  You will have to select this one yourself." &
    CRLF
856|          End If
857|        End If
858|        Exit For
859|      End If
860|    Next nF
861|
862|
863|
864|
865|       'The fill style
866|    For nF = 1 To nFields
867|     If strAsName(nF) = "zqFillStyle" Then
868|       i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFillStyle)
869|       If i <> -1 Then
870|         Me.cmb_zqFillStyle.ListIndex = i
871|        Else
872|         i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFillStyle)
873|         If i <> -1 Then
874|           Me.cmb_zqFillStyle.ListIndex = i
875|          Else
876|
877|            strProblems = strProblems & "This theme originally used " & strRawName(nF) &
    " for the line style, but this routine cannot find it.  You will have to select this one
    yourself." & CRLF
878|          End If
879|        End If
880|        Exit For
881|      End If
882|    Next nF
883|
884|     txtCriteria = strWhere
```

```
 885|
 886|
 887|   flgIgnoreMapSQLChange = False
 888|    If strProblems <> "" Then
 889|      Beep
 890|      MsgBox strProblems
 891|    End If
 892|
 893| End Sub
 894|
 895| Sub Form_Load ()
 896| FlagActivateDragDrop = 0
 897| Me!cmbTblQryList.Clear
 898| For i = 1 To nTblQrys
 899|  Me!cmbTblQryList.AddItem strTblQryList(i)
 900| Next i
 901| End Sub
 902|
 903| Sub Form_Paint ()
 904|   cmdOK.FontItalic = False
 905|   If cmbTblQryList.ListIndex = -1 Then cmbTblQryList.ListIndex = 0
 906|   ShowSample
 907| End Sub
 908|
 909| Sub imgFieldName_Click ()
 910|    cmbFieldList.Drag
 911| End Sub
 912|
 913| Sub lblThemeName_DragDrop (Source As Control, X As Single, Y As Single)
 914|      txtThemeName = Source
 915| End Sub
 916|
 917| Sub picSample_Paint ()
 918|  ShowSample
 919| End Sub
 920|
 921| Sub SetMapSQL ()
 922| If flgIgnoreMapSQLChange Then Exit Sub
 923| ShowSample
 924| Static TempSQL As String
 925|
 926| TempSQL = "SELECT DISTINCTROW "
 927| ' Assemble a new MapSQL statement.
 928| TempSQL = TempSQL & AddVarNameAsNeeded(cmb_zqPolygonOutline,
      "zqPolygonOutline") & " , "
 929| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqFillColor.Text), "zqFillColor") & " , "
 930| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqFillStyle.Text), "zqFillStyle") & " , "
 931| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqOutlineColor.Text),
      "zqOutlineColor") & " , "
 932| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqOutlineWidth.Text),
```

```
        "zqOutlineWidth") & " , "
 933|   TempSQL = TempSQL &
        AddVarNameAsNeeded(GetGoodValueField(cmb_zqFeatureID.Text), "zqFeatureID") &
        " "
 934|
 935|   TempSQL = TempSQL & " FROM [" & cmbTblQryList & "]  "
 936|
 937|   'Do not add any criteria of your own.
 938|   If (txtCriteria = "") Then
 939|      TempSQL = TempSQL
 940|     Else
 941|      TempSQL = TempSQL & "  WHERE (" & txtCriteria & ") "
 942|   End If
 943|
 944|
 945|   'Build the Order By clause
 946|   Dim strOrderBy As String, Temp As String    'changed 950118
 947|     strOrderBy = ""
 948|
 949|     Temp = GetFieldForOrderBy(cmb_zqOutlineWidth.Text)
 950|     If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
 951|
 952|     Temp = GetFieldForOrderBy(cmb_zqFillStyle.Text)
 953|     If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
 954|
 955|     Temp = GetFieldForOrderBy(cmb_zqOutlineColor.Text)
 956|     If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
 957|
 958|     Temp = GetFieldForOrderBy(cmb_zqFillColor.Text)
 959|     If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
 960|
 961|   If strOrderBy <> "" Then
 962|       strOrderBy = " ORDER BY " & Left$(strOrderBy, Len(strOrderBy) - 2)
 963|     End If
 964|
 965|     TempSQL = TempSQL & strOrderBy & " ; "
 966|
 967|
 968|   ' Now put the SQL statement into the appropriate text box.
 969|   flgIgnoreMapSQLChange = True
 970|     txtMapSQL = TempSQL
 971|   flgIgnoreMapSQLChange = False
 972|
 973|   End Sub
 974|
 975|   Sub ShowSample ()
 976|    ReDim Triangle(1 To 3) As APIPOINT
 977|    Me.picSample.Cls
 978|    Triangle(1).X = 10
 979|    Triangle(1).Y = 10
 980|
 981|    Triangle(2).X = 70
 982|    Triangle(2).Y = 10
 983|
```

```
    Triangle(3).X = 40
    Triangle(3).Y = 70
On Error Resume Next
    picSample.DrawWidth = Val(cmb_zqOutlineWidth)
    picSample.ForeColor = Val(cmb_zqOutlineColor)
    picSample.FillStyle = Val(cmb_zqFillStyle)
    picSample.FillColor = Val(cmb_zqFillColor)

    ReturnVal = Polyline(Me.picSample.hDC, Triangle(1), 3)
    ReturnVal = Polygon(Me.picSample.hDC, Triangle(1), 3)
End Sub

Sub txtCriteria_Change ()
    SetMapSQL
End Sub

Sub txtCriteria_DragDrop (Source As Control, X As Single, Y As Single)
'txtCriteria = Mid$(txtCriteria, 1, txtCriteria.SelStart) & " " &
AddBracketsAsNeeded(CStr(Source)) & " " & Mid$(txtCriteria, txtCriteria.SelStart +
txtCriteria.SelLength + 1, 32768)
txtCriteria = Mid$(txtCriteria, 1, txtCriteria.SelStart) & " [" & Source & "] " &
Mid$(txtCriteria, txtCriteria.SelStart + txtCriteria.SelLength + 1, 32768)
'Drops the selected text at the current insertion point
End Sub

Sub txtMapSQL_KeyPress (KeyAscii As Integer)

    If flgIgnoreMapSQLChange Then Exit Sub
    If KeyAscii <> 13 Then Exit Sub
    KeyAscii = 0'Swallow the return key
     Dim strSQL As String
     strSQL = txtMapSQL
     Call DecodeMapSQL(strSQL)

End Sub

Sub txtSpatialQuerySQL_DragDrop (Source As Control, X As Single, Y As Single)
Call DBFormSQLAddField(CStr(Source))
End Sub

Sub txtSpatialQuerySQL_KeyPress (KeyAscii As Integer)
If KeyAscii = 13 Then
    KeyAscii = 0
    'Other decoding of the SQL statement should go here
End If

End Sub

Sub txtThemeName_DragDrop (Source As Control, X As Single, Y As Single)
     txtThemeName = Mid$(txtThemeName, 1, txtThemeName.SelStart) & " " & "[" &
Source & "]" & " " & Mid$(txtThemeName, txtThemeName.SelStart +
txtThemeName.SelLength + 1, 32768)
End Sub

```

```
Sub zlblFillColor_Click ()
  Dim ColorWas As Long, NewColor As Long
  ColorWas = Val(cmb_zqFillColor)
  NewColor = GetColorWithCmnDialog(ColorWas)
  If NewColor <> ColorWas Then AddAndHighlightItem Format$(NewColor),
cmb_zqFillColor

End Sub

Sub zlblFillColor_DragDrop (Source As Control, X As Single, Y As Single)
 cmb_zqFillColor.AddItem "[" & Source & "]"
 cmb_zqFillColor.ListIndex = cmb_zqFillColor.ListCount - 1

End Sub

Sub zlblFillStyle_DragDrop (Source As Control, X As Single, Y As Single)
 cmb_zqFillStyle.AddItem "[" & Source & "]"
 cmb_zqFillStyle.ListIndex = cmb_zqFillStyle.ListCount - 1
End Sub

Sub zlblOutlineColor_Click ()
  Dim ColorWas As Long, NewColor As Long
  ColorWas = Val(cmb_zqOutlineColor)
  NewColor = GetColorWithCmnDialog(ColorWas)
  If NewColor <> ColorWas Then AddAndHighlightItem Format$(NewColor),
cmb_zqOutlineColor

End Sub

Sub zlblOutlineColor_DragDrop (Source As Control, X As Single, Y As Single)
 cmb_zqOutlineColor.AddItem "[" & Source & "]"
 cmb_zqOutlineColor.ListIndex = cmb_zqOutlineColor.ListCount - 1
End Sub

Sub zzzlblFields_Click ()
  cmbFieldList.Drag
End Sub

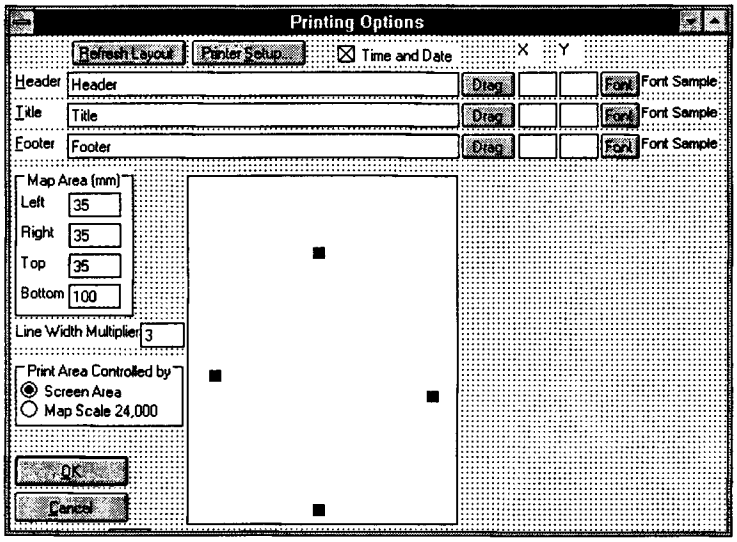
Sub zzzlblName_DragDrop (Source As Control, X As Single, Y As Single)
cmb_zqFeatureID.AddItem "[" & Source & "]"'Sets all of the following Z-field to whatever
was dropped on it
cmb_zqFeatureID.ListIndex = cmb_zqFeatureID.ListCount - 1

End Sub

Sub zzzlblZSize_DragDrop (Source As Control, X As Single, Y As Single)
 cmb_zqOutlineWidth.AddItem "[" & Source & "]"'Sets all of the following Z-field to
whatever was dropped on it
 cmb_zqOutlineWidth.ListIndex = cmb_zqOutlineWidth.ListCount - 1
End Sub

Sub zzzzlblCriteia_DragDrop (Source As Control, X As Single, Y As Single)
'txtCriteria = AddBracketsAsNeeded(CStr(Source))
txtCriteria = "[" & Source & "]"
```

```
1083|  End Sub
1084|
1085|  Sub zzzzlblDBFormSQL_DragDrop (Source As Control, X As Single, Y As Single)
1086|  Call DBFormSQLAddField(CStr(Source))
1087|  End Sub
1088|
```


Printing Options
Refresh Layout
Printer Setup...
Time and Date
X
Y
Header
Header
Drag
Font
Font Sample
Title
Title
Drag
Font
Font Sample
Footer
Footer
Drag
Font
Font Sample
Map Area (mm)
Left
35
Right
35
Top
35
Bottom
100
Line Width Multiplier
3
Print Area Controlled by
Screen Area
Map Scale 24,000
OK
Cancel

```
VERSION 2.00
Begin Form frmPrinterOptions
  Caption        = "Printing Options"
  ClientHeight   = 5790
  ClientLeft     = 330
  ClientTop      = 765
  ClientWidth    = 8310
  ControlBox     = 0  'False
  Height         = 6195
  Left           = 270
  LinkTopic      = "Form1"
  ScaleHeight    = 102.129
  ScaleMode      = 6  'Millimeter
  ScaleWidth     = 146.579
  Top            = 420
  Width          = 8430
  Begin Frame zfrmAreaControl
    Caption        = "Print Area Controlled by"
    FontBold       = 0  'False
    FontItalic     = 0  'False
    FontName       = "MS Sans Serif"
    FontSize       = 8.25
    FontStrikethru = 0  'False
    FontUnderline  = 0  'False
    Height         = 795
    Left           = 60
    TabIndex       = 44
    Top            = 3780
    Width          = 1935
    Begin TextBox txtPrintScale
      BorderStyle    = 0  'None
      FontBold       = 0  'False
      FontItalic     = 0  'False
      FontName       = "MS Sans Serif"
      FontSize       = 8.25
      FontStrikethru = 0  'False
      FontUnderline  = 0  'False
      Height         = 240
      Left           = 1140
      TabIndex       = 47
      Text           = "24,000"
      Top            = 510
      Width          = 735
    End
    Begin OptionButton optScreenArea
      Caption        = "Screen Area"
      FontBold       = 0  'False
      FontItalic     = 0  'False
      FontName       = "MS Sans Serif"
      FontSize       = 8.25
      FontStrikethru = 0  'False
      FontUnderline  = 0  'False
      Height         = 255
      Left           = 60
```

```
55|       TabIndex      = 46
56|       Top           = 240
57|       Value         = -1 'True
58|       Width         = 1455
59|     End
60|     Begin OptionButton optMapScale
61|       Caption       = "Map Scale"
62|       FontBold      = 0  'False
63|       FontItalic    = 0  'False
64|       FontName      = "MS Sans Serif"
65|       FontSize      = 8.25
66|       FontStrikethru = 0  'False
67|       FontUnderline = 0  'False
68|       Height        = 240
69|       Left          = 60
70|       TabIndex      = 45
71|       Top           = 480
72|       Width         = 1095
73|     End
74|   End
75|   Begin CheckBox chkTimeDate
76|     Caption       = "Time and Date"
77|     FontBold      = 0  'False
78|     FontItalic    = 0  'False
79|     FontName      = "MS Sans Serif"
80|     FontSize      = 8.25
81|     FontStrikethru = 0  'False
82|     FontUnderline = 0  'False
83|     Height        = 195
84|     Left          = 3780
85|     TabIndex      = 43
86|     Top           = 120
87|     Value         = 1  'Checked
88|     Width         = 1395
89|   End
90|   Begin TextBox txtWidthMultiplier
91|     Alignment     = 1  'Right Justify
92|     FontBold      = 0  'False
93|     FontItalic    = 0  'False
94|     FontName      = "MS Sans Serif"
95|     FontSize      = 8.25
96|     FontStrikethru = 0  'False
97|     FontUnderline = 0  'False
98|     Height        = 285
99|     Left          = 1500
100|    TabIndex      = 41
101|    Text          = "3"
102|    Top           = 3360
103|    Width         = 510
104|  End
105|  Begin CommandButton cmdDrag
106|    Caption       = "Drag"
107|    FontBold      = 0  'False
108|    FontItalic    = 0  'False
```

```
109|     FontName      = "MS Sans Serif"
110|     FontSize      = 8.25
111|     FontStrikethru = 0  'False
112|     FontUnderline  = 0  'False
113|     Height        = 285
114|     Index         = 2
115|     Left          = 5220
116|     TabIndex      = 4
117|     Top           = 1140
118|     Width         = 615
119|   End
120|   Begin CommandButton cmdFont
121|     Caption       = "Font"
122|     FontBold      = 0  'False
123|     FontItalic    = 0  'False
124|     FontName      = "MS Sans Serif"
125|     FontSize      = 8.25
126|     FontStrikethru = 0  'False
127|     FontUnderline  = 0  'False
128|     Height        = 283
129|     Index         = 2
130|     Left          = 6840
131|     TabIndex      = 5
132|     Top           = 1140
133|     Width         = 454
134|   End
135|   Begin TextBox txtMapBlurb
136|     DragIcon      = PRNTOPT3.FRX:0000
137|     FontBold      = 0  'False
138|     FontItalic    = 0  'False
139|     FontName      = "MS Sans Serif"
140|     FontSize      = 8.25
141|     FontStrikethru = 0  'False
142|     FontUnderline  = 0  'False
143|     Height        = 285
144|     Index         = 2
145|     Left          = 660
146|     TabIndex      = 6
147|     Text          = "Footer"
148|     Top           = 1140
149|     Width         = 4535
150|   End
151|   Begin TextBox txtBlurbY
152|     Alignment     = 1  'Right Justify
153|     FontBold      = 0  'False
154|     FontItalic    = 0  'False
155|     FontName      = "MS Sans Serif"
156|     FontSize      = 8.25
157|     FontStrikethru = 0  'False
158|     FontUnderline  = 0  'False
159|     Height        = 285
160|     Index         = 2
161|     Left          = 6360
162|     TabIndex      = 7
```

```
163|     Top          =  1140
164|     Width        =  454
165|   End
166|   Begin TextBox txtBlurbX
167|     Alignment    =  1  'Right Justify
168|     FontBold     =  0  'False
169|     FontItalic   =  0  'False
170|     FontName     =  "MS Sans Serif"
171|     FontSize     =  8.25
172|     FontStrikethru =  0  'False
173|     FontUnderline  =  0  'False
174|     Height       =  285
175|     Index        =  2
176|     Left         =  5880
177|     TabIndex     =  8
178|     Top          =  1140
179|     Width        =  454
180|   End
181|   Begin CommandButton cmdDrag
182|     Caption      =  "Drag"
183|     FontBold     =  0  'False
184|     FontItalic   =  0  'False
185|     FontName     =  "MS Sans Serif"
186|     FontSize     =  8.25
187|     FontStrikethru =  0  'False
188|     FontUnderline  =  0  'False
189|     Height       =  285
190|     Index        =  1
191|     Left         =  5220
192|     TabIndex     =  40
193|     Top          =  780
194|     Width        =  615
195|   End
196|   Begin CommandButton cmdFont
197|     Caption      =  "Font"
198|     FontBold     =  0  'False
199|     FontItalic   =  0  'False
200|     FontName     =  "MS Sans Serif"
201|     FontSize     =  8.25
202|     FontStrikethru =  0  'False
203|     FontUnderline  =  0  'False
204|     Height       =  283
205|     Index        =  1
206|     Left         =  6840
207|     TabIndex     =  39
208|     Top          =  780
209|     Width        =  454
210|   End
211|   Begin TextBox txtMapBlurb
212|     DragIcon     =  PRNTOPT3.FRX:0302
213|     FontBold     =  0  'False
214|     FontItalic   =  0  'False
215|     FontName     =  "MS Sans Serif"
216|     FontSize     =  8.25
```

```
217|     FontStrikethru = 0  'False
218|     FontUnderline  = 0  'False
219|     Height       = 285
220|     Index        = 1
221|     Left         = 660
222|     TabIndex     = 37
223|     Text         = "Title"
224|     Top          = 780
225|     Width        = 4535
226|   End
227|   Begin TextBox txtBlurbY
228|     Alignment    = 1  'Right Justify
229|     FontBold     = 0  'False
230|     FontItalic   = 0  'False
231|     FontName     = "MS Sans Serif"
232|     FontSize     = 8.25
233|     FontStrikethru = 0  'False
234|     FontUnderline  = 0  'False
235|     Height       = 285
236|     Index        = 1
237|     Left         = 6360
238|     TabIndex     = 36
239|     Top          = 780
240|     Width        = 454
241|   End
242|   Begin TextBox txtBlurbX
243|     Alignment    = 1  'Right Justify
244|     FontBold     = 0  'False
245|     FontItalic   = 0  'False
246|     FontName     = "MS Sans Serif"
247|     FontSize     = 8.25
248|     FontStrikethru = 0  'False
249|     FontUnderline  = 0  'False
250|     Height       = 285
251|     Index        = 1
252|     Left         = 5880
253|     TabIndex     = 35
254|     Top          = 780
255|     Width        = 454
256|   End
257|   Begin CommandButton cmdDrag
258|     Caption      = "Drag"
259|     FontBold     = 0  'False
260|     FontItalic   = 0  'False
261|     FontName     = "MS Sans Serif"
262|     FontSize     = 8.25
263|     FontStrikethru = 0  'False
264|     FontUnderline  = 0  'False
265|     Height       = 285
266|     Index        = 0
267|     Left         = 5220
268|     TabIndex     = 34
269|     Top          = 420
270|     Width        = 615
```

```
271|   End
272|   Begin CommandButton cmdFont
273|     Caption       = "Font"
274|     FontBold      = 0  'False
275|     FontItalic    = 0  'False
276|     FontName      = "MS Sans Serif"
277|     FontSize      = 8.25
278|     FontStrikethru = 0  'False
279|     FontUnderline = 0  'False
280|     Height        = 283
281|     Index         = 0
282|     Left          = 6840
283|     TabIndex      = 33
284|     Top           = 420
285|     Width         = 454
286|   End
287|   Begin Frame frmMapArea
288|     Caption       = "Map Area (mm)"
289|     FontBold      = 0  'False
290|     FontItalic    = 0  'False
291|     FontName      = "MS Sans Serif"
292|     FontSize      = 8.25
293|     FontStrikethru = 0  'False
294|     FontUnderline = 0  'False
295|     Height        = 1755
296|     Left          = 60
297|     TabIndex      = 21
298|     Top           = 1530
299|     Width         = 1365
300|     Begin TextBox txtLeftMargin
301|       Alignment   = 1  'Right Justify
302|       FontBold    = 0  'False
303|       FontItalic  = 0  'False
304|       FontName    = "MS Sans Serif"
305|       FontSize    = 8.25
306|       FontStrikethru = 0  'False
307|       FontUnderline = 0  'False
308|       Height      = 285
309|       Left        = 620
310|       TabIndex    = 25
311|       Text        = "35"
312|       Top         = 300
313|       Width       = 600
314|     End
315|     Begin TextBox txtRightMargin
316|       Alignment   = 1  'Right Justify
317|       FontBold    = 0  'False
318|       FontItalic  = 0  'False
319|       FontName    = "MS Sans Serif"
320|       FontSize    = 8.25
321|       FontStrikethru = 0  'False
322|       FontUnderline = 0  'False
323|       Height      = 285
324|       Left        = 620
```

```
325|       TabIndex      = 24
326|       Text          = "35"
327|       Top           = 660
328|       Width         = 600
329|     End
330|     Begin TextBox txtTopMargin
331|       Alignment     = 1  'Right Justify
332|       FontBold      = 0  'False
333|       FontItalic    = 0  'False
334|       FontName      = "MS Sans Serif"
335|       FontSize      = 8.25
336|       FontStrikethru = 0  'False
337|       FontUnderline = 0  'False
338|       Height        = 285
339|       Left          = 620
340|       TabIndex      = 23
341|       Text          = "35"
342|       Top           = 1020
343|       Width         = 600
344|     End
345|     Begin TextBox txtBottomMargin
346|       Alignment     = 1  'Right Justify
347|       FontBold      = 0  'False
348|       FontItalic    = 0  'False
349|       FontName      = "MS Sans Serif"
350|       FontSize      = 8.25
351|       FontStrikethru = 0  'False
352|       FontUnderline = 0  'False
353|       Height        = 285
354|       Left          = 620
355|       TabIndex      = 22
356|       Text          = "100"
357|       Top           = 1380
358|       Width         = 600
359|     End
360|     Begin Label zzlblMapPos
361|       Caption       = "Left"
362|       FontBold      = 0  'False
363|       FontItalic    = 0  'False
364|       FontName      = "MS Sans Serif"
365|       FontSize      = 8.25
366|       FontStrikethru = 0  'False
367|       FontUnderline = 0  'False
368|       Height        = 255
369|       Index         = 0
370|       Left          = 60
371|       TabIndex      = 29
372|       Top           = 300
373|       Width         = 375
374|     End
375|     Begin Label zzlblMapPos
376|       Caption       = "Right"
377|       FontBold      = 0  'False
378|       FontItalic    = 0  'False
```

```
379|       FontName      = "MS Sans Serif"
380|       FontSize      = 8.25
381|       FontStrikethru = 0  'False
382|       FontUnderline  = 0  'False
383|       Height        = 255
384|       Index         = 1
385|       Left          = 60
386|       TabIndex      = 28
387|       Top           = 660
388|       Width         = 495
389|     End
390|     Begin Label zzlblMapPos
391|       Caption       = "Top"
392|       FontBold      = 0  'False
393|       FontItalic    = 0  'False
394|       FontName      = "MS Sans Serif"
395|       FontSize      = 8.25
396|       FontStrikethru = 0  'False
397|       FontUnderline  = 0  'False
398|       Height        = 255
399|       Index         = 2
400|       Left          = 60
401|       TabIndex      = 27
402|       Top           = 1020
403|       Width         = 495
404|     End
405|     Begin Label zzlblMapPos
406|       Caption       = "Bottom"
407|       FontBold      = 0  'False
408|       FontItalic    = 0  'False
409|       FontName      = "MS Sans Serif"
410|       FontSize      = 8.25
411|       FontStrikethru = 0  'False
412|       FontUnderline  = 0  'False
413|       Height        = 255
414|       Index         = 3
415|       Left          = 60
416|       TabIndex      = 26
417|       Top           = 1380
418|       Width         = 615
419|     End
420|   End
421|   Begin TextBox txtBlurbX
422|     Alignment       = 1  'Right Justify
423|     FontBold        = 0  'False
424|     FontItalic      = 0  'False
425|     FontName        = "MS Sans Serif"
426|     FontSize        = 8.25
427|     FontStrikethru  = 0  'False
428|     FontUnderline   = 0  'False
429|     Height          = 285
430|     Index           = 0
431|     Left            = 5880
432|     TabIndex        = 16
```

```
433|     Top           = 420
434|     Width         = 454
435|   End
436|   Begin TextBox txtBlurbY
437|     Alignment     = 1 'Right Justify
438|     FontBold      = 0 'False
439|     FontItalic    = 0 'False
440|     FontName      = "MS Sans Serif"
441|     FontSize      = 8.25
442|     FontStrikethru = 0 'False
443|     FontUnderline = 0 'False
444|     Height        = 285
445|     Index         = 0
446|     Left          = 6360
447|     TabIndex      = 15
448|     Top           = 420
449|     Width         = 454
450|   End
451|   Begin TextBox txtMapBlurb
452|     DragIcon      = PRNTOPT3.FRX:0604
453|     FontBold      = 0 'False
454|     FontItalic    = 0 'False
455|     FontName      = "MS Sans Serif"
456|     FontSize      = 8.25
457|     FontStrikethru = 0 'False
458|     FontUnderline = 0 'False
459|     Height        = 285
460|     Index         = 0
461|     Left          = 660
462|     TabIndex      = 14
463|     Text          = "Header"
464|     Top           = 420
465|     Width         = 4535
466|   End
467|   Begin CommandButton cmdRefreshLayout
468|     Caption       = "&Refresh Layout "
469|     FontBold      = 0 'False
470|     FontItalic    = 0 'False
471|     FontName      = "MS Sans Serif"
472|     FontSize      = 8.25
473|     FontStrikethru = 0 'False
474|     FontUnderline = 0 'False
475|     Height        = 283
476|     Left          = 720
477|     TabIndex      = 9
478|     Top           = 60
479|     Width         = 1304
480|   End
481|   Begin CommonDialog CMDialog1
482|     Left          = 1140
483|     Top           = 5760
484|   End
485|   Begin PictureBox picPageLayout
486|     Height        = 4095
```

```
487|     Left        =  2040
488|     ScaleHeight   =  71.702
489|     ScaleMode     =  6 'Millimeter
490|     ScaleWidth    =  54.769
491|     TabIndex      =  3
492|     Top         =  1620
493|     Width       =  3135
494|     Begin PictureBox picBottom
495|       BackColor     =  &H00000000&
496|       DragMode      =  1 'Automatic
497|       Height        =  142
498|       Left        =  1440
499|       ScaleHeight   =  105
500|       ScaleWidth    =  105
501|       TabIndex      =  13
502|       Top         =  3840
503|       Width       =  142
504|     End
505|     Begin PictureBox picRight
506|       BackColor     =  &H00000000&
507|       DragMode      =  1 'Automatic
508|       Height        =  142
509|       Left        =  2760
510|       ScaleHeight   =  105
511|       ScaleWidth    =  105
512|       TabIndex      =  12
513|       Top         =  2520
514|       Width       =  142
515|     End
516|     Begin PictureBox picTop
517|       BackColor     =  &H00000000&
518|       DragMode      =  1 'Automatic
519|       Height        =  142
520|       Left        =  1440
521|       ScaleHeight   =  105
522|       ScaleWidth    =  105
523|       TabIndex      =  11
524|       Top         =  840
525|       Width       =  142
526|     End
527|     Begin PictureBox picLeft
528|       BackColor     =  &H00000000&
529|       DragMode      =  1 'Automatic
530|       Height        =  142
531|       Left        =  240
532|       ScaleHeight   =  105
533|       ScaleWidth    =  105
534|       TabIndex      =  10
535|       Top         =  2280
536|       Width       =  142
537|     End
538|   End
539|   Begin CommandButton cmdPrinterSetup
540|     Caption       =  "Printer &Setup..."
```

```
541|     FontBold      = 0  'False
542|     FontItalic    = 0  'False
543|     FontName      = "MS Sans Serif"
544|     FontSize      = 8.25
545|     FontStrikethru = 0  'False
546|     FontUnderline = 0  'False
547|     Height        = 283
548|     Left          = 2100
549|     TabIndex      = 2
550|     Top           = 60
551|     Width         = 1304
552|   End
553|   Begin CommandButton cmdCancel
554|     Cancel        = -1 'True
555|     Caption       = "&Cancel"
556|     FontBold      = 0  'False
557|     FontItalic    = 0  'False
558|     FontName      = "MS Sans Serif"
559|     FontSize      = 8.25
560|     FontStrikethru = 0  'False
561|     FontUnderline = 0  'False
562|     Height        = 340
563|     Left          = 60
564|     TabIndex      = 1
565|     Top           = 5340
566|     Width         = 1304
567|   End
568|   Begin CommandButton cmdOK
569|     Caption       = "&OK"
570|     Default       = -1 'True
571|     FontBold      = 0  'False
572|     FontItalic    = 0  'False
573|     FontName      = "MS Sans Serif"
574|     FontSize      = 8.25
575|     FontStrikethru = 0  'False
576|     FontUnderline = 0  'False
577|     Height        = 340
578|     Left          = 60
579|     TabIndex      = 0
580|     Top           = 4920
581|     Width         = 1304
582|   End
583|   Begin Label zlblWidthMultiplier
584|     Caption       = "Line Width Multiplier"
585|     FontBold      = 0  'False
586|     FontItalic    = 0  'False
587|     FontName      = "MS Sans Serif"
588|     FontSize      = 8.25
589|     FontStrikethru = 0  'False
590|     FontUnderline = 0  'False
591|     Height        = 255
592|     Left          = 60
593|     TabIndex      = 42
594|     Top           = 3360
```

```
595|     Width         =  1815
596|   End
597|   Begin Label lblSpecimen
598|     AutoSize      =  -1  'True
599|     Caption       =  "Font Sample"
600|     FontBold      =  0   'False
601|     FontItalic    =  0   'False
602|     FontName      =  "Arial"
603|     FontSize      =  8.25
604|     FontStrikethru =  0   'False
605|     FontUnderline =  0   'False
606|     Height        =  210
607|     Index         =  2
608|     Left          =  7320
609|     TabIndex      =  17
610|     Top           =  1140
611|     Width         =  885
612|   End
613|   Begin Label lblSpecimen
614|     AutoSize      =  -1  'True
615|     Caption       =  "Font Sample"
616|     FontBold      =  0   'False
617|     FontItalic    =  0   'False
618|     FontName      =  "Arial"
619|     FontSize      =  8.25
620|     FontStrikethru =  0   'False
621|     FontUnderline =  0   'False
622|     Height        =  210
623|     Index         =  1
624|     Left          =  7320
625|     TabIndex      =  20
626|     Top           =  780
627|     Width         =  885
628|   End
629|   Begin Label lblSpecimen
630|     AutoSize      =  -1  'True
631|     Caption       =  "Font Sample"
632|     FontBold      =  0   'False
633|     FontItalic    =  0   'False
634|     FontName      =  "Arial"
635|     FontSize      =  8.25
636|     FontStrikethru =  0   'False
637|     FontUnderline =  0   'False
638|     Height        =  210
639|     Index         =  0
640|     Left          =  7320
641|     TabIndex      =  30
642|     Top           =  420
643|     Width         =  885
644|   End
645|   Begin Label zlblTextElement
646|     Caption       =  "&Footer"
647|     FontBold      =  0   'False
648|     FontItalic    =  0   'False
```

```
649|     FontName      = "MS Sans Serif"
650|     FontSize      = 8.25
651|     FontStrikethru = 0  'False
652|     FontUnderline  = 0  'False
653|     Height        = 255
654|     Index         = 2
655|     Left          = 60
656|     TabIndex      = 32
657|     Top           = 1140
658|     Width         = 735
659|   End
660|   Begin Label zlblTextElement
661|     Caption       = "&Title"
662|     FontBold      = 0  'False
663|     FontItalic    = 0  'False
664|     FontName      = "MS Sans Serif"
665|     FontSize      = 8.25
666|     FontStrikethru = 0  'False
667|     FontUnderline  = 0  'False
668|     Height        = 255
669|     Index         = 1
670|     Left          = 60
671|     TabIndex      = 38
672|     Top           = 780
673|     Width         = 735
674|   End
675|   Begin Label zlblTextElement
676|     Caption       = "&Header"
677|     FontBold      = 0  'False
678|     FontItalic    = 0  'False
679|     FontName      = "MS Sans Serif"
680|     FontSize      = 8.25
681|     FontStrikethru = 0  'False
682|     FontUnderline  = 0  'False
683|     Height        = 255
684|     Index         = 0
685|     Left          = 60
686|     TabIndex      = 31
687|     Top           = 420
688|     Width         = 735
689|   End
690|   Begin Label zlblY
691|     Caption       = "Y"
692|     FontBold      = 0  'False
693|     FontItalic    = 0  'False
694|     FontName      = "MS Sans Serif"
695|     FontSize      = 8.25
696|     FontStrikethru = 0  'False
697|     FontUnderline  = 0  'False
698|     Height        = 255
699|     Left          = 6360
700|     TabIndex      = 19
701|     Top           = 60
702|     Width         = 255
```

```
703|    End
704|    Begin Label zlblX
705|      Caption       =  "X"
706|      FontBold      =  0  'False
707|      FontItalic    =  0  'False
708|      FontName      =  "MS Sans Serif"
709|      FontSize      =  8.25
710|      FontStrikethru =  0  'False
711|      FontUnderline  =  0  'False
712|      Height        =  255
713|      Left          =  5880
714|      TabIndex      =  18
715|      Top           =  60
716|      Width         =  255
717|    End
718| End
719| Option Explicit
720| Dim MapArea As RealRect, PageArea As RealRect
721|
722| Sub AdjustPICBoundaries (NewBoundaries As RealRect)
723|    picRight.Left = NewBoundaries.Right
724|    picLeft.Left = NewBoundaries.Left - picLeft.Width
725|    Dim VertMiddle As Single
726|    VertMiddle = .5 * (NewBoundaries.Bottom + NewBoundaries.Top)
727|    picRight.Top = VertMiddle
728|    picLeft.Top = VertMiddle
729|
730|    picTop.Top = NewBoundaries.Top - picTop.Height 'was +
731|    picBottom.Top = NewBoundaries.Bottom
732|    Dim HorzMiddle As Single
733|    HorzMiddle = .5 * (NewBoundaries.Left + NewBoundaries.Right)
734|    picTop.Left = HorzMiddle
735|    picBottom.Left = HorzMiddle
736|
737|    picPageLayout_Paint
738|
739| End Sub
740|
741| Sub cmdCancel_Click ()
742| Me.cmdCancel.SetFocus
743| Me.Hide
744| End Sub
745|
746| Sub cmdDrag_Click (Index As Integer)
747|    txtMapBlurb(Index).DragMode = 1
748| End Sub
749|
750| Sub cmdFont_Click (Index As Integer)
751|  'Copy information from label to common font dialog
752|  On Error Resume Next
753|    Me!CMDialog1.FontName = lblSpecimen(Index).FontName
754|    Me!CMDialog1.FontBold = lblSpecimen(Index).FontBold
755|    Me!CMDialog1.FontItalic = lblSpecimen(Index).FontItalic
756|    Me!CMDialog1.FontSize = lblSpecimen(Index).FontSize
```

```
757|
758|     Me!CMDialog1.Flags = CF_BOTH
759|     Me!CMDialog1.CancelError = True
760|
761|     Me!CMDialog1.Action = 4'Font dialog
762|
763|
764|     If Err <> 0 Then Exit Sub
765|      lblSpecimen(Index).FontName = Me!CMDialog1.FontName
766|      lblSpecimen(Index).FontBold = Me!CMDialog1.FontBold
767|      lblSpecimen(Index).FontItalic = Me!CMDialog1.FontItalic
768|      lblSpecimen(Index).FontSize = Me!CMDialog1.FontSize
769|
770|
771|
772|  End Sub
773|
774|  Sub cmdOK_Click ()
775|  Me.cmdOK.SetFocus
776|  Me.Hide
777|  Me.cmdOK.FontItalic = True
778|  End Sub
779|
780|  Sub cmdPrinterSetup_Click ()
781|   printer.EndDoc 'Needed so that previous calls to Printer.Scalemode
782|                 ' dont interfere with the ability to reset the printer.
783|   CMDialog1.PrinterDefault = True
784|   CMDialog1.Action = 5
785|   If printer.Width <> PageArea.Right - PageArea.Left Then cmdRefreshLayout_Click
786|  End Sub
787|
788|  Sub cmdRefreshLayout_Click ()
789|
790|  Dim PageHeight As Single, PageWidth As Single
791|    printer.ScaleMode = 6 'Puts into millimeter mode
792|    PageHeight = printer.ScaleHeight
793|    picPageLayout.Height = .2 * PageHeight
794|    PageWidth = printer.ScaleWidth
795|    picPageLayout.Width = .2 * PageWidth
796|    PageArea.Left = 0
797|    PageArea.Right = PageWidth
798|    PageArea.Top = 0
799|    PageArea.Bottom = PageHeight
800|    picPageLayout.Scale (0, 0)-(PageWidth, PageHeight)
801|
802|
803|   MapArea.Left = .1 * PageWidth
804|   MapArea.Right = .9 * PageWidth
805|   MapArea.Top = .2 * PageHeight
806|   MapArea.Bottom = .8 * PageHeight
807|
808|
809|
810|  AdjustPICBoundaries MapArea
```

```
811| UpdateTextBoxes MapArea
812| End Sub
813|
814| Sub DrawTextBackground (strText As String, rctFitIn As RealRect, Color As Long)
815| ' call PrintInBox (strText , Target, rctFitIn )
816| "This routine will print the text strText starting in
817| " the upper left of rctFitIn.  Line wraps will occur at
818| " rctFitIn.Right (rctFitIn.Bottom is ignored)
819|
820| 'Target can be a picturebox, the Printer object, or a form
821| '(Target must support the Print method)
822| 'It is assumed the caller has already selected the font
823| ' and such for the Target
824|
825| Dim strCopy As String, strThisLine As String, ThisWord As String, strMaybe
826| Dim nChar As Long, nCh As Long, xMax As Single, xSoFar As Single, YTotalHeight As
    Single
827|
828|  strCopy = Trim$(strText)
829|  xMax = rctFitIn.Right - rctFitIn.Left
830|  YTotalHeight = 0
831|
832|  strThisLine = "": strMaybe = ""
833|  xMax = xMax * 5'Scaling factor for printer to this view
834| 'Maybe set text align
835| Do
836|    ThisWord = NextWord(strCopy, nChar)
837|    strMaybe = strThisLine & " " & ThisWord
838|    xSoFar = Me!picPageLayout.TextWidth(strMaybe)
839|   ' dDebug.Print Target.CurrentX, Target.CurrentY, xSoFar, xMax
840|    If xSoFar > xMax Then
841|       YTotalHeight = YTotalHeight + Me!picPageLayout.TextHeight(strThisLine)
842|       strThisLine = ThisWord
843|      Else
844|       strThisLine = strThisLine & " " & ThisWord
845|    End If
846|
847| Loop Until ThisWord = ""
848| YTotalHeight = (YTotalHeight + Me!picPageLayout.TextHeight(strThisLine))
849| YTotalHeight = YTotalHeight * .2
850| Me!picPageLayout.Line (rctFitIn.Left, rctFitIn.Top)-(rctFitIn.Right, rctFitIn.Top -
    YTotalHeight), Color, BF
851|
852| End Sub
853|
854| Sub Form_DragDrop (Source As Control, X As Single, Y As Single)
855|   ' If Source Is txtMapTitle Then txtMapTitle.DragMode = 0
856| End Sub
857|
858| Sub Form_Load ()
859| 'This whole page layout thing is going to simply be done at 1:5 scale
860| '  so that if the user wants use some huge pagesize, they need to
861| '  tweak this a bit.
862| cmdRefreshLayout_Click
```

```
End Sub

Sub Form_Paint ()
  cmdOK.FontItalic = False
  If PageArea.Left = PageArea.Right Then cmdRefreshLayout_Click
End Sub

Sub picPageLayout_DragDrop (Source As Control, X As Single, Y As Single)
 Dim NewX As Single, NewY As Single
 NewX = X
 NewY = Y

 If Source Is picLeft Then MapArea.Left = NewX - .5 * picLeft.Width
 If Source Is picRight Then MapArea.Right = NewX
 If Source Is picTop Then MapArea.Top = NewY - .5 * picTop.Height
 If Source Is picBottom Then MapArea.Bottom = NewY
 AdjustPICBoundaries MapArea
 UpdateTextBoxes MapArea

 'Always set the dragmode to manual for the text boxes.
   txtMapBlurb(0).DragMode = 0
   txtMapBlurb(1).DragMode = 0
   txtMapBlurb(2).DragMode = 0

 If Source Is txtMapBlurb(0) Then
    txtBlurbX(0) = X
    txtBlurbY(0) = Y
    Call picPageLayout_Paint
 End If
 If Source Is txtMapBlurb(1) Then
    txtBlurbX(1) = X
    txtBlurbY(1) = Y
    Call picPageLayout_Paint
 End If
 If Source Is txtMapBlurb(2) Then
    txtBlurbX(2) = X
    txtBlurbY(2) = Y
    Call picPageLayout_Paint
 End If



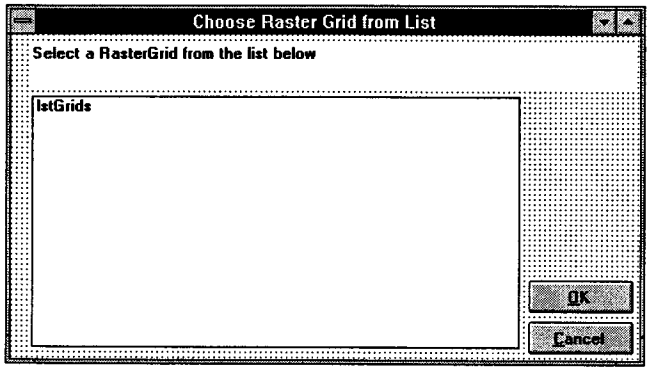
End Sub

Sub picPageLayout_DragOver (Source As Control, X As Single, Y As Single, State As
Integer)
Exit Sub
 Dim NewX As Single, NewY As Single
 NewX = X
 NewY = Y

 If Source Is picLeft Then MapArea.Left = NewX
 If Source Is picRight Then MapArea.Right = NewX
 If Source Is picTop Then MapArea.Top = NewY
```

```
916|  If Source Is picBottom Then MapArea.Bottom = NewY
917|  AdjustPICBoundaries MapArea
918|
919|
920|  End Sub
921|
922|  Sub picPageLayout_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As
      Single)
923|   'Me.Caption = "Mouse is in picPageLayout. X=" & X & ";   Y=" & Y
924|  End Sub
925|
926|  Sub picPageLayout_Paint ()
927|
928|    If MapArea.Left <> MapArea.Right Then
929|       picPageLayout.Cls
930|       picPageLayout.Line (MapArea.Left, MapArea.Top)-(MapArea.Right,
     MapArea.Bottom), RGB(0, 0, 128), BF
931|    End If
932|
933|  Dim n As Integer
934|  Dim rctFitIn As RealRect
935|  rctFitIn.Right = picPageLayout.ScaleWidth - 25.4
936|
937|  For n = 0 To 2
938|    If txtBlurbX(n) <> "" And txtBlurbY(n) <> "" Then
939|       picPageLayout.FontName = lblSpecimen(n).FontName
940|       picPageLayout.FontBold = lblSpecimen(n).FontBold
941|       picPageLayout.FontItalic = lblSpecimen(n).FontItalic
942|       picPageLayout.FontSize = lblSpecimen(n).FontSize '* .2  'Prints at one-fifth scale.
943|
944|       rctFitIn.Left = txtBlurbX(n)
945|       rctFitIn.Top = txtBlurbY(n)
946|
947|       Call DrawTextBackground(CStr(txtMapBlurb(n).Text), rctFitIn, RGB(128, 0, 0))
948|  '     picPageLayout.Print txtMapBlurb(n)
949|    End If
950|  Next n
951|
952|  End Sub
953|
954|  Sub txtBlurbX_Change (Index As Integer)
955|      Call picPageLayout_Paint
956|  End Sub
957|
958|  Sub txtBlurbY_Change (Index As Integer)
959|     Call picPageLayout_Paint
960|  End Sub
961|
962|  Sub txtMapBlurb_Change (Index As Integer)
963|       Call picPageLayout_Paint
964|  End Sub
965|
966|  Sub txtMapTitle_Change ()
967|     Call picPageLayout_Paint
```

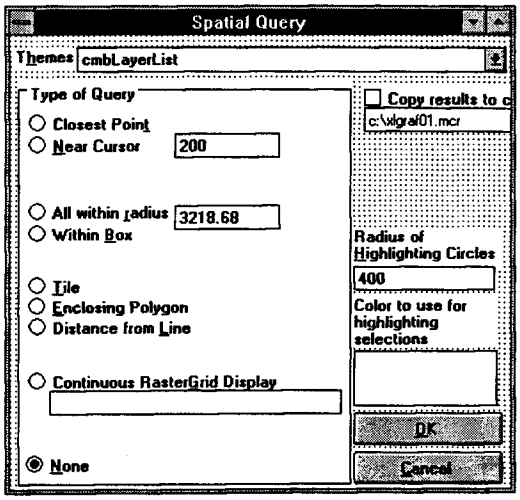
```
968|  End Sub
969|
970|  Sub txtPrintScale_GotFocus ()
971|      txtPrintScale.SelStart = 0: txtPrintScale.SelLength = 32000
972|  End Sub
973|
974|  Sub UpdateTextBoxes (NewBoundaries As RealRect)
975|    txtLeftMargin = NewBoundaries.Left
976|    txtRightMargin = NewBoundaries.Right
977|    txtTopMargin = NewBoundaries.Top
978|    txtBottomMargin = NewBoundaries.Bottom
979|  End Sub
980|
```


Choose Raster Grid from List
Select a RasterGrid from the list below
lstGrids
OK
Cancel

```
VERSION 2.00
Begin Form frmRasterGridList
  BorderStyle    = 3 'Fixed Double
  Caption        = "Choose Raster Grid from List"
  ClientHeight   = 3810
  ClientLeft     = 225
  ClientTop      = 765
  ClientWidth    = 7290
  ControlBox     = 0 'False
  Height         = 4215
  Left           = 165
  LinkTopic      = "Form1"
  MaxButton      = 0 'False
  MinButton      = 0 'False
  ScaleHeight    = 3810
  ScaleWidth     = 7290
  Top            = 420
  Width          = 7410
  Begin CommandButton cmdCancel
    Caption      = "&Cancel"
    Height       = 375
    Left         = 6000
    TabIndex     = 3
    Top          = 3360
    Width        = 1215
  End
  Begin CommandButton cmdOK
    Caption      = "&OK"
    Height       = 375
    Left         = 6000
    TabIndex     = 2
    Top          = 2880
    Width        = 1215
  End
  Begin ListBox lstGrids
    Height       = 2955
    Left         = 240
    TabIndex     = 1
    Top          = 720
    Width        = 5655
  End
  Begin Label lblPrompt
    Caption      = "Select a RasterGrid from the list below"
    Height       = 495
    Left         = 240
    TabIndex     = 0
    Top          = 120
    Width        = 6975
  End
End
Option Explicit

Sub cmdCancel_Click ()
Me.Hide
```

```
55| Me!cmdOK.FontItalic = False
56| End Sub
57|
58| Sub cmdOK_Click ()
59|   Me.Hide
60|   Me!cmdOK.FontItalic = True
61| End Sub
62|
63| Sub Form_Load ()
64| On Error GoTo ERRFormLoad
65|  'Dim snpListGrids As snapshot
66|  'Set snpListGrids = MapDB.CreateSnapshot("SELECT DISTINCT [GridName] FROM
    [zWHEATtblRasterGrids];")
67|  'snpListGrids.MoveFirst
68|  'Do
69|  '  lstGrids.AddItem snpListGrids("GridName")
70|  '  snpListGrids.MoveNext
71|  'Loop While Not snpListGrids.EOF
72|  'snpListGrids.Close
73| Exit Sub
74| ERRFormLoad:
75|  Select Case Err
76|   Case 3011, 3078
77|    'ZDefineGridTable MapDB
78|    MsgBox "There are no RasterGrids in this database."
79|    Resume
80|   Case 3021 'No  current record.
81|    Err = 0
82|    Exit Sub
83|    Resume
84|   Case Else
85|    MsgBox "Error #" & Err & " occurred trying to list RasterGrids." & CRLF & Error
86|    Err = 0
87|    Exit Sub
88|    Resume
89|  End Select
90| End Sub
91|
92| Sub Form_Paint ()
93|  Me!cmdOK.FontItalic = False
94| End Sub
95|
```


Spatial Query
Themes
cmbLayerList
Type of Query
Closest Point
Near Cursor
200
All within radius
3218.68
Within Box
Tile
Enclosing Polygon
Distance from Line
Continuous RasterGrid Display
None
Copy results to c
c:\vlgraf01.mcr
Radius of Highlighting Circles
400
Color to use for highlighting selections
OK
Cancel

```
VERSION 2.00
Begin Form frmSpatialQuery
  Caption      = "Spatial Query"
  ClientHeight = 5280
  ClientLeft   = 2685
  ClientTop    = 720
  ClientWidth  = 5775
  ControlBox   = 0  'False
  Height       = 5685
  Left         = 2625
  LinkTopic    = "Form1"
  ScaleHeight  = 5280
  ScaleWidth   = 5775
  Top          = 375
  Width        = 5895
  Begin TextBox txtMacroName
    FontBold      = 0  'False
    FontItalic    = 0  'False
    FontName      = "MS Sans Serif"
    FontSize      = 8.25
    FontStrikethru = 0  'False
    FontUnderline  = 0  'False
    Height        = 285
    Left          = 4080
    TabIndex      = 22
    Text          = "c:\xlgraf01.mcr "
    Top           = 840
    Width         = 3495
  End
  Begin CheckBox chkRunMacro
    Caption       = "Copy results to clipboard then run this macro"
    Height        = 255
    Left          = 4080
    TabIndex      = 21
    Top           = 600
    Width         = 3495
  End
  Begin TextBox txtHighlightRadius
    Height        = 285
    Left          = 3960
    TabIndex      = 18
    Text          = "400"
    Top           = 2700
    Width         = 1635
  End
  Begin PictureBox Picture1
    Height        = 645
    Left          = 3960
    ScaleHeight   = 615
    ScaleWidth    = 1635
    TabIndex      = 15
    Top           = 3690
    Width         = 1665
  End
```

```
Begin CommandButton cmdCancel
  Caption      = "&Cancel"
  Height       = 375
  Left         = 3960
  TabIndex     = 11
  Top          = 4860
  Width        = 1725
End
Begin CommandButton cmdOK
  Caption      = "&OK"
  Height       = 375
  Left         = 3960
  TabIndex     = 10
  Top          = 4410
  Width        = 1725
End
Begin Frame framTypeOfSearch
  Caption      = "Type of Query"
  Height       = 4695
  Left         = 90
  TabIndex     = 2
  Top          = 540
  Width        = 3840
  Begin TextBox txtRasterGrid
    FontBold      = 0  'False
    FontItalic    = 0  'False
    FontName      = "MS Sans Serif"
    FontSize      = 8.25
    FontStrikethru = 0  'False
    FontUnderline = 0  'False
    Height        = 285
    Left          = 360
    TabIndex      = 20
    Top           = 3600
    Width         = 3375
  End
  Begin OptionButton optRaster
    Caption       = "Continuous Raster&Grid Display"
    Height        = 255
    Left          = 120
    TabIndex      = 19
    Top           = 3360
    Width         = 2895
  End
  Begin OptionButton optNone
    Caption       = "&None"
    Height        = 255
    Left          = 90
    TabIndex      = 14
    Top           = 4320
    Value         = -1  'True
    Width         = 2055
  End
  Begin TextBox txtBoxDimensions
```

```
109|       Height      = 285
110|       Left        = 1800
111|       TabIndex    = 13
112|       Text        = "200"
113|       Top         = 600
114|       Visible     = 0  'False
115|       Width       = 1215
116|     End
117|     Begin OptionButton optNearBox
118|       Caption     = "&Near Cursor"
119|       Enabled     = 0  'False
120|       Height      = 255
121|       Left        = 120
122|       TabIndex    = 12
123|       Top         = 600
124|       Visible     = 0  'False
125|       Width       = 1575
126|     End
127|     Begin OptionButton optWithinBox
128|       Caption     = "Within &Box"
129|       Enabled     = 0  'False
130|       Height      = 255
131|       Left        = 120
132|       TabIndex    = 9
133|       Top         = 1635
134|       Visible     = 0  'False
135|       Width       = 1305
136|     End
137|     Begin TextBox txtSearchRadius
138|       Height      = 285
139|       Left        = 1800
140|       TabIndex    = 8
141|       Text        = "3218.68"
142|       Top         = 1440
143|       Width       = 1215
144|     End
145|     Begin OptionButton optLine
146|       Caption     = "Distance from &Line"
147|       Height      = 255
148|       Left        = 120
149|       TabIndex    = 7
150|       Top         = 2730
151|       Width       = 2175
152|     End
153|     Begin OptionButton optPolygon
154|       Caption     = "&Enclosing Polygon"
155|       Height      = 255
156|       Left        = 120
157|       TabIndex    = 6
158|       Top         = 2490
159|       Width       = 2175
160|     End
161|     Begin OptionButton optTile
162|       Caption     = "&Tile"
```

```
163|       Height      = 255
164|       Left        = 120
165|       TabIndex    = 5
166|       Top         = 2250
167|       Width       = 855
168|     End
169|     Begin OptionButton optRadius
170|       Caption     = "All within &radius"
171|       Height      = 315
172|       Left        = 120
173|       TabIndex    = 4
174|       Top         = 1350
175|       Width       = 1695
176|     End
177|     Begin OptionButton optClosestPoint
178|       Caption     = "Closest Poin&t"
179|       Height      = 240
180|       Left        = 120
181|       TabIndex    = 3
182|       Top         = 360
183|       Width       = 2295
184|     End
185|   End
186|   Begin ComboBox cmbLayerList
187|     Height        = 300
188|     Left          = 765
189|     Sorted        = -1 'True
190|     Style         = 2 'Dropdown List
191|     TabIndex      = 1
192|     Top           = 135
193|     Width         = 4960
194|   End
195|   Begin Label zlblHighlightRadius
196|     Caption       = "Radius of &Highlighting Circles"
197|     Height        = 465
198|     Left          = 3960
199|     TabIndex      = 17
200|     Top           = 2250
201|     Width         = 1725
202|   End
203|   Begin Label zlblHighlightColor
204|     Caption       = "Color to use for highlighting selections"
205|     Height        = 555
206|     Left          = 3960
207|     TabIndex      = 16
208|     Top           = 3060
209|     Width         = 1635
210|   End
211|   Begin Label lblLayers
212|     Caption       = "T&hemes"
213|     Height        = 255
214|     Left          = 60
215|     TabIndex      = 0
216|     Top           = 150
```

```
217|      Width         =  735
218|    End
219| End
220| Option Explicit
221|
222| Sub cmbLayerList_Click ()
223|      chkRunMacro = 0
224| End Sub
225|
226| Sub cmdCancel_Click ()
227| Me.Hide
228| End Sub
229|
230| Sub cmdOK_Click ()
231| If cmbLayerList = "" Then
232|    MsgBox "A map layer must be selected before trying to perform spatial querying on it."
     & CRLF & "Or you can hit CANCEL to not modify the spatial query right now and get out
     of here."
233|    cmbLayerList.SetFocus
234|    SendKeys "%{DOWN}"
235|    Exit Sub
236| End If
237| On Error GoTo cmdOKErr
238| ' maybe put in a bit about checking for pre-existence of
239| '  the table or file.
240|
241|  Me.cmdOK.SetFocus
242|
243|  Me.Hide
244|  cmdOK.FontItalic = True
245| Exit Sub
246| cmdOKErr:
247|
248|
249|
250|
251|
252| End Sub
253|
254| Sub Form_Paint ()
255|  If cmbLayerList.ListIndex = -1 Then cmbLayerList.ListIndex = 0
256|  cmdOK.FontItalic = False
257| End Sub
258|
259| Sub optRaster_Click ()
260|
261| Dim GridName As String
262| ALLTypesAllowed(1) = DB_LONG
263|
264| GridName = UserChoiceOfRasterGrids(ALLTypesAllowed(), MapDB, "Choose a
     RasterGrid for continous spatial querying")
265|
266| If GridName = "" Then Exit Sub
267| txtRasterGrid = GridName
```

```
End Sub

Sub Picture1_DblClick ()

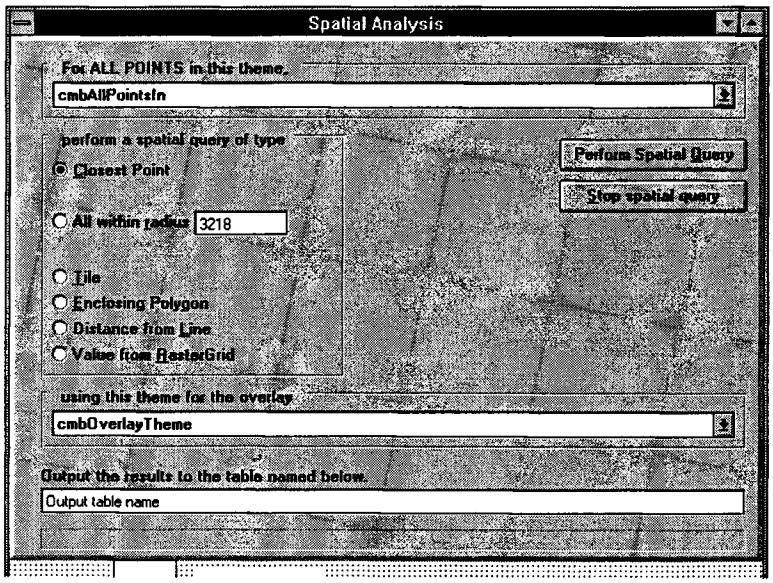
 Picture1.BackColor = GetColorWithCmnDialog(Picture1.BackColor)
End Sub

Sub Picture1_MouseDown (button As Integer, Shift As Integer, X As Single, Y As Single)
 Picture1.BackColor = GetColorWithCmnDialog(Picture1.BackColor)
End Sub

Sub txtMacroFileName_DblClick ()
 frmOZ.CMDialog1(0).Filename = txtMacroName
 frmOZ.CMDialog1(0).Filter = "WHEAT Macros|*.mcr"
 frmOZ.CMDialog1(0).Action = 1
 txtMacroName = frmOZ.CMDialog1(0).Filename
 Unload frmOZ
End Sub

Sub zlblHighlightColor_Click ()
 Picture1_DblClick
End Sub

```


Spatial Analysis
For ALL POINTS in this theme,
cmbAllPointsIn
perform a spatial query of type
Closest Point
All within radius
3218
Tile
Enclosing Polygon
Distance from Line
Value from RasterGrid
Perform Spatial Query
Stop spatial query
using this theme for the overlay
cmbOverlayTheme
Output the results to the table named below.
Output table name

```
VERSION 2.00
Begin Form frmSpatialAnalysis
  Caption        = "Spatial Analysis"
  ClientHeight   = 6570
  ClientLeft     = 405
  ClientTop      = 870
  ClientWidth    = 8730
  Height         = 6975
  Icon           = SPTANLY1.FRX:0000
  Left           = 345
  LinkTopic      = "Form1"
  ScaleHeight    = 6570
  ScaleWidth     = 8730
  Top            = 525
  Width          = 8850
  Begin PictureBox picMap
    Height       = 495
    Left         = 1200
    ScaleHeight  = 465
    ScaleWidth   = 705
    TabIndex     = 23
    Top          = 6120
    Visible      = 0  'False
    Width        = 735
  End
  Begin SSPanel zzpnlMain
    Align        = 1  'Align Top
    BackColor    = &H00C0C0C0&
    BevelWidth   = 4
    Font3D       = 0  'None
    Height       = 6135
    Left         = 0
    TabIndex     = 0
    Top          = 0
    Width        = 8730
    Begin SSCommand cmdStop
      Caption    = "&Stop spatial query"
      Font3D     = 0  'None
      Height     = 375
      Left       = 6360
      TabIndex   = 21
      Top        = 1680
      Width      = 2175
    End
    Begin SSCommand cmdRunSpatialQuery
      Caption    = "Perform Spatial &Query"
      Font3D     = 0  'None
      Height     = 375
      Left       = 6360
      TabIndex   = 20
      Top        = 1200
      Width      = 2175
    End
    Begin SSFrame zFrameTypeOfQuery
```

```
55|       Caption      =  " perform a spatial query of type "
56|       Font3D       =  0  'None
57|       Height       =  2895
58|       Left         =  360
59|       TabIndex     =  10
60|       Top          =  1080
61|       Width        =  3495
62|       Begin SSOption optRaster
63|         Caption      =  "Value from &RasterGrid"
64|         Font3D       =  0  'None
65|         Height       =  255
66|         Left         =  120
67|         TabIndex     =  18
68|         TabStop      =  0  'False
69|         Top          =  2500
70|         Width        =  2175
71|       End
72|       Begin SSOption optLine
73|         Caption      =  "Distance from &Line"
74|         Font3D       =  0  'None
75|         Height       =  255
76|         Left         =  120
77|         TabIndex     =  17
78|         TabStop      =  0  'False
79|         Top          =  2200
80|         Width        =  1935
81|       End
82|       Begin SSPanel pnlSearchRadius
83|         AutoSize     =  3  'AutoSize Child To Panel
84|         BackColor    =  &H00C0C0C0&
85|         BevelOuter   =  1  'Inset
86|         Caption      =  "Panel3D1"
87|         FloodType    =  1  'Left To Right
88|         Font3D       =  0  'None
89|         Height       =  300
90|         Left         =  1755
91|         TabIndex     =  15
92|         Top          =  960
93|         Width        =  1095
94|         Begin TextBox txtSearchRadius
95|           FontBold      =  0  'False
96|           FontItalic    =  0  'False
97|           FontName      =  "MS Sans Serif"
98|           FontSize      =  8.25
99|           FontStrikethru =  0  'False
100|          FontUnderline =  0  'False
101|          Height        =  285
102|          Left          =  15
103|          TabIndex      =  16
104|          Text          =  "3218"
105|          Top           =  15
106|          Width         =  1065
107|        End
108|      End
```

```
109|       Begin SSOption optPolygon
110|         Caption      = "&Enclosing Polygon"
111|         Font3D       = 0 'None
112|         Height       = 255
113|         Left         = 120
114|         TabIndex     = 14
115|         TabStop      = 0 'False
116|         Top          = 1900
117|         Width        = 1935
118|       End
119|       Begin SSOption optTile
120|         Caption      = "&Tile"
121|         Font3D       = 0 'None
122|         Height       = 255
123|         Left         = 120
124|         TabIndex     = 13
125|         TabStop      = 0 'False
126|         Top          = 1600
127|         Width        = 735
128|       End
129|       Begin SSOption optRadius
130|         Caption      = "All within &radius"
131|         Font3D       = 0 'None
132|         Height       = 255
133|         Left         = 120
134|         TabIndex     = 12
135|         TabStop      = 0 'False
136|         Top          = 960
137|         Width        = 1695
138|       End
139|       Begin SSOption optClosestPoint
140|         Caption      = "&Closest Point"
141|         Font3D       = 0 'None
142|         Height       = 255
143|         Left         = 120
144|         TabIndex     = 11
145|         Top          = 360
146|         Value        = -1 'True
147|         Width        = 2655
148|       End
149|     End
150|     Begin SSPanel pnlOutTable
151|       AutoSize       = 3 'AutoSize Child To Panel
152|       BackColor      = &H00C0C0C0&
153|       BevelOuter     = 1 'Inset
154|       Caption        = "Panel3D3"
155|       FloodType      = 1 'Left To Right
156|       Font3D         = 0 'None
157|       Height         = 320
158|       Left           = 360
159|       TabIndex       = 8
160|       Top            = 5280
161|       Width          = 8175
162|       Begin TextBox txtOutTable
```

```
163|         FontBold      = 0  'False
164|         FontItalic    = 0  'False
165|         FontName      = "MS Sans Serif"
166|         FontSize      = 8.25
167|         FontStrikethru = 0  'False
168|         FontUnderline = 0  'False
169|         Height        = 285
170|         Left          = 15
171|         TabIndex      = 9
172|         Text          = "Output table name"
173|         Top           = 15
174|         Width         = 8145
175|       End
176|     End
177|     Begin SSFrame zFrameOtherTheme
178|       Caption       = " using this theme for the overlay "
179|       Font3D        = 0 'None
180|       Height        = 735
181|       Left          = 360
182|       TabIndex      = 3
183|       Top           = 4080
184|       Width         = 8175
185|       Begin SSPanel pnlOtherTheme
186|         AutoSize      = 3 'AutoSize Child To Panel
187|         BackColor     = &H00C0C0C0&
188|         BevelOuter    = 1 'Inset
189|         Caption       = "Panel3D2"
190|         FloodType     = 1 'Left To Right
191|         Font3D        = 0 'None
192|         Height        = 330
193|         Left          = 120
194|         TabIndex      = 6
195|         Top           = 280
196|         Width         = 7935
197|         Begin ComboBox cmbOverlayTheme
198|           Height        = 300
199|           Left          = 15
200|           Style         = 2 'Dropdown List
201|           TabIndex      = 7
202|           Top           = 15
203|           Width         = 7905
204|         End
205|       End
206|     End
207|     Begin SSFrame zFrameAllPointsIn
208|       Caption       = " For ALL POINTS in this theme, "
209|       Font3D        = 0 'None
210|       Height        = 735
211|       Left          = 360
212|       TabIndex      = 2
213|       Top           = 240
214|       Width         = 8175
215|       Begin SSPanel pnlAllPointsTable
216|         AutoSize      = 3 'AutoSize Child To Panel
```

```
217|          BackColor     =  &H00C0C0C0&
218|          BevelOuter    =  1  'Inset
219|          Caption       =  "Panel3D1"
220|          FloodType     =  1  'Left To Right
221|          Font3D        =  0  'None
222|          Height        =  330
223|          Left          =  120
224|          TabIndex      =  4
225|          Top           =  280
226|          Width         =  7935
227|          Begin ComboBox cmbAllPointsIn
228|            Height      =  300
229|            Left        =  15
230|            Style       =  2  'Dropdown List
231|            TabIndex    =  5
232|            Top         =  15
233|            Width       =  7905
234|          End
235|        End
236|      End
237|      Begin SSPanel pnlPercentDone
238|        BackColor     =  &H00C0C0C0&
239|        BevelOuter    =  1  'Inset
240|        Caption       =  "Panel3D1"
241|        FloodType     =  1  'Left To Right
242|        Font3D        =  0  'None
243|        Height        =  255
244|        Left          =  360
245|        TabIndex      =  1
246|        Top           =  5760
247|        Width         =  8175
248|      End
249|      Begin Label Label1
250|        BackStyle     =  0  'Transparent
251|        Caption       =  "Output the results to the table named below."
252|        Height        =  255
253|        Left          =  360
254|        TabIndex      =  19
255|        Top           =  5040
256|        Width         =  5295
257|      End
258|    End
259|    Begin Label lblTrigger
260|      Height        =  255
261|      Left          =  2160
262|      TabIndex      =  22
263|      Top           =  6240
264|      Width         =  1455
265|    End
266|  End
267|  Option Explicit
268|  Dim flgStopQS As Long 'Used to halt execution within do-loops.
269|  Dim N As Long, nRecsOut As Long
270|  Dim dynSpatialQuery As dynaset, dynQSResults As dynaset, QSTheme As typTheme
```

```
Dim lngSearchType As Long, SearchRadius As Single, lngSearchThemeID As Long
Dim QSDataOut() As typTaggedIDNum

Dim PointTheme As typTheme, OverlayTheme As typTheme

Dim strTableUsedForOutput As String, strSearchType As String



'Added to allow raster grid queries in spatial analysis
Dim QSRasterGridNumber As Long

Sub cmdRunSpatialQuery_Click ()
 On Error GoTo ERRcmdRunSpatialQuery_Click

  InitializeSpatialQueries

    strTableUsedForOutput = Left(txtOutTable.Text, 64)
  DefineOutTable MapDB, strTableUsedForOutput
    txtOutTable = strTableUsedForOutput

'For logging purposes
   Dim InputDescription As String
   InputDescription = "For each point in " & PointTheme.ThemeName & CRLF & "     " &
PointTheme.MapSource & CRLF
   InputDescription = InputDescription & "find " & UCase$(strSearchType) & " in " & CRLF
   InputDescription = InputDescription & OverlayTheme.ThemeName & CRLF & "     " &
OverlayTheme.MapSource & CRLF
   InputDescription = InputDescription & "Started processing at " & Now

  RunSpatialQueries

'Write log information out to the LOG Table
   Dim dnsLOG As dynaset

    Set dnsLOG = MapDB.CreateDynaset("ztblWHEAT_LOG")
    dnsLOG.AddNew
    dnsLOG("ProgramName") = strProgName & strVersion & " Spatial Analysis Module
WHEAT Electronic Mapping Package"
    dnsLOG("Inputs") = InputDescription
    dnsLOG("Outputs") = nRecsOut & " records were put into table [" &
strTableUsedForOutput & "] in database " & MapDB.Name & CRLF & "Done at " & Now
    dnsLOG("TimeStamp") = Now
    dnsLOG.Update

Beep
Beep
MsgBox "Finished running spatial query."


Exit Sub


ERRcmdRunSpatialQuery_Click:
```

```
321| Select Case Err
322|    Case 3011
323|       ZDefine_ztblWHEATLOG MapDB
324|       Err = 0
325|       Resume
326|    Case Else
327|       Beep
328|       MsgBox "Error #" & Err & " occurred in cmdRunSpatialQuery_Click." & CRLF &
    Error
329|       Err = 0
330|       Exit Sub
331|       Resume
332| End Select
333|
334|
335|
336| End Sub
337|
338| Sub DefineOutTable (DB As Database, TableName As String)
339|  Dim NewTable As New TableDef
340|
341| On Error GoTo ERRmfNewTable
342| If TableName = "" Then
343|    Exit Sub
344| End If
345|  NewTable.Name = TableName
346|
347|  Dim nfPointName As New Field
348|  nfPointName.Type = DB_TEXT: nfPointName.Size = 255: nfPointName.Name =
    "PointName"
349|  NewTable.Fields.Append nfPointName
350|
351|  Dim nfOverlayName As New Field
352|  nfOverlayName.Type = DB_TEXT: nfOverlayName.Size = 255: nfOverlayName.Name
    = "OverlayName"
353|  NewTable.Fields.Append nfOverlayName
354|
355|  Dim nfDistance As New Field
356|  nfDistance.Type = DB_SINGLE: nfDistance.Name = "Distance"
357|  NewTable.Fields.Append nfDistance
358|
359|  Dim nfBLOBJoining As New Field
360|  nfBLOBJoining.Type = DB_LONGBINARY: nfBLOBJoining.Name = "BLOBJoining"
361|  NewTable.Fields.Append nfBLOBJoining
362|
363|
364|  DB.TableDefs.Append NewTable
365|
366| Exit Sub
367| ERRmfNewTable:
368| Select Case Err
369|  Case 3010 'Table of that name already exists
370|    TableName = InputBox$("Table " & TableName & " already exists.  Please choose a
    new name.", "Defining Table", "TableName " & Format$(Now))
```

```
371|        If TableName = "" Then
372|           Err = 0
373|           Exit Sub
374|           Resume Next
375|        End If
376|     NewTable.Name = TableName
377|     Resume
378|  Case Else
379|     MsgBox "Error #" & Err & " occured in frmGridEdit.mfNewTable." & CRLF & Error
380|     Err = 0
381|     Exit Sub
382|     Resume
383|  End Select
384|
385|
386|
387|  End Sub
388|
389|  Sub Form_Resize ()
390|    zzpnlMain.Height = Me.ScaleHeight
391|  End Sub
392|
393|  Sub InitializeSpatialQueries ()
394|  CopyTYPTheme Themes(cmbAllPointsIn.ItemData(cmbAllPointsIn.ListIndex)),
     PointTheme
395|
396|      'Added 95 03 29 to allow checking of contouring/gridding
397|       If optRaster Then
398|        'Load raster grid
399|          Dim strGridWanted  As String
400|          strGridWanted = cmbOverlayTheme
401|          If strGridWanted = "" Then Exit Sub
402|          QSRasterGridNumber = UBound(RasterGrids) + 1 'Set a flag so Mouse_Move
     knows to run the spatial query
403|          ReDim Preserve RasterGrids(0 To QSRasterGridNumber)
404|          Call LoadRasterGridIntoGlobalMem(strGridWanted,
     RasterGrids(QSRasterGridNumber), MapDB)
405|          lngSearchType = QS_RASTERGRID
406|          strSearchType = "Find value in raster grid "
407|          OverlayTheme.MapSource = cmbOverlayTheme
408|          OverlayTheme.DBSource = cmbOverlayTheme
409|          OverlayTheme.ThemeName = cmbOverlayTheme
410|         Else
411|          QSRasterGridNumber = 0
412|
413|
414|
415|   CopyTYPTheme
     Themes(cmbOverlayTheme.ItemData(cmbOverlayTheme.ListIndex)), OverlayTheme
416|
417|          End If
418|
419|  If PointTheme.MapSource = OverlayTheme.MapSource Then
420|     MsgBox "For the most part, point-point self-overlays result in the following conclusion:
```

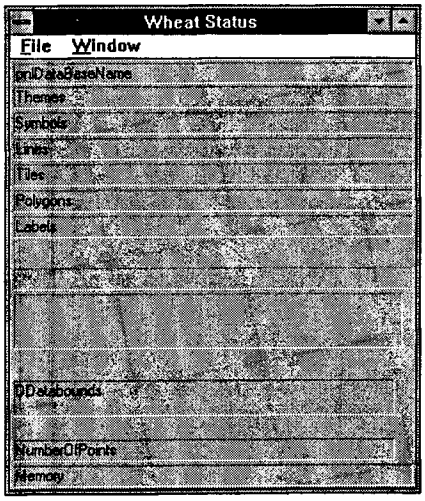
```
      Point A is the closest point to Point A. " & CRLF & "You should probably create a
      Delaunay Triangulation and examine that." & CRLF & "Exiting subroutine."
421|    Exit Sub
422|   End If
423|
424|
425|      ReDim QSDataOut(0 To 0)
426|       If optClosestPoint Then
427|          lngSearchType = QS_CLOSESTPOINT
428|          strSearchType = "Closest point "
429|         End If
430|
431|
432|       If optRadius Then
433|          lngSearchType = QS_RADIUSSEARCH
434|          QSDataOut(0).Dist1 = txtSearchRadius
435|          strSearchType = "All within radius of " & Format(QSDataOut(0).Dist1)
436|         End If
437|
438|
439|       If optPolygon Then
440|          If OverlayTheme.ThemeType <> SOLIDPOLYGON Then
441|             MsgBox "You are trying to perform a point-in-polygon overlay, but the overlay
      theme is not of type polygon. " & CRLF & "Exiting subroutine."
442|             Exit Sub
443|            End If
444|          lngSearchType = QS_WITHINPOLY
445|          strSearchType = "Enclosing polygons "
446|         End If
447|
448|       If optTile Then
449|          If OverlayTheme.ThemeType <> TILEBLOCK Then
450|             MsgBox "You are trying to perform a point-in-tile overlay, but the overlay theme
      is not of type TILEBLOCK. " & CRLF & "Exiting subroutine."
451|             Exit Sub
452|            End If
453|
454|          lngSearchType = QS_WITHINTILE
455|          strSearchType = "Enclosing tile "
456|         End If
457|
458|       If optLine Then
459|          If (OverlayTheme.ThemeType <> JOINTEDLINE) And
      (OverlayTheme.ThemeType <> SOLIDPOLYGON) Then
460|             MsgBox "You are trying to perform a distance-to-line overlay, but the overlay
      theme is not of type JOINTED LINE. " & CRLF & "Exiting subroutine."
461|             Exit Sub
462|            End If
463|          lngSearchType = QS_LINEDIST
464|          strSearchType = "Find closest line "
465|         End If
466|
467|
468|       'Added 95 03 29 GWP
```

```
469|       If optRaster Then Exit Sub
470|
471|            Dim strQSsql As String
472|            strQSsql = OverlayTheme.DBSource
473|            lngSearchThemeID =
    cmbOverlayTheme.ItemData(cmbOverlayTheme.ListIndex)
474|       Set dynSpatialQuery = MapDB.CreateDynaset(strQSsql)
475|       nRecsOut = 0
476| Exit Sub
477| ERR_mqSpatial_Click:
478| If Err = 91 Then 'The user closed the form, but did not hit cancel, so
479|              ' there is no active control on frmSpatQuery
480|   Err = 0
481|   Exit Sub
482|   'frmSpatialQuery.Show
483|   'frmSpatialQuery.cmdCancel.SetFocus
484|   'SendKeys "{ENTER}"
485|   Resume
486| End If
487|    MsgBox "Error number " & Err & " occurred while setting the spatial query, in
    mqSpatial." & CRLF & Error$
488|    Err = 0
489|    Exit Sub
490|    Resume Next
491|
492| End Sub
493|
494| Sub lblTrigger_Change ()
495|  If lblTrigger = "CLEAR" Then
496|    cmbAllPointsIn.Clear
497|    cmbOverlayTheme.Clear
498|    Exit Sub
499|  End If
500|  If lblTrigger = "STOP" Then
501|    flgStopQS = True
502|    DoEvents
503|    flgStopQS = False
504|    Exit Sub
505|  End If
506|  Dim N As Long
507|  cmbAllPointsIn.Clear
508|  cmbOverlayTheme.Clear
509|  For N = 1 To nThemes
510|     If Themes(N).ThemeType = ONEPOINT Then
511|          cmbAllPointsIn.AddItem Themes(N).ThemeName
512|          cmbAllPointsIn.ItemData(cmbAllPointsIn.NewIndex) = N
513|        End If
514|       cmbOverlayTheme.AddItem Themes(N).ThemeName
515|       cmbOverlayTheme.ItemData(cmbOverlayTheme.NewIndex) = N
516|  Next N
517|  If cmbAllPointsIn.ListCount < 1 Then
518|    MsgBox "The spatial analysis tool only performs spatial queries for points in a symbol
    theme, and you do not seem to have a symbol theme loaded."
519|    Exit Sub
```

```
520|   End If
521|   cmbAllPointsIn.ListIndex = 0
522|  ' cmbOverlayTheme.SetFocus
523|  ' SendKeys "%{DOWN}"
524| End Sub
525|
526| Sub optRaster_Click (Value As Integer)
527| Dim GridName As String
528|    ALLTypesAllowed(1) = DB_LONG
529|    GridName = UserChoiceOfRasterGrids(ALLTypesAllowed(), MapDB, "Choose a
    RasterGrid for continous spatial querying")
530|    If GridName = "" Then Exit Sub
531|    cmbOverlayTheme.AddItem GridName
532|    cmbOverlayTheme.ListIndex = cmbOverlayTheme.ListCount - 1
533| End Sub
534|
535| Sub RunSpatialQueries ()
536| On Error GoTo ERRRunSpatialQueries
537| Dim QS_X As Single, QS_Y As Single, varTime As Variant, strMouseLoc  As String
538| Dim nStart As Long, N  As Long, M As Long
539| ReDim ShortLine(1 To 2) As RealPoint
540| Dim Pt As RealPoint
541|
542|
543| Dim tblResultsOut As table
544|
545| Set tblResultsOut = MapDB.OpenTable(strTableUsedForOutput)
546|
547| Dim CellValue As Variant 'Added 95 03 29 to allow checking of contouring/gridding
548| For N = PointTheme.FirstLoc To PointTheme.LastLoc
549|  'me.pnlPercentDone.floodpercent=LimitTo((N-PointTheme.FirstLoc)/(PointTheme.LastLoc-PointTheme
550|  Me.pnlPercentDone.FloodPercent = (N - PointTheme.FirstLoc) / (PointTheme.LastLoc
    - PointTheme.FirstLoc) * 100
551|  DoEvents
552|  If flgStopQS Then Exit Sub
553|   ReDim Preserve QSDataOut(0 To 0)
554|   nStart = Points(N).nStart
555|   Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pt, nStart, One)
556|    QS_X = Pt.X
557|    QS_Y = Pt.Y
558|    ShortLine(1) = Pt
559|
560|    'Added 95 03 29 to allow checking of contouring/gridding
561|    If lngSearchType = QS_RASTERGRID Then
562|       'CellValue = GetXYCellValueGenericFromGlobalMem(QS_X, QS_Y,
    RasterGrids(QSRasterGridNumber), RasterGrids(QSRasterGridNumber).hGMem)
563|       tblResultsOut.AddNew
564|       tblResultsOut("PointName") = Points(N).FeatureID
565|       tblResultsOut("Distance") = DSng(GetXYCellValueGenericFromGlobalMem(QS_X,
    QS_Y, RasterGrids(QSRasterGridNumber),
    RasterGrids(QSRasterGridNumber).hGMem))
566|       tblResultsOut.Update
567|     Else
568|
```

```
569|
570|
571|
572|
573|
574|    dynSpatialQuery.Filter = ""
575|    Call SpatialSearch(QS_X, QS_Y, OverlayTheme, hGMemAllPoints, lngSearchType,
     dynSpatialQuery, QSDataOut(), 0&, Me.picMap)
576|    nRecsOut = UBound(QSDataOut)
577|    For M = 1 To UBound(QSDataOut)  'This won't execute if QSDataOut only has an
     element 0
578|         tblResultsOut.AddNew
579|         tblResultsOut("PointName") = Points(N).FeatureID
580|         tblResultsOut("OverlayName") = QSDataOut(M).FeatureID
581|         tblResultsOut("Distance") = QSDataOut(M).Dist1
582|         If QSDataOut(M).X <> 0 Then 'This only checks for whether a value was
     assigned.  Should be slicker.
583|            ShortLine(2).X = QSDataOut(M).X
584|            ShortLine(2).Y = QSDataOut(M).Y
585|            tblResultsOut("BLOBJoining") = RealPoints2BLOB(ShortLine())
586|          End If
587|         tblResultsOut.Update
588|    Next M
589|
590|
591|     'Added 95 03 29 to allow checking of contouring/gridding
592|    End If
593| Next N
594|
595|
596|
597| 'Added 95 03 29 to allow checking of contouring/gridding
598| If lngSearchType = QS_RASTERGRID Then
599|    ' discard the previously defined RasterGrid for spatial querying
600|    lngSearchType = GlobalFree(RasterGrids(QSRasterGridNumber).hGMem)
601|    RasterGrids(QSRasterGridNumber).hGMem = 0
602|    RasterGrids(QSRasterGridNumber).LayerName = "DISCARDED ALREADY"
603|    QSRasterGridNumber = 0
604|    'These should be harmless enough, even if QSRasterGridNumber=0
605|    'Leave the cursor alone.
606| End If
607|
608|
609|
610| Exit Sub
611|
612| ERRRunSpatialQueries:
613| Select Case Err
614|
615|  Case Else
616|    MsgBox "Error #" & Err & " occured in frmGridEdit.RunSpatialQueries." & CRLF &
     Error
617|    Err = 0
618|    Exit Sub
```

```
619|     Resume
620|
621| End Select
622|
623|
624| End Sub
625|
```


Wheat Status
File Window
gnlDataBaseName
Themes
Symbols
Lines
Tiles
Polygons
Labels
DDatabounds
NumberOfPoints
Memory

```
VERSION 2.00
Begin Form frmStatus
  BackColor      = &H00C0C0C0&
  Caption        = "Wheat Status"
  ClientHeight   = 4995
  ClientLeft     = 3870
  ClientTop      = 1275
  ClientWidth    = 4680
  Height         = 5685
  Left           = 3810
  LinkTopic      = "Form1"
  MDIChild       = -1 'True
  ScaleHeight    = 4995
  ScaleWidth     = 4680
  Top            = 645
  Width          = 4800
  Begin SSPanel zzpnlHost
    Alignment      = 0 'Left Justify - TOP
    BackColor      = &H00C0C0C0&
    BorderWidth    = 1
    Font3D         = 0 'None
    ForeColor      = &H00000000&
    Height         = 5000
    Left           = 0
    TabIndex       = 0
    Top            = 0
    Width          = 4695
    Begin SSPanel DDataBounds
      Alignment      = 0 'Left Justify - TOP
      BackColor      = &H00C0C0C0&
      BevelInner     = 1 'Inset
      BevelOuter     = 0 'None
      BorderWidth    = 1
      Caption        = "DDatabounds"
      Font3D         = 0 'None
      FontBold       = 0 'False
      FontItalic     = 0 'False
      FontName       = "MS Sans Serif"
      FontSize       = 8.25
      FontStrikethru = 0 'False
      FontUnderline  = 0 'False
      ForeColor      = &H00000000&
      Height         = 450
      Left           = 30
      TabIndex       = 15
      Top            = 3700
      Width          = 4425
    End
    Begin PictureBox picTrigger
      BackColor      = &H00C0C0C0&
      BorderStyle    = 0 'None
      Height         = 375
      Left           = 270
      ScaleHeight    = 375
```

```
55|       ScaleWidth    = 4245
56|       TabIndex      = 14
57|       Top           = 3600
58|       Width         = 4245
59|     End
60|     Begin SSPanel NNumberOfPoints
61|       Alignment     = 0 'Left Justify - TOP
62|       BackColor     = &H00C0C0C0&
63|       BevelInner    = 1 'Inset
64|       BevelOuter    = 0 'None
65|       BorderWidth   = 1
66|       Caption       = "NumberOfPoints"
67|       Font3D        = 0 'None
68|       FontBold      = 0 'False
69|       FontItalic    = 0 'False
70|       FontName      = "MS Sans Serif"
71|       FontSize      = 8.25
72|       FontStrikethru = 0 'False
73|       FontUnderline = 0 'False
74|       ForeColor     = &H00000000&
75|       Height        = 285
76|       Left          = 30
77|       TabIndex      = 11
78|       Top           = 4400
79|       Width         = 4425
80|     End
81|     Begin SSPanel Activity
82|       Alignment     = 0 'Left Justify - TOP
83|       BackColor     = &H00C0C0C0&
84|       BevelInner    = 1 'Inset
85|       BevelOuter    = 0 'None
86|       BorderWidth   = 1
87|       Font3D        = 0 'None
88|       FontBold      = 0 'False
89|       FontItalic    = 0 'False
90|       FontName      = "MS Sans Serif"
91|       FontSize      = 8.25
92|       FontStrikethru = 0 'False
93|       FontUnderline = 0 'False
94|       ForeColor     = &H00000000&
95|       Height        = 670
96|       Left          = 30
97|       TabIndex      = 10
98|       Top           = 2700
99|       Width         = 4515
100|    End
101|    Begin SSPanel PercentDone
102|      Alignment     = 0 'Left Justify - TOP
103|      BackColor     = &H00C0C0C0&
104|      BevelInner    = 1 'Inset
105|      BevelOuter    = 0 'None
106|      BorderWidth   = 1
107|      Caption       = "Panel3D8"
108|      FloodType     = 1 'Left To Right
```

```
109|       Font3D       = 0 'None
110|       FontBold     = 0 'False
111|       FontItalic   = 0 'False
112|       FontName     = "MS Sans Serif"
113|       FontSize     = 8.25
114|       FontStrikethru = 0 'False
115|       FontUnderline = 0 'False
116|       ForeColor    = &H00000000&
117|       Height       = 285
118|       Left         = 30
119|       TabIndex     = 9
120|       Top          = 2400
121|       Width        = 4630
122|     End
123|     Begin SSPanel MMemory
124|       Alignment    = 0 'Left Justify - TOP
125|       BackColor    = &H00C0C0C0&
126|       BevelInner   = 1 'Inset
127|       BevelOuter   = 0 'None
128|       BorderWidth  = 1
129|       Caption      = "Memory "
130|       Font3D       = 0 'None
131|       FontBold     = 0 'False
132|       FontItalic   = 0 'False
133|       FontName     = "MS Sans Serif"
134|       FontSize     = 8.25
135|       FontStrikethru = 0 'False
136|       FontUnderline = 0 'False
137|       ForeColor    = &H00000000&
138|       Height       = 285
139|       Left         = 30
140|       TabIndex     = 8
141|       Top          = 4700
142|       Width        = 4630
143|     End
144|     Begin SSPanel LLabels
145|       Alignment    = 0 'Left Justify - TOP
146|       BackColor    = &H00C0C0C0&
147|       BevelInner   = 1 'Inset
148|       BevelOuter   = 0 'None
149|       BorderWidth  = 1
150|       Caption      = "Labels"
151|       Font3D       = 0 'None
152|       FontBold     = 0 'False
153|       FontItalic   = 0 'False
154|       FontName     = "MS Sans Serif"
155|       FontSize     = 8.25
156|       FontStrikethru = 0 'False
157|       FontUnderline = 0 'False
158|       ForeColor    = &H00000000&
159|       Height       = 285
160|       Left         = 30
161|       TabIndex     = 7
162|       Top          = 1800
```

```
163|       Width         = 4630
164|     End
165|     Begin SSPanel PPolygons
166|       Alignment     = 0 'Left Justify - TOP
167|       BackColor     = &H00C0C0C0&
168|       BevelInner    = 1 'Inset
169|       BevelOuter    = 0 'None
170|       BorderWidth   = 1
171|       Caption       = "Polygons"
172|       Font3D        = 0 'None
173|       FontBold      = 0 'False
174|       FontItalic    = 0 'False
175|       FontName      = "MS Sans Serif"
176|       FontSize      = 8.25
177|       FontStrikethru = 0 'False
178|       FontUnderline = 0 'False
179|       ForeColor     = &H00000000&
180|       Height        = 285
181|       Left          = 30
182|       TabIndex      = 6
183|       Top           = 1500
184|       Width         = 4630
185|     End
186|     Begin SSPanel TTiles
187|       Alignment     = 0 'Left Justify - TOP
188|       BackColor     = &H00C0C0C0&
189|       BevelInner    = 1 'Inset
190|       BevelOuter    = 0 'None
191|       BorderWidth   = 1
192|       Caption       = "Tiles"
193|       Font3D        = 0 'None
194|       FontBold      = 0 'False
195|       FontItalic    = 0 'False
196|       FontName      = "MS Sans Serif"
197|       FontSize      = 8.25
198|       FontStrikethru = 0 'False
199|       FontUnderline = 0 'False
200|       ForeColor     = &H00000000&
201|       Height        = 285
202|       Left          = 30
203|       TabIndex      = 5
204|       Top           = 1200
205|       Width         = 4630
206|     End
207|     Begin SSPanel LLines
208|       Alignment     = 0 'Left Justify - TOP
209|       BackColor     = &H00C0C0C0&
210|       BevelInner    = 1 'Inset
211|       BevelOuter    = 0 'None
212|       BorderWidth   = 1
213|       Caption       = "Lines"
214|       Font3D        = 0 'None
215|       FontBold      = 0 'False
216|       FontItalic    = 0 'False
```

```
217|       FontName      = "MS Sans Serif"
218|       FontSize      = 8.25
219|       FontStrikethru = 0  'False
220|       FontUnderline  = 0  'False
221|       ForeColor     = &H00000000&
222|       Height        = 285
223|       Left          = 30
224|       TabIndex      = 4
225|       Top           = 900
226|       Width         = 4630
227|     End
228|     Begin SSPanel SSymbols
229|       Alignment     = 0  'Left Justify - TOP
230|       BackColor     = &H00C0C0C0&
231|       BevelInner    = 1  'Inset
232|       BevelOuter    = 0  'None
233|       BorderWidth   = 1
234|       Caption       = "Symbols"
235|       Font3D        = 0  'None
236|       FontBold      = 0  'False
237|       FontItalic    = 0  'False
238|       FontName      = "MS Sans Serif"
239|       FontSize      = 8.25
240|       FontStrikethru = 0  'False
241|       FontUnderline  = 0  'False
242|       ForeColor     = &H00000000&
243|       Height        = 285
244|       Left          = 30
245|       TabIndex      = 3
246|       Top           = 600
247|       Width         = 4630
248|     End
249|     Begin SSPanel TThemes
250|       Alignment     = 0  'Left Justify - TOP
251|       BackColor     = &H00C0C0C0&
252|       BevelInner    = 1  'Inset
253|       BevelOuter    = 0  'None
254|       BorderWidth   = 1
255|       Caption       = "Themes"
256|       Font3D        = 0  'None
257|       FontBold      = 0  'False
258|       FontItalic    = 0  'False
259|       FontName      = "MS Sans Serif"
260|       FontSize      = 8.25
261|       FontStrikethru = 0  'False
262|       FontUnderline  = 0  'False
263|       ForeColor     = &H00000000&
264|       Height        = 285
265|       Left          = 30
266|       TabIndex      = 2
267|       Top           = 300
268|       Width         = 4630
269|     End
270|     Begin SSPanel DataBaseName
```

```
271|       Alignment     =  0  'Left Justify - TOP
272|       BackColor     =  &H00C0C0C0&
273|       BevelInner    =  1  'Inset
274|       BevelOuter    =  0  'None
275|       BorderWidth   =  1
276|       Caption       =  "pnlDataBaseName"
277|       Font3D        =  0  'None
278|       FontBold      =  0  'False
279|       FontItalic    =  0  'False
280|       FontName      =  "MS Sans Serif"
281|       FontSize      =  8.25
282|       FontStrikethru =  0  'False
283|       FontUnderline =  0  'False
284|       ForeColor     =  &H00000000&
285|       Height        =  270
286|       Left          =  30
287|       TabIndex      =  1
288|       Top           =  30
289|       Width         =  4630
290|     End
291|   End
292|   Begin Label lblPercentageDone
293|     Height        =  195
294|     Left          =  4950
295|     TabIndex      =  12
296|     Top           =  2610
297|     Width         =  1275
298|   End
299|   Begin Label lblActivity
300|     Height        =  285
301|     Left          =  4950
302|     TabIndex      =  13
303|     Top           =  2970
304|     Width         =  1275
305|   End
306|   Begin Menu zmnuFile
307|     Caption       =  "&File"
308|     Begin Menu mfExit
309|       Caption     =  "E&xit"
310|     End
311|   End
312|   Begin Menu zmnuWindow
313|     Caption       =  "&Window"
314|     WindowList    =  -1  'True
315|   End
316| End
317| Option Explicit
318|
319| Sub Form_Load ()
320|  Me.Top = 45
321|  Me.Left = 5800
322|
323|  Me.Height = 5600
324|  Me.Width = 3600
```

```
End Sub

Sub Form_Resize ()
    zzpnlHost.Width = Me.ScaleWidth
    zzpnlHost.Height = Me.ScaleHeight
    Dim PNLWidth As Single
    PNLWidth = zzpnlHost.Width - 60
    DataBaseName.Width = PNLWidth
    TThemes.Width = PNLWidth
    SSymbols.Width = PNLWidth
    LLines.Width = PNLWidth
    TTiles.Width = PNLWidth
    PPolygons.Width = PNLWidth
    LLabels.Width = PNLWidth
    PercentDone.Width = PNLWidth
    Activity.Width = PNLWidth
    DDataBounds.Width = PNLWidth

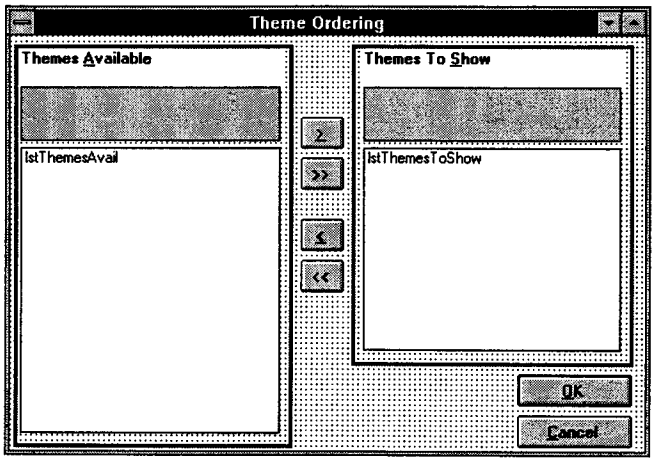
    MMemory.Width = PNLWidth
    NNumberOfPoints.Width = PNLWidth
End Sub

Sub lblPercentageDone_Change ()
  Static PercentageDone As Integer, LastPercentDone As Integer
  PercentageDone = lblPercentageDone
  If PercentageDone < 0 Then PercentageDone = 0
  If PercentageDone > 100 Then PercentageDone = 100
  If PercentageDone < 100 Then PercentDone.FloodType = 1
  If PercentageDone <> LastPercentDone Then
     PercentDone.FloodPercent = PercentageDone
     LastPercentDone = PercentageDone
  End If
  If PercentageDone = 100 Then
        PercentDone.FloodType = 0
        PercentDone.Caption = "DONE"
  End If
End Sub

Sub mfExit_Click ()
  ShutDownAndFreeMem
End Sub

Sub picTrigger_Paint ()
 DataBaseName = "Current Database is " & strMapDBName
 TThemes = "There are currently " & nThemes & " themes loaded."
 SSymbols = "There are " & UBound(Points) & " symbols loaded."
 LLines = "There are " & UBound(Lines) & " lines loaded."
 TTiles = "There are " & UBound(Tiles) & " tiles loaded."
 PPolygons = "There are " & UBound(Polys) & " polygons loaded."
 LLabels = "There are " & UBound(Labels) & " labels loaded."
 NNumberOfPoints = "There are " & hGMemLastPoint & " coordinate pairs stored in
memory."
 MMemory = "Currently using " & lngHGAllPointsMemSize & " bytes of memory.
Roughly " & GetFreeSpace(1) & " still available."
```

```
377|  DDataBounds = "Data Range X: " & DataBounds.Right & " to " & DataBounds.Left & " ;
      Y " & DataBounds.Bottom & " to " & DataBounds.Top
378|  End Sub
379|
```


Theme Ordering
Themes Available
lstThemesAvail
>
>>
<
<<
Themes To Show
lstThemesToShow
OK
Cancel

```
VERSION 2.00
Begin Form frmThemeOrder
  Caption       = "Theme Ordering"
  ClientHeight  = 4860
  ClientLeft    = 1095
  ClientTop     = 1485
  ClientWidth   = 7365
  ControlBox    = 0  'False
  Height        = 5265
  Left          = 1035
  LinkTopic     = "Form1"
  MaxButton     = 0  'False
  MinButton     = 0  'False
  ScaleHeight   = 85.725
  ScaleMode     = 6  'Millimeter
  ScaleWidth    = 129.911
  Top           = 1140
  Width         = 7485
  Begin CommandButton cmdCancel
    Cancel      = -1  'True
    Caption     = "&Cancel"
    Height      = 375
    Left        = 5880
    TabIndex    = 11
    Top         = 4440
    Width       = 1335
  End
  Begin CommandButton cmdOK
    Caption     = "&OK"
    Default     = -1  'True
    Height      = 375
    Left        = 5880
    TabIndex    = 10
    Top         = 3960
    Width       = 1335
  End
  Begin CommandButton cmdRemoveAll
    Caption     = "<<"
    Height      = 375
    Left        = 3360
    TabIndex    = 7
    Top         = 2640
    Width       = 495
  End
  Begin CommandButton cmdRemoveOne
    Caption     = "&<"
    Height      = 375
    Left        = 3360
    TabIndex    = 6
    Top         = 2160
    Width       = 495
  End
  Begin CommandButton cmdAddOne
    Caption     = "&>"
```

```
   Height      = 375
   Left        = 3360
   TabIndex    = 5
   Top         = 960
   Width       = 495
End
Begin CommandButton cmdAddAll
   Caption     = ">>"
   Height      = 375
   Left        = 3360
   TabIndex    = 4
   Top         = 1440
   Width       = 495
End
Begin ListBox lstThemesToShow
   FontBold      = 0  'False
   FontItalic    = 0  'False
   FontName      = "MS Sans Serif"
   FontSize      = 8.25
   FontStrikethru = 0  'False
   FontUnderline = 0  'False
   Height        = 2370
   Left          = 4080
   TabIndex      = 1
   Top           = 1320
   Width         = 3000
End
Begin ListBox lstThemesAvail
   FontBold      = 0  'False
   FontItalic    = 0  'False
   FontName      = "MS Sans Serif"
   FontSize      = 8.25
   FontStrikethru = 0  'False
   FontUnderline = 0  'False
   Height        = 3345
   Left          = 120
   TabIndex      = 0
   Top           = 1320
   Width         = 3000
End
Begin Label lblCurrentUsed
   BackColor     = &H00C0C0C0&
   BorderStyle   = 1  'Fixed Single
   FontBold      = 0  'False
   FontItalic    = 0  'False
   FontName      = "MS Sans Serif"
   FontSize      = 8.25
   FontStrikethru = 0  'False
   FontUnderline = 0  'False
   ForeColor     = &H00000000&
   Height        = 645
   Left          = 4080
   TabIndex      = 9
   Top           = 600
```

```
109|     Width         =  3000
110|     WordWrap      =  -1  'True
111|   End
112|   Begin Label ZlblThemesToShow
113|     Caption       =  "Themes To &Show"
114|     Height        =  495
115|     Left          =  4080
116|     TabIndex      =  8
117|     Top           =  165
118|     Width         =  3015
119|   End
120|   Begin Shape Shape2
121|     BorderColor   =  &H000000FF&
122|     BorderWidth   =  3
123|     Height        =  3735
124|     Left          =  3960
125|     Top           =  120
126|     Width         =  3255
127|   End
128|   Begin Label lblCurrentItem
129|     BackColor     =  &H00C0C0C0&
130|     BorderStyle   =  1  'Fixed Single
131|     FontBold      =  0  'False
132|     FontItalic    =  0  'False
133|     FontName      =  "MS Sans Serif"
134|     FontSize      =  8.25
135|     FontStrikethru =  0  'False
136|     FontUnderline =  0  'False
137|     ForeColor     =  &H00000000&
138|     Height        =  645
139|     Left          =  120
140|     TabIndex      =  3
141|     Top           =  600
142|     Width         =  3000
143|     WordWrap      =  -1  'True
144|   End
145|   Begin Shape Shape1
146|     BorderColor   =  &H00FF0000&
147|     BorderWidth   =  3
148|     Height        =  4695
149|     Left          =  60
150|     Top           =  120
151|     Width         =  3195
152|   End
153|   Begin Label ZlblThemesAvailable
154|     Caption       =  "Themes &Available"
155|     Height        =  495
156|     Left          =  120
157|     TabIndex      =  2
158|     Top           =  165
159|     Width         =  3015
160|   End
161| End
162| Option Explicit
```

```
Dim N As Long

Sub cmdAddAll_Click ()
 For N = 0 To lstThemesAvail.ListCount - 1
    lstThemesToShow.AddItem lstThemesAvail.List(N)
 Next N
End Sub

Sub cmdAddOne_Click ()
If lstThemesToShow.ListIndex = -1 Then
    lstThemesToShow.AddItem lstThemesAvail
   Else
    lstThemesToShow.AddItem lstThemesAvail, lstThemesToShow.ListIndex
End If
End Sub

Sub cmdCancel_Click ()
 cmdCancel.SetFocus
 Me.Hide
End Sub

Sub cmdOK_Click ()
 cmdOK.SetFocus
 Me.Hide
 cmdOK.FontItalic = True
End Sub

Sub cmdRemoveAll_Click ()
 lstThemesToShow.Clear
End Sub

Sub cmdRemoveOne_Click ()
On Error Resume Next
 lstThemesToShow.RemoveItem lstThemesToShow.ListIndex

'xxx do something about ERR=5

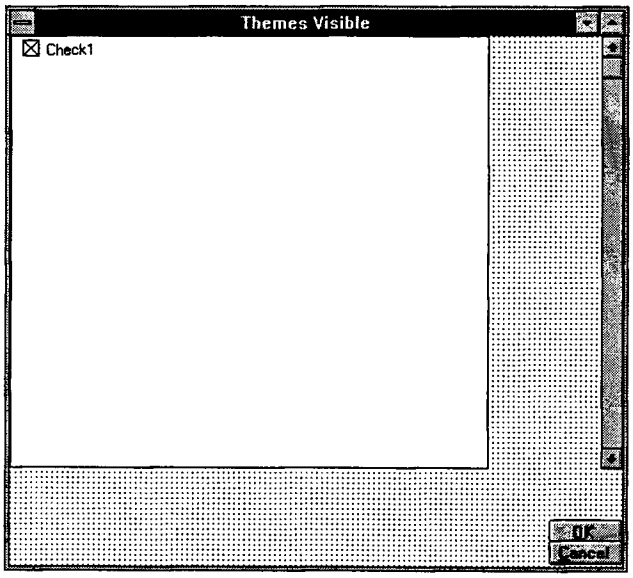
End Sub

Sub Form_Paint ()
  cmdOK.FontItalic = False
  'ZlblThemesAvailable.AutoSize = True
  ZlblThemesAvailable.WordWrap = True
  'ZlblThemesToShow.AutoSize = True
  ZlblThemesToShow.WordWrap = True
End Sub

Sub lstThemesAvail_Click ()
    lblCurrentItem = lstThemesAvail
End Sub

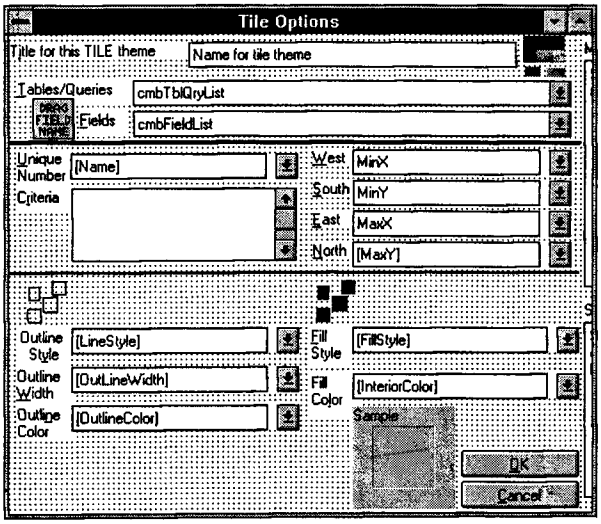
Sub lstThemesAvail_DblClick ()
 If lstThemesAvail.ListIndex <> -1 Then Call cmdAddOne_Click
End Sub
```

```
217|
218|  Sub lstThemesToShow_Click ()
219|   lblCurrentUsed = lstThemesToShow
220|  End Sub
221|
222|  Sub lstThemesToShow_DblClick ()
223|   If lstThemesToShow.ListIndex <> -1 Then Call cmdRemoveOne_Click
224|  End Sub
225|
```


Themes Visible
Check1
OK
Cancel

```
VERSION 2.00
Begin Form frmThemesVisible
  Caption        = "Themes Visible"
  ClientHeight   = 6195
  ClientLeft     = 510
  ClientTop      = 585
  ClientWidth    = 7095
  ControlBox     = 0  'False
  Height         = 6600
  Left           = 450
  LinkTopic      = "Form1"
  MaxButton      = 0  'False
  MinButton      = 0  'False
  ScaleHeight    = 109.273
  ScaleMode      = 6  'Millimeter
  ScaleWidth     = 125.148
  Top            = 240
  Width          = 7215
  Begin CommandButton cmdCancel
    Caption      = "&Cancel"
    Height       = 278
    Left         = 6240
    TabIndex     = 4
    Top          = 5880
    Width        = 855
  End
  Begin CommandButton cmdOK
    Caption      = "&OK"
    Height       = 278
    Left         = 6240
    TabIndex     = 3
    Top          = 5640
    Width        = 855
  End
  Begin VScrollBar VScroll
    Height       = 5055
    LargeChange  = 20
    Left         = 6840
    SmallChange  = 5
    TabIndex     = 1
    Top          = 0
    Width        = 255
  End
  Begin PictureBox picHolder
    Height       = 5055
    Left         = 0
    ScaleHeight  = 88.636
    ScaleMode    = 6  'Millimeter
    ScaleWidth   = 97.102
    TabIndex     = 0
    Top          = 0
    Width        = 5535
    Begin CheckBox chkThemeVisible
      Caption    = "Check1"
```

```
55|        FontBold      =  0  'False
56|        FontItalic    =  0  'False
57|        FontName      =  "MS Sans Serif"
58|        FontSize      =  8.25
59|        FontStrikethru =  0  'False
60|        FontUnderline  =  0  'False
61|        Height        =  255
62|        Index         =  0
63|        Left          =  120
64|        TabIndex      =  2
65|        Top           =  0
66|        Value         =  1  'Checked
67|        Width         =  5295
68|     End
69|   End
70| End
71| Option Explicit
72|
73| Sub cmdCancel_Click ()
74|  cmdOK.FontItalic = False
75|  Me.Hide
76| End Sub
77|
78| Sub cmdOK_Click ()
79|  Me.Hide
80|  Me.cmdOK.FontItalic = True
81| End Sub
82|
83| Sub Form_Resize ()
84|   'picHolder.height = Me.scaleheight - 10
85|   picHolder.Width = Me.ScaleWidth - 5
86|   VScroll.Height = Me.ScaleHeight - 10
87|   VScroll.Left = Me.ScaleWidth - 4.75
88|   cmdOK.Top = Me.ScaleHeight - 10
89|   cmdCancel.Top = Me.ScaleHeight - 5
90| End Sub
91|
92| Sub picHolder_Click ()
93|  cmdOK.FontItalic = False
94| End Sub
95|
96| Sub VScroll_Change ()
97|    Me.picHolder.Top = -VScroll
98| End Sub
99|
```


Tile Options
Title for this TILE theme
Name for tile theme
Tables/Queries
cmbTblQryList
Fields
cmbFieldList
Unique Number
[Name]
Criteria
West
MinX
South
MinY
East
MaxX
North
[MaxY]
Outline Style
[LineStyle]
Outline Width
[OutLineWidth]
Outline Color
[OutlineColor]
Fill Style
[FillStyle]
Fill Color
[InteriorColor]
Sample
OK
Cancel

```
VERSION 2.00
Begin Form frmTileOptions
  Caption        = "Tile Options"
  ClientHeight   = 5580
  ClientLeft     = 105
  ClientTop      = 450
  ClientWidth    = 6735
  ControlBox     = 0  'False
  Height         = 5985
  Icon           = TILEOPT7.FRX:0000
  Left           = 45
  LinkTopic      = "Form1"
  MinButton      = 0  'False
  ScaleHeight    = 5580
  ScaleWidth     = 6735
  Top            = 105
  Width          = 6855
  Begin SSPanel zpnlOuter
    Alignment      = 0  'Left Justify - TOP
    AutoSize       = 3  'AutoSize Child To Panel
    BackColor      = &H00C0C0C0&
    BevelInner     = 1  'Inset
    BevelOuter     = 0  'None
    BorderWidth    = 15
    Font3D         = 0  'None
    ForeColor      = &H00FFFFFF&
    Height         = 1185
    Left           = 3960
    TabIndex       = 34
    Top            = 4350
    Width          = 1185
    Begin SSPanel zpnlSampleWord
      Alignment      = 0  'Left Justify - TOP
      AutoSize       = 3  'AutoSize Child To Panel
      BackColor      = &H00C0C0C0&
      BevelOuter     = 0  'None
      BorderWidth    = 1
      Caption        = "Sample"
      Font3D         = 0  'None
      FontBold       = 0  'False
      FontItalic     = 0  'False
      FontName       = "MS Sans Serif"
      FontSize       = 8.25
      FontStrikethru = 0  'False
      FontUnderline  = 0  'False
      ForeColor      = &H00000000&
      Height         = 220
      Left           = 0
      TabIndex       = 36
      Top            = 0
      Width          = 735
    End
    Begin PictureBox picSample
      BackColor      = &H00C0C0C0&
```

```
55|       BorderStyle    = 0  'None
56|       Height         = 700
57|       Left           = 240
58|       ScaleHeight    = 47
59|       ScaleMode      = 3  'Pixel
60|       ScaleWidth     = 47
61|       TabIndex       = 35
62|       Top            = 240
63|       Width          = 700
64|     End
65|   End
66|   Begin ComboBox cmb_zqFeatureID
67|     FontBold      = 0  'False
68|     FontItalic    = 0  'False
69|     FontName      = "MS Sans Serif"
70|     FontSize      = 8.25
71|     FontStrikethru = 0  'False
72|     FontUnderline = 0  'False
73|     Height        = 300
74|     HelpContextID = 1500
75|     Left          = 705
76|     TabIndex      = 7
77|     Text          = "[Name]"
78|     Top           = 1395
79|     Width         = 2625
80|   End
81|   Begin ComboBox cmb_zqOutlineStyle
82|     FontBold      = 0  'False
83|     FontItalic    = 0  'False
84|     FontName      = "MS Sans Serif"
85|     FontSize      = 8.25
86|     FontStrikethru = 0  'False
87|     FontUnderline = 0  'False
88|     Height        = 300
89|     HelpContextID = 2200
90|     Left          = 720
91|     TabIndex      = 19
92|     Text          = "[LineStyle]"
93|     Top           = 3420
94|     Width         = 2625
95|   End
96|   Begin ComboBox cmb_zqOutlineWidth
97|     FontBold      = 0  'False
98|     FontItalic    = 0  'False
99|     FontName      = "MS Sans Serif"
100|    FontSize      = 8.25
101|    FontStrikethru = 0  'False
102|    FontUnderline = 0  'False
103|    Height        = 300
104|    HelpContextID = 2100
105|    Left          = 720
106|    TabIndex      = 21
107|    Text          = "[OutLineWidth]"
108|    Top           = 3870
```

```
109|     Width         = 2625
110|   End
111|   Begin ComboBox cmb_zqFillColor
112|     FontBold      = 0  'False
113|     FontItalic    = 0  'False
114|     FontName      = "MS Sans Serif"
115|     FontSize      = 8.25
116|     FontStrikethru = 0  'False
117|     FontUnderline = 0  'False
118|     Height        = 300
119|     HelpContextID = 2000
120|     Left          = 3960
121|     TabIndex      = 27
122|     Text          = "[InteriorColor]"
123|     Top           = 3960
124|     Width         = 2625
125|   End
126|   Begin ComboBox cmb_zqFillStyle
127|     FontBold      = 0  'False
128|     FontItalic    = 0  'False
129|     FontName      = "MS Sans Serif"
130|     FontSize      = 8.25
131|     FontStrikethru = 0  'False
132|     FontUnderline = 0  'False
133|     Height        = 300
134|     HelpContextID = 2300
135|     Left          = 3960
136|     TabIndex      = 25
137|     Text          = "[FillStyle]"
138|     Top           = 3420
139|     Width         = 2625
140|   End
141|   Begin ComboBox cmb_zqOutlineColor
142|     FontBold      = 0  'False
143|     FontItalic    = 0  'False
144|     FontName      = "MS Sans Serif"
145|     FontSize      = 8.25
146|     FontStrikethru = 0  'False
147|     FontUnderline = 0  'False
148|     Height        = 300
149|     HelpContextID = 2000
150|     Left          = 720
151|     TabIndex      = 23
152|     Text          = "[OutlineColor]"
153|     Top           = 4320
154|     Width         = 2625
155|   End
156|   Begin ComboBox cmb_zqMinX
157|     FontBold      = 0  'False
158|     FontItalic    = 0  'False
159|     FontName      = "MS Sans Serif"
160|     FontSize      = 8.25
161|     FontStrikethru = 0  'False
162|     FontUnderline = 0  'False
```

```
163|     Height        = 300
164|     HelpContextID  = 1000
165|     Left          = 3960
166|     TabIndex      = 17
167|     Text          = "MinX"
168|     Top           = 1350
169|     Width         = 2535
170|   End
171|   Begin ComboBox cmb_zqMaxY
172|     FontBold      = 0  'False
173|     FontItalic    = 0  'False
174|     FontName      = "MS Sans Serif"
175|     FontSize      = 8.25
176|     FontStrikethru = 0  'False
177|     FontUnderline = 0  'False
178|     Height        = 300
179|     HelpContextID  = 1100
180|     Left          = 3960
181|     TabIndex      = 11
182|     Text          = "[MaxY]"
183|     Top           = 2430
184|     Width         = 2535
185|   End
186|   Begin ComboBox cmb_zqMaxX
187|     FontBold      = 0  'False
188|     FontItalic    = 0  'False
189|     FontName      = "MS Sans Serif"
190|     FontSize      = 8.25
191|     FontStrikethru = 0  'False
192|     FontUnderline = 0  'False
193|     Height        = 300
194|     HelpContextID  = 1000
195|     Left          = 3960
196|     TabIndex      = 15
197|     Text          = "MaxX"
198|     Top           = 2070
199|     Width         = 2535
200|   End
201|   Begin ComboBox cmb_zqMinY
202|     FontBold      = 0  'False
203|     FontItalic    = 0  'False
204|     FontName      = "MS Sans Serif"
205|     FontSize      = 8.25
206|     FontStrikethru = 0  'False
207|     FontUnderline = 0  'False
208|     Height        = 300
209|     HelpContextID  = 1100
210|     Left          = 3960
211|     TabIndex      = 13
212|     Text          = "MinY"
213|     Top           = 1710
214|     Width         = 2535
215|   End
216|   Begin TextBox txtThemeName
```

```
    FontBold      = 0  'False
    FontItalic    = 0  'False
    FontName      = "MS Sans Serif"
    FontSize      = 8.25
    FontStrikethru = 0  'False
    FontUnderline = 0  'False
    Height        = 300
    Left          = 2070
    TabIndex      = 1
    Text          = "Name for tile theme "
    Top           = 90
    Width         = 3795
  End
  Begin TextBox txtCriteria
    FontBold      = 0  'False
    FontItalic    = 0  'False
    FontName      = "MS Sans Serif"
    FontSize      = 8.25
    FontStrikethru = 0  'False
    FontUnderline = 0  'False
    Height        = 855
    HelpContextID = 1600
    Left          = 720
    MultiLine     = -1  'True
    ScrollBars    = 2  'Vertical
    TabIndex      = 9
    Top           = 1800
    Width         = 2595
  End
  Begin TextBox txtSpatialQuerySQL
    FontBold      = 0  'False
    FontItalic    = 0  'False
    FontName      = "MS Sans Serif"
    FontSize      = 8.25
    FontStrikethru = 0  'False
    FontUnderline = 0  'False
    Height        = 2055
    Left          = 6650
    MultiLine     = -1  'True
    ScrollBars    = 2  'Vertical
    TabIndex      = 33
    Text          = "SQL Statement to produce data displayed on DataBase Form"
    Top           = 3360
    Width         = 2565
  End
  Begin TextBox txtMapSQL
    FontBold      = 0  'False
    FontItalic    = 0  'False
    FontName      = "MS Sans Serif"
    FontSize      = 8.25
    FontStrikethru = 0  'False
    FontUnderline = 0  'False
    Height        = 2655
    Left          = 6650
```

```
271|     MultiLine      = -1 'True
272|     ScrollBars     = 2 'Vertical
273|     TabIndex       = 31
274|     Text           = "SQL statement to produce map data"
275|     Top            = 300
276|     Width          = 2505
277|   End
278|   Begin ComboBox cmbTblQryList
279|     FontBold       = 0 'False
280|     FontItalic     = 0 'False
281|     FontName       = "MS Sans Serif"
282|     FontSize       = 8.25
283|     FontStrikethru = 0 'False
284|     FontUnderline  = 0 'False
285|     Height         = 300
286|     Left           = 1425
287|     Style          = 2 'Dropdown List
288|     TabIndex       = 3
289|     Top            = 540
290|     Width          = 5070
291|   End
292|   Begin CommandButton cmdCancel
293|     Caption        = "&Cancel"
294|     FontBold       = 0 'False
295|     FontItalic     = 0 'False
296|     FontName       = "MS Sans Serif"
297|     FontSize       = 8.25
298|     FontStrikethru = 0 'False
299|     FontUnderline  = 0 'False
300|     Height         = 315
301|     Left           = 5220
302|     TabIndex       = 29
303|     Top            = 5220
304|     Width          = 1365
305|   End
306|   Begin CommandButton cmdOK
307|     Caption        = "&OK"
308|     FontBold       = 0 'False
309|     FontItalic     = 0 'False
310|     FontName       = "MS Sans Serif"
311|     FontSize       = 8.25
312|     FontStrikethru = 0 'False
313|     FontUnderline  = 0 'False
314|     Height         = 315
315|     Left           = 5220
316|     TabIndex       = 28
317|     Top            = 4860
318|     Width          = 1365
319|   End
320|   Begin ComboBox cmbFieldList
321|     DragIcon       = TILEOPT7.FRX:0302
322|     FontBold       = 0 'False
323|     FontItalic     = 0 'False
324|     FontName       = "MS Sans Serif"
```

```
325|     FontSize      = 8.25
326|     FontStrikethru = 0  'False
327|     FontUnderline  = 0  'False
328|     Height        = 300
329|     Left          = 1425
330|     Sorted        = -1 'True
331|     Style         = 2  'Dropdown List
332|     TabIndex      = 5
333|     Top           = 900
334|     Width         = 5070
335|   End
336|   Begin Label ACTION
337|     BackColor     = &H0000FFFF&
338|     ForeColor     = &H000000FF&
339|     Height        = 195
340|     Left          = 960
341|     TabIndex      = 37
342|     Top           = 2100
343|     Visible       = 0  'False
344|     Width         = 1995
345|   End
346|   Begin Image imgFieldName
347|     Height        = 480
348|     Left          = 270
349|     Picture       = TILEOPT7.FRX:0604
350|     Top           = 750
351|     Width         = 480
352|   End
353|   Begin Label zzzlblName
354|     Caption       = "&Unique Number"
355|     FontBold      = 0  'False
356|     FontItalic    = 0  'False
357|     FontName      = "MS Sans Serif"
358|     FontSize      = 8.25
359|     FontStrikethru = 0  'False
360|     FontUnderline  = 0  'False
361|     Height        = 405
362|     Left          = 90
363|     TabIndex      = 6
364|     Top           = 1350
365|     Width         = 615
366|   End
367|   Begin Label zzzlblZSymbol
368|     Alignment     = 2  'Center
369|     Caption       = "Outline St&yle"
370|     FontBold      = 0  'False
371|     FontItalic    = 0  'False
372|     FontName      = "MS Sans Serif"
373|     FontSize      = 8.25
374|     FontStrikethru = 0  'False
375|     FontUnderline  = 0  'False
376|     Height        = 435
377|     Left          = 90
378|     TabIndex      = 18
```

```
379|     Top          = 3420
380|     Width        = 615
381|   End
382|   Begin Image zzImageInteriors
383|     Height       = 480
384|     Left         = 3510
385|     Picture      = TILEOPT7.FRX:0906
386|     Top          = 2880
387|     Width        = 480
388|   End
389|   Begin Image zzImageOutlines
390|     Height       = 480
391|     Left         = 180
392|     Picture      = TILEOPT7.FRX:0C08
393|     Top          = 2880
394|     Width        = 480
395|   End
396|   Begin Label zzzlblZSize
397|     Caption      = "Outline &Width"
398|     FontBold     = 0  'False
399|     FontItalic   = 0  'False
400|     FontName     = "MS Sans Serif"
401|     FontSize     = 8.25
402|     FontStrikethru = 0  'False
403|     FontUnderline  = 0  'False
404|     Height       = 405
405|     Left         = 90
406|     TabIndex     = 20
407|     Top          = 3870
408|     Width        = 675
409|   End
410|   Begin Label zlblFillColor
411|     Caption      = "Fill Co&lor"
412|     FontBold     = 0  'False
413|     FontItalic   = 0  'False
414|     FontName     = "MS Sans Serif"
415|     FontSize     = 8.25
416|     FontStrikethru = 0  'False
417|     FontUnderline  = 0  'False
418|     Height       = 495
419|     Left         = 3480
420|     TabIndex     = 26
421|     Top          = 3960
422|     Width        = 555
423|   End
424|   Begin Label zlblFillStyle
425|     Caption      = "&Fill Style"
426|     FontBold     = 0  'False
427|     FontItalic   = 0  'False
428|     FontName     = "MS Sans Serif"
429|     FontSize     = 8.25
430|     FontStrikethru = 0  'False
431|     FontUnderline  = 0  'False
432|     Height       = 465
```

```
433|     Left        = 3480
434|     TabIndex    = 24
435|     Top         = 3420
436|     Width       = 465
437|   End
438|   Begin Label zlblOutlineColor
439|     Caption     = "Outli&ne Color"
440|     FontBold    = 0  'False
441|     FontItalic  = 0  'False
442|     FontName    = "MS Sans Serif"
443|     FontSize    = 8.25
444|     FontStrikethru = 0  'False
445|     FontUnderline  = 0  'False
446|     Height      = 375
447|     Left        = 90
448|     TabIndex    = 22
449|     Top         = 4320
450|     Width       = 645
451|   End
452|   Begin Line zzLine2
453|     BorderWidth   = 2
454|     X1          = 0
455|     X2          = 6570
456|     Y1          = 2790
457|     Y2          = 2790
458|   End
459|   Begin Label lblWest
460|     Caption     = "&West"
461|     FontBold    = 0  'False
462|     FontItalic  = 0  'False
463|     FontName    = "MS Sans Serif"
464|     FontSize    = 8.25
465|     FontStrikethru = 0  'False
466|     FontUnderline  = 0  'False
467|     Height      = 195
468|     Left        = 3495
469|     TabIndex    = 16
470|     Top         = 1350
471|     Width       = 555
472|   End
473|   Begin Label lblEast
474|     Caption     = "&East"
475|     FontBold    = 0  'False
476|     FontItalic  = 0  'False
477|     FontName    = "MS Sans Serif"
478|     FontSize    = 8.25
479|     FontStrikethru = 0  'False
480|     FontUnderline  = 0  'False
481|     Height      = 195
482|     Left        = 3495
483|     TabIndex    = 14
484|     Top         = 2070
485|     Width       = 555
486|   End
```

```
Begin Label lblSouth
  Caption       = "&South"
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 195
  Left          = 3500
  TabIndex      = 12
  Top           = 1710
  Width         = 555
End
Begin Label lblNorth
  Caption       = "&North"
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 195
  Left          = 3495
  TabIndex      = 10
  Top           = 2430
  Width         = 555
End
Begin Line zzLine1
  BorderWidth   = 3
  X1            = 0
  X2            = 6570
  Y1            = 1260
  Y2            = 1260
End
Begin Image zIMGExample
  Height        = 480
  Left          = 5940
  Picture       = TILEOPT7.FRX:0F0A
  Top           = 15
  Width         = 480
End
Begin Label lblThemeName
  Caption       = "T&itle for this TILE theme"
  FontBold      = 0  'False
  FontItalic    = 0  'False
  FontName      = "MS Sans Serif"
  FontSize      = 8.25
  FontStrikethru = 0  'False
  FontUnderline = 0  'False
  Height        = 210
  Left          = 0
  TabIndex      = 0
  Top           = 90
```

```
541|     Width         = 2175
542|   End
543|   Begin Label zzzzlblCriteia
544|     Caption       = "C&riteria"
545|     FontBold      = 0  'False
546|     FontItalic    = 0  'False
547|     FontName      = "MS Sans Serif"
548|     FontSize      = 8.25
549|     FontStrikethru = 0  'False
550|     FontUnderline = 0  'False
551|     Height        = 255
552|     Left          = 90
553|     TabIndex      = 8
554|     Top           = 1800
555|     Width         = 615
556|   End
557|   Begin Label zzzzlblDBFormSQL
558|     Caption       = "Spatial Query SQL"
559|     FontBold      = 0  'False
560|     FontItalic    = 0  'False
561|     FontName      = "MS Sans Serif"
562|     FontSize      = 8.25
563|     FontStrikethru = 0  'False
564|     FontUnderline = 0  'False
565|     Height        = 465
566|     Left          = 6650
567|     TabIndex      = 32
568|     Top           = 3120
569|     Width         = 1500
570|   End
571|   Begin Label zzzzlblMapTileSQL
572|     Caption       = "Map Tile SQL"
573|     FontBold      = 0  'False
574|     FontItalic    = 0  'False
575|     FontName      = "MS Sans Serif"
576|     FontSize      = 8.25
577|     FontStrikethru = 0  'False
578|     FontUnderline = 0  'False
579|     Height        = 465
580|     Left          = 6650
581|     TabIndex      = 30
582|     Top           = 60
583|     Width         = 1500
584|   End
585|   Begin Label zzzlblFields
586|     Caption       = "&Fields"
587|     FontBold      = 0  'False
588|     FontItalic    = 0  'False
589|     FontName      = "MS Sans Serif"
590|     FontSize      = 8.25
591|     FontStrikethru = 0  'False
592|     FontUnderline = 0  'False
593|     Height        = 285
594|     Left          = 810
```

```
595|      TabIndex      = 4
596|      Top           = 900
597|      Width         = 555
598|    End
599|    Begin Label zzzlblTables
600|      Caption       = "&Tables/Queries"
601|      FontBold      = 0  'False
602|      FontItalic    = 0  'False
603|      FontName      = "MS Sans Serif"
604|      FontSize      = 8.25
605|      FontStrikethru = 0  'False
606|      FontUnderline = 0  'False
607|      Height        = 285
608|      Left          = 90
609|      TabIndex      = 2
610|      Top           = 540
611|      Width         = 1305
612|    End
613| End
614| Option Explicit
615| Option Compare Text
616| Dim flgValidSyntax As Integer
617| Dim ListOfFieldsForDBForm As String
618|
619| Dim flgActivateDragDrop As Integer
620| Dim tblTileMap As Table
621| Dim FieldsInCurrentQT() As String, FieldListWBrackets() As String
622|
623| Dim flgIgnoreMapSQLChange As Long
624|
625| Sub ACTION_Change ()
626|   Dim tmp As String
627|   tmp = UCase$(Me!ACTION.Caption)
628|   Select Case tmp
629|    Case "Decode MapSQL"
630|       tmp = Me!txtMapSQL
631|       Call DecodeMapSQL(tmp)
632|    Case "wait" 'do nothing
633|    Case "refresh"
634|       Call Form_Load
635|    Case Else
636|   End Select
637| End Sub
638|
639| Sub cmb_zqFeatureID_Change ()
640|  SetMapSQL
641|  Call DBFormSQLAddField("")
642| End Sub
643|
644| Sub cmb_zqFeatureID_Click ()
645|   SetMapSQL
646|   Call DBFormSQLAddField("")
647| End Sub
648|
```

```
649  Sub cmb_zqFeatureID_DragDrop (Source As Control, X As Single, Y As Single)
650    cmb_zqFeatureID.AddItem "[" & Source & "]"
651    cmb_zqFeatureID.ListIndex = cmb_zqFeatureID.ListCount - 1
652  'Drops the selected text over the entire field, for use on
653  ' NAME only, because we'll need this for matching and it
654  ' NAME should never be a calculated field, at leat not easily.
655
656  End Sub
657
658  Sub cmb_zqFillColor_Change ()
659   SetMapSQL
660  End Sub
661
662  Sub cmb_zqFillColor_Click ()
663   SetMapSQL
664  End Sub
665
666  Sub cmb_zqFillColor_DragDrop (Source As Control, X As Single, Y As Single)
667   cmb_zqFillColor.AddItem Mid$(cmb_zqFillColor, 1, cmb_zqFillColor.SelStart) & " " & "["
       & Source & "]" & " " & Mid$(cmb_zqFillColor, cmb_zqFillColor.SelStart +
       cmb_zqFillColor.SelLength + 1, 32768)
668   cmb_zqFillColor.ListIndex = cmb_zqFillColor.ListCount - 1
669  End Sub
670
671  Sub cmb_zqFillStyle_Change ()
672   SetMapSQL
673  End Sub
674
675  Sub cmb_zqFillStyle_Click ()
676   SetMapSQL
677  End Sub
678
679  Sub cmb_zqFillStyle_DragDrop (Source As Control, X As Single, Y As Single)
680   cmb_zqFillStyle.AddItem Mid$(cmb_zqFillStyle, 1, cmb_zqFillStyle.SelStart) & " " & "["
       & Source & "]" & " " & Mid$(cmb_zqFillStyle, cmb_zqFillStyle.SelStart +
       cmb_zqFillStyle.SelLength + 1, 32768)
681   cmb_zqFillStyle.ListIndex = cmb_zqFillStyle.ListCount - 1
682  End Sub
683
684  Sub cmb_zqMaxX_Change ()
685   SetMapSQL
686  End Sub
687
688  Sub cmb_zqMaxX_Click ()
689   SetMapSQL
690  End Sub
691
692  Sub cmb_zqMaxX_DragDrop (Source As Control, X As Single, Y As Single)
693    AddAndHighlightItem "[" & Source & "]", cmb_zqMaxX
694  End Sub
695
696  Sub cmb_zqMaxY_Change ()
697   SetMapSQL
698  End Sub
```

```
699|
700| Sub cmb_zqMaxY_Click ()
701|  SetMapSQL
702| End Sub
703|
704| Sub cmb_zqMaxY_DragDrop (Source As Control, X As Single, Y As Single)
705|   AddAndHighlightItem "[" & Source & "]", cmb_zqMaxY 'Sets all of the following Z-field to
      whatever was dropped on it
706| End Sub
707|
708| Sub cmb_zqMinX_Change ()
709|  SetMapSQL
710| End Sub
711|
712| Sub cmb_zqMinX_Click ()
713|  SetMapSQL
714| End Sub
715|
716| Sub cmb_zqMinX_DragDrop (Source As Control, X As Single, Y As Single)
717|   AddAndHighlightItem "[" & Source & "]", cmb_zqMinX
718| End Sub
719|
720| Sub cmb_zqMinY_Change ()
721|  SetMapSQL
722| End Sub
723|
724| Sub cmb_zqMinY_Click ()
725|  SetMapSQL
726| End Sub
727|
728| Sub cmb_zqMinY_DragDrop (Source As Control, X As Single, Y As Single)
729|  AddAndHighlightItem "[" & Source & "]", cmb_zqMinY
730| End Sub
731|
732| Sub cmb_zqOutlineColor_Change ()
733|  SetMapSQL
734| End Sub
735|
736| Sub cmb_zqOutlineColor_Click ()
737|  SetMapSQL
738| End Sub
739|
740| Sub cmb_zqOutlineColor_DragDrop (Source As Control, X As Single, Y As Single)
741|  cmb_zqOutlineColor.AddItem Mid$(cmb_zqOutlineColor, 1,
      cmb_zqOutlineColor.SelStart) & " " & "[" & Source & "]" & " " &
      Mid$(cmb_zqOutlineColor, cmb_zqOutlineColor.SelStart +
      cmb_zqOutlineColor.SelLength + 1, 32768)
742|  cmb_zqOutlineColor.ListIndex = cmb_zqOutlineColor.ListCount - 1
743| End Sub
744|
745| Sub cmb_zqOutlineStyle_Change ()
746|  SetMapSQL
747| End Sub
748|
```

```
Sub cmb_zqOutlineStyle_Click ()
 SetMapSQL
End Sub

Sub cmb_zqOutlineStyle_DragDrop (Source As Control, X As Single, Y As Single)
 cmb_zqOutlineStyle = Mid$(cmb_zqOutlineStyle, 1, cmb_zqOutlineStyle.SelStart) & " "
& "[" & Source & "]" & " " & Mid$(cmb_zqOutlineStyle, cmb_zqOutlineStyle.SelStart +
cmb_zqOutlineStyle.SelLength + 1, 32768)
End Sub

Sub cmb_zqOutlineWidth_Change ()
 SetMapSQL
End Sub

Sub cmb_zqOutlineWidth_Click ()
 SetMapSQL
End Sub

Sub cmb_zqOutlineWidth_DragDrop (Source As Control, X As Single, Y As Single)
 cmb_zqOutlineWidth.AddItem Mid$(cmb_zqOutlineWidth, 1,
cmb_zqOutlineWidth.SelStart) & " " & "[" & Source & "]" & " " &
Mid$(cmb_zqOutlineWidth, cmb_zqOutlineWidth.SelStart +
cmb_zqOutlineWidth.SelLength + 1, 32768)
 cmb_zqOutlineWidth.ListIndex = cmb_zqOutlineWidth.ListCount - 1
End Sub

Sub cmbFieldList_KeyDown (KeyCode As Integer, Shift As Integer)
  If KeyCode = KEY_INSERT And Shift = 2 Then 'CTRL-Insert?
    clipboard.Clear
    clipboard.SetText ("[" & cmbFieldList & "]")
  End If
End Sub

Sub cmbTblQryList_Click ()
On Error GoTo ERRcmbTblQryList_Click

If Me!ACTION.Caption = "wait" Then Exit Sub
If Not flgIgnoreMapSQLChange Then
  'For usual case, set the SpatialQuerySQL and the theme name
  txtSpatialQuerySQL = "SELECT * FROM " &
AddBracketsAsNeeded(CStr(cmbTblQryList))
  ListOfFieldsForDBForm = "*"
  txtThemeName = "Tiles: " & cmbTblQryList & "  " & Now
End If
Dim TableName As String
 TableName = cmbTblQryList
 Set snpFieldList = MapDB.CreateSnapshot(TableName)
 nFields = snpFieldList.Fields.Count

 cmbFieldList.Clear
 'ReDim Preserve strFieldsList(1 To nFields)
 ReDim FieldsInCurrentQT(1 To nFields), FieldListWBrackets(1 To nFields)

 For i = 1 To nFields
```

```
797|     cmbFieldList.AddItem snpFieldList.Fields(i - 1).Name
798|     FieldsInCurrentQT(i) = snpFieldList.Fields(i - 1).Name
799|     FieldListWBrackets(i) = "[" & snpFieldList.Fields(i - 1).Name & "]"
800|   Next i
801|
802|   snpFieldList.Close
803|   cmbFieldList.ListIndex = 0
804|   txtCriteria = ""
805|
806|   cmb_zqFeatureID.Clear
807|   FillCMBFromSGSTLST FieldListWBrackets(), FeatureNameLookFor,
      FeatureNameShowAlways, cmb_zqFeatureID
808|   cmb_zqFeatureID.ListIndex = 0
809|
810|   cmb_zqMinX.Clear
811|   FillCMBFromSGSTLST FieldListWBrackets(), "WEST MINX " & XLookFor,
      XShowAlways, cmb_zqMinX
812|   cmb_zqMinX.ListIndex = 0
813|
814|   cmb_zqMaxX.Clear
815|   FillCMBFromSGSTLST FieldListWBrackets(), "EAST MAXX " & XLookFor,
      XShowAlways, cmb_zqMaxX
816|   cmb_zqMaxX.ListIndex = 0
817|
818|   cmb_zqMinY.Clear
819|   FillCMBFromSGSTLST FieldListWBrackets(), "SOUTH MINY " & YLookFor,
      YShowAlways, cmb_zqMinY
820|   cmb_zqMinY.ListIndex = 0
821|
822|   cmb_zqMaxY.Clear
823|   FillCMBFromSGSTLST FieldListWBrackets(), "NORTH MAXY " & YLookFor,
      YShowAlways, cmb_zqMaxY
824|   cmb_zqMaxY.ListIndex = 0
825|
826|   cmb_zqOutlineStyle.Clear
827|   FillCMBFromSGSTLST FieldListWBrackets(), LineStyleLookFor, LineStyleShowAlways,
      cmb_zqOutlineStyle
828|   cmb_zqOutlineStyle.ListIndex = 0
829|
830|   cmb_zqFillColor.Clear
831|   FillCMBFromSGSTLST FieldListWBrackets(), ColorLookFor, ColorShowAlways,
      cmb_zqFillColor
832|   cmb_zqFillColor.ListIndex = 0
833|
834|   cmb_zqOutlineColor.Clear
835|   FillCMBFromSGSTLST FieldListWBrackets(), ColorLookFor, ColorShowAlways,
      cmb_zqOutlineColor
836|   cmb_zqOutlineColor.ListIndex = 0
837|
838|   cmb_zqOutlineWidth.Clear
839|   FillCMBFromSGSTLST FieldListWBrackets(), LineWidthLookFor, "0<=>No_Outline " &
      LineWidthShowAlways, cmb_zqOutlineWidth
840|   cmb_zqOutlineWidth.ListIndex = 0
841|
```

```
842|  cmb_zqFillStyle.Clear
843|  FillCMBFromSGSTLST FieldListWBrackets(), FillStyleLookFor, FillStyleShowAlways,
      cmb_zqFillStyle
844|  cmb_zqFillStyle.ListIndex = 0
845|
846|  flgActivateDragDrop = 0
847|
848| Exit Sub
849| ERRcmbTblQryList_Click:
850|  MsgBox "Error #" & Err & " occurred in cmbTblQryList_Click." & CRLF & Error
851|  Err = 0
852|  Exit Sub
853|  Resume Next
854|
855| End Sub
856|
857| Sub cmdCancel_Click ()
858| Me.Hide
859| End Sub
860|
861| Sub cmdOK_Click ()
862| If cmbTblQryList = "" Then
863|    MsgBox "A query or table must be selected before trying to display its results." & CRLF
      & "Or you can hit CANCEL to not add a point theme right now and get out of here."
864|    cmbTblQryList.SetFocus
865|    SendKeys "%{DOWN}"
866|    Exit Sub
867| End If
868|
869|
870|   'SetMapSQL
871|   'Call DBFormSQLAddField("")
872|   Me!cmdOK.SetFocus
873| Me.Hide
874|   Me!cmdOK.FontItalic = True
875|
876| End Sub
877|
878| Sub DBFormSQLAddField (NewField As String)
879| If flgIgnoreMapSQLChange Then Exit Sub
880| If NewField <> "" Then
881|  If ListOfFieldsForDBForm = "*" Then
882|    ListOfFieldsForDBForm = "[" & NewField & "]"
883|   Else
884|    ListOfFieldsForDBForm = ListOfFieldsForDBForm & ", [" & NewField & "]"
885|  End If
886| End If
887| txtSpatialQuerySQL = "SELECT " & ListOfFieldsForDBForm & ", " & cmb_zqFeatureID &
      " AS zqFeatureID FROM " & "[" & cmbTblQryList & "]"
888| End Sub
889|
890| Sub DecodeMapSQL (strSQLIn As String)
891|  Dim i As Long, h As Integer, strTemp As String
892|  strSQLIn = Trim$(strSQLIn)
```

```
893|   If strSQLIn = "" Then Exit Sub
894|
895|
896|
897|  'If the MapSQL was changed by another form or by the user,
898|  '   set the comboboxes to reflect the changes as best possible.
899|   Dim strParameters  As String, strFields As String, strFrom As String, strWhere As
      String, strGroupBy As String, strHaving As String, strOrderBy As String
900|
901|
902|
903|   If SplitSQLStatement(strSQLIn, strParameters, strFields, strFrom, strWhere,
      strGroupBy, strHaving, strOrderBy) = False Then Exit Sub
904|
905|   If strParameters <> "" Then
906|     MsgBox "WHEAT programs do not run parameter queries.  Please choose another
      map SQL string."
907|     Exit Sub
908|   End If
909|   flgIgnoreMapSQLChange = True
910|   Dim strProblems As String
911|
912|   strFrom = Strip2Brackets(strFrom)
913|  'Set the source table or query
914|   i = IndexOfStringInCOMBOBOX(strFrom, Me!cmbTblQryList)
915|   Me!cmbTblQryList.ListIndex = i
916|    DoEvents 'Let the click event occur for the QT List
917|
918|  'Get field assignments
919|   Dim strRawName() As String, strAsName() As String, nFields As Long, nF As Long
920|   Call PullFieldNamesFromString(strFields, strRawName(), strAsName(), nFields)
921|   'For i = 1 To nFields
922|   '    dDebug.Print strRawName(i), , strAsName(i)
923|   'Next i
924|
925|  '   'Set boundbox Coordinate field
926|
927|    'Set MinX Coordinate field
928|   For nF = 1 To nFields
929|    If strAsName(nF) = "zqMinX" Then
930|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqMinX)
931|      If i <> -1 Then
932|        Me.cmb_zqMinX.ListIndex = i
933|       Else
934|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
      for MinX coordinate, but this routine cannot find it.  You will have to select this one
      yourself." & CRLF
935|      End If
936|      Exit For
937|    End If
938|   Next nF
939|
940|    'Set MinY Coordinate field
941|   For nF = 1 To nFields
```

```
942|    If strAsName(nF) = "zqMinY" Then
943|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqMinY)
944|      If i <> -1 Then
945|        Me.cmb_zqMinY.ListIndex = i
946|       Else
947|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
       for MinY coordinate, but this routine cannot find it.  You will have to select this one
       yourself." & CRLF
948|      End If
949|      Exit For
950|    End If
951|   Next nF
952|
953|    'Set MaxX Coordinate field
954|   For nF = 1 To nFields
955|    If strAsName(nF) = "zqMaxX" Then
956|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqMaxX)
957|      If i <> -1 Then
958|        Me.cmb_zqMaxX.ListIndex = i
959|       Else
960|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
       for MaxX coordinate, but this routine cannot find it.  You will have to select this one
       yourself." & CRLF
961|      End If
962|      Exit For
963|    End If
964|   Next nF
965|
966|    'Set MaxX Coordinate field
967|   For nF = 1 To nFields
968|    If strAsName(nF) = "zqMaxY" Then
969|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqMaxY)
970|      If i <> -1 Then
971|        Me.cmb_zqMaxY.ListIndex = i
972|       Else
973|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
       for MaxY coordinate, but this routine cannot find it.  You will have to select this one
       yourself." & CRLF
974|      End If
975|      Exit For
976|    End If
977|   Next nF
978|
979|
980|
981|
982|    'Set FeatureID field
983|   For nF = 1 To nFields
984|    If strAsName(nF) = "zqFeatureID" Then
985|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFeatureID)
986|      If i <> -1 Then
987|        Me.cmb_zqFeatureID.ListIndex = i
988|       Else
989|        strProblems = strProblems & "This theme originally used " & strRawName(nF) & "
```

```
for feature ID, but this routine cannot find it.  You will have to select this one yourself." &
CRLF
      End If
      Exit For
    End If
  Next nF

    'The fill style
  For nF = 1 To nFields
   If strAsName(nF) = "zqFillStyle" Then
     i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFillStyle)
     If i <> -1 Then
       Me.cmb_zqFillStyle.ListIndex = i
      Else
       i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFillStyle)
       If i <> -1 Then
         Me.cmb_zqFillStyle.ListIndex = i
        Else

          strProblems = strProblems & "This theme originally used " & strRawName(nF) &
" for the line style, but this routine cannot find it.  You will have to select this one
yourself." & CRLF
       End If
     End If
     Exit For
   End If
  Next nF


   'Set outline style field
  For nF = 1 To nFields
   If strAsName(nF) = "zqOutlineStyle" Then
     i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqOutlineStyle)
     If i <> -1 Then
       Me.cmb_zqOutlineStyle.ListIndex = i
      Else
       i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqOutlineStyle)
       If i <> -1 Then
         Me.cmb_zqOutlineStyle.ListIndex = i
        Else
          strProblems = strProblems & "This theme originally used " & strRawName(nF) &
" for the outline style, but this routine cannot find it.  You will have to select this one
yourself." & CRLF
       End If
     End If
     Exit For
   End If
  Next nF



    'outline color
  For nF = 1 To nFields
   If strAsName(nF) = "zqOutlineColor" Then
```

```
1038|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqOutlineColor)
1039|      If i <> -1 Then
1040|        Me.cmb_zqOutlineColor.ListIndex = i
1041|       Else
1042|        i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqOutlineColor)
1043|        If i <> -1 Then
1044|          Me.cmb_zqOutlineColor.ListIndex = i
1045|         Else
1046|          strProblems = strProblems & "This theme originally used " & strRawName(nF) &
      " for the color, but this routine cannot find it.  You will have to select this one yourself." &
      CRLF
1047|        End If
1048|      End If
1049|      Exit For
1050|    End If
1051|   Next nF
1052|
1053|    'fill color
1054|   For nF = 1 To nFields
1055|    If strAsName(nF) = "zqFillColor" Then
1056|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFillColor)
1057|      If i <> -1 Then
1058|        Me.cmb_zqFillColor.ListIndex = i
1059|       Else
1060|        i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFillColor)
1061|        If i <> -1 Then
1062|          Me.cmb_zqFillColor.ListIndex = i
1063|         Else
1064|          strProblems = strProblems & "This theme originally used " & strRawName(nF) &
      " for the color, but this routine cannot find it.  You will have to select this one yourself." &
      CRLF
1065|        End If
1066|      End If
1067|      Exit For
1068|    End If
1069|   Next nF
1070|
1071|
1072|
1073|
1074|      'The fill style
1075|   For nF = 1 To nFields
1076|    If strAsName(nF) = "zqFillStyle" Then
1077|      i = IndexOfStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFillStyle)
1078|      If i <> -1 Then
1079|        Me.cmb_zqFillStyle.ListIndex = i
1080|       Else
1081|        i = IndexOfSubStringInCOMBOBOX(strRawName(nF), Me!cmb_zqFillStyle)
1082|        If i <> -1 Then
1083|          Me.cmb_zqFillStyle.ListIndex = i
1084|         Else
1085|
1086|          strProblems = strProblems & "This theme originally used " & strRawName(nF) &
      " for the line style, but this routine cannot find it.  You will have to select this one
```

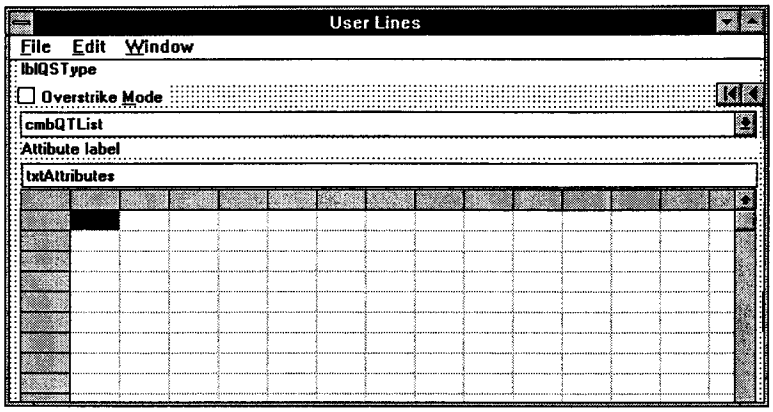
```
        yourself." & CRLF
1087|         End If
1088|       End If
1089|       Exit For
1090|     End If
1091|   Next nF
1092|
1093|    txtCriteria = strWhere
1094|
1095|
1096|   flgIgnoreMapSQLChange = False
1097|    If strProblems <> "" Then
1098|      Beep
1099|      MsgBox strProblems
1100|    End If
1101|
1102|  End Sub
1103|
1104|  Sub Form_Load ()
1105|  Me!cmbTblQryList.Clear
1106|  For i = 1 To nTblQrys
1107|   Me!cmbTblQryList.AddItem strTblQryList(i)
1108|  Next i
1109|  flgActivateDragDrop = False
1110|  End Sub
1111|
1112|  Sub Form_Paint ()
1113|  If cmbTblQryList.ListIndex = -1 Then cmbTblQryList.ListIndex = 0
1114|  cmdOK.FontItalic = False
1115|  End Sub
1116|
1117|  Sub imgFieldName_Click ()
1118|   cmbFieldList.Drag
1119|  End Sub
1120|
1121|  Sub lblEast_DragDrop (Source As Control, X As Single, Y As Single)
1122|   AddAndHighlightItem "[" & Source & "]", cmb_zqMaxX
1123|  End Sub
1124|
1125|  Sub lblNorth_DragDrop (Source As Control, X As Single, Y As Single)
1126|    AddAndHighlightItem "[" & Source & "]", cmb_zqMaxY
1127|  End Sub
1128|
1129|  Sub lblSouth_DragDrop (Source As Control, X As Single, Y As Single)
1130|  AddAndHighlightItem "[" & Source & "]", cmb_zqMinY
1131|  End Sub
1132|
1133|  Sub lblWest_DragDrop (Source As Control, X As Single, Y As Single)
1134|   AddAndHighlightItem "[" & Source & "]", cmb_zqMinX
1135|  End Sub
1136|
1137|  Sub picSample_Paint ()
1138|   ShowSample
1139|  End Sub
```

```
1140|
1141| Sub SetMapSQL ()
1142| If flgIgnoreMapSQLChange Then Exit Sub
1143|
1144|  ShowSample
1145| Dim TempSQL As String
1146| TempSQL = "SELECT DISTINCTROW "
1147| ' Assemble a new MapSQL statement.
1148|
1149| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqMinX.Text), "zqMinX") & " , "
1150| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqMaxX.Text), "zqMaxX") & " , "
1151| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqMinY.Text), "zqMinY") & " , "
1152| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqMaxY.Text), "zqMaxY") & " , "
1153|
1154| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqOutlineStyle.Text),
      "zqOutlineStyle") & " , "
1155| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqOutlineColor.Text),
      "zqOutlineColor") & " , "
1156| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqOutlineWidth.Text),
      "zqOutlineWidth") & " , "
1157|
1158| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqFillColor.Text), "zqFillColor") & " , "
1159| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqFillStyle.Text), "zqFillStyle") & " , "
1160|
1161| TempSQL = TempSQL &
      AddVarNameAsNeeded(GetGoodValueField(cmb_zqFeatureID.Text), "zqFeatureID") &
      "  "
1162|
1163| TempSQL = TempSQL & " FROM [" & cmbTblQryList & "]  "
1164|
1165|
1166| If (txtCriteria = "") Then
1167|    TempSQL = TempSQL
1168|   Else
1169|    TempSQL = TempSQL & "  WHERE (" & txtCriteria & ") "
1170| End If
1171|
1172|
1173|
1174|
1175| 'Build the Order By clause
1176| Dim strOrderBy As String, Temp As String    'changed 950118
1177|   strOrderBy = ""
1178|
1179|    Temp = GetFieldForOrderBy(cmb_zqOutlineWidth.Text)
```

```
1180|    If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
1181|
1182|    Temp = GetFieldForOrderBy(cmb_zqOutlineStyle.Text)
1183|    If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
1184|
1185|    Temp = GetFieldForOrderBy(cmb_zqFillStyle.Text)
1186|    If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
1187|
1188|
1189|    Temp = GetFieldForOrderBy(cmb_zqOutlineColor.Text)
1190|    If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
1191|
1192|    Temp = GetFieldForOrderBy(cmb_zqFillColor.Text)
1193|    If Temp <> "" Then strOrderBy = strOrderBy & Temp & " , "
1194|
1195|    If strOrderBy <> "" Then
1196|      strOrderBy = " ORDER BY " & Left$(strOrderBy, Len(strOrderBy) - 2)
1197|    End If
1198|
1199|    TempSQL = TempSQL & strOrderBy & " ;"
1200|
1201|  ' Now put the SQL statement into the appropriate text box.
1202|  flgIgnoreMapSQLChange = True
1203|     txtMapSQL = TempSQL
1204|  flgIgnoreMapSQLChange = False
1205|
1206|  End Sub
1207|
1208|  Sub ShowSample ()
1209|    On Error Resume Next
1210|    picSample.Cls
1211|    picSample.DrawStyle = Val(cmb_zqOutlineStyle.Text)
1212|    picSample.DrawWidth = Val(cmb_zqOutlineWidth)
1213|    picSample.ForeColor = Val(cmb_zqOutlineColor)
1214|    picSample.FillStyle = Val(cmb_zqFillStyle)
1215|    picSample.FillColor = Val(cmb_zqFillColor)
1216|    Dim TempColor As Long, SampleHDC As Integer
1217|    SampleHDC = Me.picSample.hDC
1218|    TempColor = Rectangle(SampleHDC, 6, 6, 35, 45)
1219|  End Sub
1220|
1221|  Sub txtCriteria_Change ()
1222|   SetMapSQL
1223|  End Sub
1224|
1225|  Sub txtCriteria_DragDrop (Source As Control, X As Single, Y As Single)
1226|  txtCriteria = Mid$(txtCriteria, 1, txtCriteria.SelStart) & " [" & CStr(Source) & "] " &
       Mid$(txtCriteria, txtCriteria.SelStart + txtCriteria.SelLength + 1, 32768)
1227|  'Drops the selected text at the current insertion point
1228|  End Sub
1229|
1230|  Sub txtMapSQL_KeyPress (KeyAscii As Integer)
1231|
1232|    If flgIgnoreMapSQLChange Then Exit Sub
```

```
1233|    If KeyAscii <> 13 Then Exit Sub
1234|    KeyAscii = 0'Swallow the return key
1235|     Dim strSQL As String
1236|     strSQL = txtMapSQL
1237|     Call DecodeMapSQL(strSQL)
1238|
1239| End Sub
1240|
1241| Sub txtSpatialQuerySQL_DragDrop (Source As Control, X As Single, Y As Single)
1242| Call DBFormSQLAddField(CStr(Source))
1243| End Sub
1244|
1245| Sub txtSpatialQuerySQL_KeyPress (KeyAscii As Integer)
1246| If KeyAscii = 13 Then
1247|    KeyAscii = 0
1248|    'Other decoding of the SQL statement should go here
1249| End If
1250|
1251| End Sub
1252|
1253| Sub zlblFillColor_Click ()
1254|   Dim ColorWas As Long, NewColor As Long
1255|   ColorWas = Val(cmb_zqFillColor)
1256|   NewColor = GetColorWithCmnDialog(ColorWas)
1257|   If NewColor <> ColorWas Then AddAndHighlightItem Format$(NewColor),
      cmb_zqFillColor
1258|
1259| End Sub
1260|
1261| Sub zlblFillColor_DragDrop (Source As Control, X As Single, Y As Single)
1262|  cmb_zqFillColor.AddItem "[" & Source & "]"
1263|  cmb_zqFillColor.ListIndex = cmb_zqFillColor.ListCount - 1
1264| End Sub
1265|
1266| Sub zlblFillStyle_DragDrop (Source As Control, X As Single, Y As Single)
1267|  cmb_zqFillStyle.AddItem "[" & Source & "]"
1268|  cmb_zqFillStyle.ListIndex = cmb_zqFillStyle.ListCount - 1
1269| End Sub
1270|
1271| Sub zlblOutlineColor_Click ()
1272|   Dim ColorWas As Long, NewColor As Long
1273|   ColorWas = Val(cmb_zqOutlineColor)
1274|   NewColor = GetColorWithCmnDialog(ColorWas)
1275|   If NewColor <> ColorWas Then AddAndHighlightItem Format$(NewColor),
      cmb_zqOutlineColor
1276|
1277| End Sub
1278|
1279| Sub zlblOutlineColor_DragDrop (Source As Control, X As Single, Y As Single)
1280|  cmb_zqOutlineColor.AddItem "[" & Source & "]"
1281|  cmb_zqOutlineColor.ListIndex = cmb_zqOutlineColor.ListCount - 1
1282| End Sub
1283|
1284| Sub zzzlblFields_Click ()
```

```
1285|  cmbFieldList.Drag
1286|  End Sub
1287|
1288|  Sub zzzlblName_DragDrop (Source As Control, X As Single, Y As Single)
1289|    cmb_zqFeatureID.AddItem "[" & Source & "]"'Sets all of the following Z-field to
       whatever was dropped on it
1290|    cmb_zqFeatureID.ListIndex = cmb_zqFeatureID.ListCount - 1
1291|  End Sub
1292|
1293|  Sub zzzlblZSize_DragDrop (Source As Control, X As Single, Y As Single)
1294|   cmb_zqOutlineWidth.AddItem "[" & Source & "]"'Sets all of the following Z-field to
       whatever was dropped on it
1295|   cmb_zqOutlineWidth.ListIndex = cmb_zqOutlineWidth.ListCount - 1
1296|  End Sub
1297|
1298|  Sub zzzlblZSymbol_DragDrop (Source As Control, X As Single, Y As Single)
1299|   cmb_zqOutlineStyle.AddItem "[" & Source & "]" 'Sets all of the following Z-field to
       whatever was dropped on it
1300|   cmb_zqOutlineStyle.ListIndex = cmb_zqOutlineStyle.ListCount - 1
1301|  End Sub
1302|
1303|  Sub zzzzlblCriteia_DragDrop (Source As Control, X As Single, Y As Single)
1304|  txtCriteria = "[" & Source & "]"
1305|  End Sub
1306|
1307|  Sub zzzzlblDBFormSQL_DragDrop (Source As Control, X As Single, Y As Single)
1308|  Call DBFormSQLAddField(CStr(Source))
1309|  End Sub
1310|
```


User Lines
File Edit Window
lblQSType
Overstrike Mode
cmbQTList
Attibute label
txtAttributes

```
VERSION 2.00
Begin Form frmUserLines
  Caption        = "User Lines"
  ClientHeight   = 4005
  ClientLeft     = 345
  ClientTop      = 990
  ClientWidth    = 8745
  Height         = 4695
  Icon           = USERLINS.FRX:0000
  Left           = 285
  LinkTopic      = "Form1"
  MDIChild       = -1 'True
  ScaleHeight    = 70.644
  ScaleMode      = 6 'Millimeter
  ScaleWidth     = 154.252
  Top            = 360
  Width          = 8865
  Begin ComboBox cmbQTList
    Height       = 300
    Left         = 120
    Sorted       = -1 'True
    Style        = 2 'Dropdown List
    TabIndex     = 6
    Top          = 600
    Visible      = 0 'False
    Width        = 8535
  End
  Begin CommonDialog CMDialog2
    Left         = 8730
    Top          = 450
  End
  Begin CommonDialog CMDialog1
    Left         = 8730
    Top          = 2700
  End
  Begin Data Data1
    Caption      = "Move record "
    Connect      = ""
    DatabaseName = ""
    Exclusive    = 0 'False
    Height       = 270
    Left         = 8190
    Options      = 0
    ReadOnly     = 0 'False
    RecordSource = ""
    Top          = 270
    Visible      = 0 'False
    Width        = 2445
  End
  Begin CheckBox chkOverstrike
    Caption      = "Overstrike &Mode"
    Height       = 255
    Left         = 90
    TabIndex     = 3
```

```
    Top          = 285
    Width        = 1757
  End
  Begin TextBox txtAttributes
    Height       = 285
    Left         = 135
    TabIndex     = 1
    Text         = "txtAttributes"
    Top          = 1200
    Width        = 8535
  End
  Begin Grid grdAttributes
    Cols         = 15
    Height       = 4515
    Left         = 120
    Rows         = 30
    TabIndex     = 0
    Top          = 1500
    Width        = 8535
  End
  Begin Label lblQSSource
    Caption      = "lblQSSource"
    Height       = 255
    Left         = 120
    TabIndex     = 5
    Top          = 600
    Width        = 8415
  End
  Begin Label lblQSType
    Caption      = "lblQSType"
    Height       = 255
    Left         = 120
    TabIndex     = 4
    Top          = 0
    Width        = 8535
  End
  Begin Label lblAttributes
    Caption      = "Attibute label"
    Height       = 210
    Left         = 135
    TabIndex     = 2
    Top          = 930
    Width        = 8535
  End
  Begin Menu zzzzFile
    Caption      = "&File"
    Begin Menu mfOpenDB
      Caption    = "&Open Database..."
      Visible    = 0  'False
    End
    Begin Menu mfCloseDB
      Caption    = "&Close Database"
      Visible    = 0  'False
    End
```

```
109|     Begin Menu zz10
110|       Caption      = "-"
111|     End
112|     Begin Menu mfNewTable
113|       Caption      = "New &Table..."
114|     End
115|     Begin Menu zzz1
116|       Caption      = "-"
117|     End
118|     Begin Menu mfNewFile
119|       Caption      = "New &File..."
120|     End
121|     Begin Menu zz3
122|       Caption      = "-"
123|       Visible      = 0  'False
124|     End
125|     Begin Menu mfSaveTableChanges
126|       Caption      = "&Save Changes"
127|       Visible      = 0  'False
128|     End
129|     Begin Menu mfAbandonTableChanges
130|       Caption      = "&Abandon Changes"
131|       Visible      = 0  'False
132|     End
133|     Begin Menu zzzzz1
134|       Caption      = "-"
135|     End
136|     Begin Menu mfExit
137|       Caption      = "E&xit"
138|     End
139|   End
140|   Begin Menu zzEdit
141|     Caption      = "&Edit"
142|     Begin Menu meUndoThisRow
143|       Caption      = "&Undo changes to this row"
144|       Shortcut     = %{BKSP}
145|     End
146|     Begin Menu zz6
147|       Caption      = "-"
148|     End
149|     Begin Menu meCopy
150|       Caption      = "&Copy"
151|       Shortcut     = ^{INSERT}
152|     End
153|     Begin Menu mePaste
154|       Caption      = "&Paste"
155|       Shortcut     = +{INSERT}
156|     End
157|     Begin Menu meCut
158|       Caption      = "Cu&t"
159|       Shortcut     = +{DEL}
160|     End
161|     Begin Menu meClearGrid
162|       Caption      = "C&lear"
```

```
163|       Shortcut      = ^X
164|     End
165|     Begin Menu zzzz1
166|       Caption       = "-"
167|     End
168|     Begin Menu meInsertRow
169|       Caption       = "&Append Records"
170|       Shortcut      = ^A
171|     End
172|     Begin Menu meDeleteRow
173|       Caption       = "&Delete Records"
174|       Shortcut      = ^D
175|     End
176|     Begin Menu zzz5
177|       Caption       = "-"
178|     End
179|     Begin Menu meAddColumn
180|       Caption       = "Add Co&lumn"
181|     End
182|     Begin Menu meColDefinition
183|       Caption       = "Column De&finition"
184|     End
185|   End
186|   Begin Menu mnuWindow
187|     Caption       = "&Window"
188|     WindowList    = -1 'True
189|   End
190| End
191| Option Explicit
192|
193| Dim flgChangeRow As Integer, flgIgnoreMovement As Integer
194| Dim flgChangesWereMade As Integer, flgBookmarkable As Integer
195|
196| Dim strBookMark() As String
197| Dim flgSkipMe() As Long
198| Dim nFields As Long, n As Long, nRecs As Long, nR As Long
199| Dim nF As Long, FieldType() As Long
200| Dim lngCurrentRow As Long, lngCurrentCol As Long
201|
202| Dim DB As Database, tblLineData As Table
203| 'Dim snpQTList As Snapshot, dnsCurrentQT As dynaset
204|
205| Dim flgAddingRecord As Integer   'Might want to eliminate this variable.
206| Dim TabChar As String * 1, CR  As String * 1, CRLF  As String * 2
207| Dim ListOfFieldTypes As String
208|
209| Sub cmbQTList_Click ()
210| Dim Temp As String
211| mousepointer = 11
212|
213| Temp = cmbQTList
214| Call SetQuery(Temp)
215| flgIgnoreMovement = True
216| grdAttributes.FixedCols = 0: grdAttributes.Cols = 1
```

```
grdAttributes.Cols = 2: grdAttributes.FixedCols = 1
grdAttributes.Rows = 2
'Function GetClipBoardTextFromDynaset (dynIn As Dynaset, flgOptions As Long) As
String
Set dynViewInGrid = DB.CreateDynaset(Temp)
Call InitializeGridForDynaset(Me!grdAttributes, dynViewInGrid)

Temp = GetClipBoardTextFromDynaset(dynViewInGrid, 0)
Call SetGridFocusColumn(Me!grdAttributes, 1)
Call FillGridFromClipBoardText(Temp, Me!grdAttributes, 12&)

grdAttributes.RemoveItem 1


'Call OverwriteGridFromDynaset(Me!grdAttributes, dynViewInGrid)
flgIgnoreMovement = False
mousepointer = 0

End Sub

Sub cmdClearGrid_Click ()
 flgIgnoreMovement = True
  grdAttributes.FixedRows = 0
  grdAttributes.Rows = 1
  grdAttributes.Rows = 2
  grdAttributes.FixedRows = 1
 flgIgnoreMovement = False
End Sub

Sub Data1_Reposition ()
'On Error Resume Next
'DisplayCurrentRecord
End Sub

Sub DefineColumn ()
    flgIgnoreMovement = True
    Dim Temp As String, tmpFieldName As String, tmpFieldType As Integer,
strFieldTypeList As String
    grdAttributes.Row = 0
    n = grdAttributes.Col
    Temp = grdAttributes
    If Temp = "" Then
   tmpFieldName = "Field" & Format$(n, "00")
   tmpFieldType = 10'Default to text
      Else
   tmpFieldType = Val(Right(Temp, 3))
   If tmpFieldType <> 0 Then
      tmpFieldName = Left(Temp, Len(Temp) - 3)
     Else
      tmpFieldName = Temp
      tmpFieldType = 10
   End If
     End If

```

```
269|    ReDim OZQuestion(1 To 4)  As typOZQuestion
270|    Dim Accepted As Integer
271|    'Define the name.
272|    OZQuestion(1).KindOfControl = CONTROL_LABEL
273|    OZQuestion(1).StringData = "This field is currently named: " & Temp & " ." & CRLF &
       "If you wish, you may enter a new name below." '& CRLF & "and droning on and on and
       on about absolutely nothing at all in the world of any possible interest to anyone."
274|    OZQuestion(2).KindOfControl = CONTROL_TEXTBOX
275|    OZQuestion(2).StringData = tmpFieldName
276|
277|    'Define the datafield type.
278|    OZQuestion(3).KindOfControl = CONTROL_LABEL
279|    OZQuestion(3).StringData = "It currently has a datatype of " & tmpFieldType
280|    OZQuestion(4).KindOfControl = CONTROL_COMBOBOX
281|    strFieldTypeList = ListOfFieldTypes
282|    OZQuestion(4).StringData = strFieldTypeList
283|    OZQuestion(4).Extra = tmpFieldType
284| 'Consult the Wizard.
285|    AskOZ OZQuestion(), Accepted
286|
287|    If Accepted Then
288|       tmpFieldType = OZQuestion(4).Extra
289|       tmpFieldName = OZQuestion(2).StringData
290|       grdAttributes = tmpFieldName & " " & Format$(tmpFieldType, "00")
291|    End If
292|    flgIgnoreMovement = False
293|
294| End Sub
295|
296| Sub DisplayCurrentRecord ()
297| On Error GoTo ERRDisplayCurrentRecord
298| flgIgnoreMovement = True
299| grdAttributes.Row = 1
300| Dim Temp As String
301| For n = 1 To nFields
302|    grdAttributes.Col = n
303|    grdAttributes = Format(Data1.Recordset.Fields(n - 1).Value)
304| Next n
305|
306|
307| flgIgnoreMovement = False
308| Exit Sub
309|
310| ERRDisplayCurrentRecord:
311| If Err = 30010 Then 'Invalid number of columns or rows
312|    Err = 0
313|    Exit Sub
314|    Resume Next
315| End If
316| MsgBox "Error number " & Err & " encountered while displaying a record." & CRLF &
       Error$
317| Err = 0
318| Exit Sub
319| Resume
```

```
320|
321|
322|
323| End Sub
324|
325| Sub Form_Load ()
326| flgIgnoreMovement = False
327| CR = Chr$(13)
328| TabChar = Chr$(9)
329| CRLF = Chr$(13) & Chr$(10)
330| ListOfFieldTypes = ""
331|
332| 'cmbTypeList.Clear
333| 'cmbTypeList.AddItem "Do not save this field", 0
334| 'cmbTypeList.AddItem "BOOLEAN", 1
335| 'cmbTypeList.AddItem "BYTE", 2
336| 'cmbTypeList.AddItem "Short Integer", 3
337| 'cmbTypeList.AddItem "Long Integer 4-bytes", 4
338| 'cmbTypeList.AddItem "CURRENCY", 5
339| 'cmbTypeList.AddItem "Single Precision REAL*4", 6
340| 'cmbTypeList.AddItem "Double Precision REAL*8", 7
341| 'cmbTypeList.AddItem "DATE", 8
342| 'cmbTypeList.AddItem "Do not Save this field", 9
343| 'cmbTypeList.AddItem "TEXT", 10
344| 'cmbTypeList.AddItem "LONGBINARY", 11
345| 'cmbTypeList.AddItem "MEMO (Long String)", 12
346|
347|
348| ListOfFieldTypes = "Do not save this field" & CRLF & "BOOLEAN" & CRLF & "BYTE" &
    CRLF
349| ListOfFieldTypes = ListOfFieldTypes & "Short Integer" & CRLF & "Long Integer 4-bytes"
    & CRLF
350| ListOfFieldTypes = ListOfFieldTypes & "CURRENCY" & CRLF & "Single Precision
    REAL*4" & CRLF & "Double Precision REAL*8" & CRLF
351| ListOfFieldTypes = ListOfFieldTypes & "DATE" & CRLF & "Do not Save this field" &
    CRLF & "TEXT" & CRLF & "LONGBINARY" & CRLF & "MEMO (Long String)" & CRLF
352|
353| Me!grdAttributes.Cols = 5
354| flgIgnoreMovement = True
355|    Me!grdAttributes.Row = 0
356|
357|    Me!grdAttributes.Col = 1
358|    Me!grdAttributes = "Name 10"
359|    Me!grdAttributes.ColWidth(1) = 1200
360|
361|    Me!grdAttributes.Col = 2
362|    Me!grdAttributes = "XYCoordinatesAsText 12"
363|    Me!grdAttributes.ColWidth(2) = 3600
364|
365|    Me!grdAttributes.Col = 3
366|    Me!grdAttributes = "TimeStamp 08"
367|    Me!grdAttributes.ColWidth(3) = 1800
368|
369|    Me!grdAttributes.Col = 4
```

```
370|     Me!grdAttributes = "SequenceNumber 08"
371|     Me!grdAttributes.ColWidth(4) = 1200
372|
373|     Me!grdAttributes.Rows = 2
374| flgIgnoreMovement = False
375| End Sub
376|
377| Sub Form_Paint ()
378| flgIgnoreMovement = False
379| End Sub
380|
381| Sub Form_QueryUnload (Cancel As Integer, UnloadMode As Integer)
382| 'Check whether changes need to be saved.
383|
384| If flgChangesWereMade Then
385|       'Present a yes/no/cancel box to the user.
386|   nF = MsgBox("Changes were made to this data set." & CRLF & "Do you wish to save
    them now?", 35)
387|   If nF = 2 Then
388|     Cancel = True
389|     Exit Sub'user chose cancel, so clear out.
390|     End If
391|   If nF = 6 Then 'User chose to save changes
392|     Call mfSaveTableChanges_Click
393|     If flgChangesWereMade Then Cancel = True 'Still need to cancel
394|   End If
395|   'flgChangesWereMade = False  'If the user chose yes or no, reset the flag.
396| End If
397|
398| End Sub
399|
400| Sub Form_Resize ()
401| If frmGridDataEdit.WindowState = 1 Then Exit Sub'iconized
402| On Error GoTo ERRResize
403|     lblAttributes.Width = frmGridDataEdit.ScaleWidth - 5
404|     txtAttributes.Width = frmGridDataEdit.ScaleWidth - 5
405|     grdAttributes.Width = frmGridDataEdit.ScaleWidth - 5
406|     grdAttributes.Height = frmGridDataEdit.ScaleHeight - 27.5
407|
408| cmbQTList.Width = Me.ScaleWidth - 10
409|
410| Exit Sub
411|
412| ERRResize:
413| If Err = 380 Then 'invalid width
414|   frmGridDataEdit.Width = frmGridDataEdit.Width * 1.2
415|   Resume
416| End If
417|
418|
419| End Sub
420|
421| Sub GetIgnoreMovement (FromValue As Long)
422|  FromValue = flgIgnoreMovement
```

```
End Sub

Sub grdAttributes_KeyDown (Keycode As Integer, Shift As Integer)
If Keycode > 48 Then
   txtAttributes.SetFocus
End If
End Sub

Sub grdAttributes_KeyPress (KeyAscii As Integer)
   If KeyAscii = 13 Then SendKeys "{DOWN}"
   If KeyAscii > 31 Then SendKeys Chr$(KeyAscii)
End Sub

Sub grdAttributes_RowColChange ()


 'every time a new cell is selected, the text appears .
 ' in textbox text1 and text1 receives focus.
 'Updates of text1 automatically appear in the grid.
If flgIgnoreMovement Then Exit Sub

   flgIgnoreMovement = True
   lngCurrentRow = grdAttributes.Row
   lngCurrentCol = grdAttributes.Col
   txtAttributes = grdAttributes.Text
   grdAttributes.Row = 0
   lblAttributes.Caption = grdAttributes.Text
   grdAttributes.Row = lngCurrentRow
   flgIgnoreMovement = False

End Sub

Sub GridClipNRowsNCols (ClipText As String, nRows As Long, nCols As Long)
   nCols = 1
   nRows = 1
   Dim TabChar As String * 1, CR  As String * 1
   CR = Chr$(13)
   TabChar = Chr$(9)
   Dim n As Long
' Find number of rows (delimeted by asc(13))
   For n = 1 To Len(ClipText)
  If Mid$(ClipText, n, 1) = CR Then
     nRows = nRows + 1
     nCols = nCols + 1
  End If
  If Mid$(ClipText, n, 1) = TabChar Then
     nCols = nCols + 1
  End If
   Next n
   nCols = nCols \ nRows
End Sub

Sub lblQSSource_Change ()
If lblQSSource = "CLEAR" Then
```

```
    grdAttributes.FixedRows = 1
    flgIgnoreMovement = True
    ClearGrid Me.grdAttributes
    flgIgnoreMovement = False
   Else
    SetQuery CStr(lblQSSource)
End If
End Sub

Sub lblQSType_Change ()
If flgChangesWereMade Then
     'Present a yes/no box to the user.
   Me.Show
   nF = MsgBox("Changes were made to this data set." & CRLF & "Do you wish to save
them now?", 36)
   If nF = 6 Then 'User chose to save changes
    Call mfSaveTableChanges_Click
    Else
    flgChangesWereMade = False
   End If
End If

End Sub

Sub ListQrysTblsInComboBox (DBIn As Database, ComboOut As ComboBox)
' this subroutine lists all queries and tables in a database
'  in a combobox, for later user selection.
    ComboOut.Clear
    Dim DB_SYSTEMOBJECT As Long
    DB_SYSTEMOBJECT = &H80000002

    Dim snpQTList As Snapshot 'declared locally so it goes away
    Set snpQTList = DBIn.ListTables()

    Dim QTName As String
    snpQTList.MoveFirst
    Do Until snpQTList.EOF 'For each Table and query
   If Not ((snpQTList("Attributes") And DB_SYSTEMOBJECT) <> 0) And
(snpQTList("Attributes") < 6) Then
     'Exclude system objects from list of tables and queries presented to user
     QTName = snpQTList("Name")
     ComboOut.AddItem QTName
   End If
   snpQTList.MoveNext
    Loop
    snpQTList.Close
    Set snpQTList = Nothing
End Sub

Sub meAddColumn_Click ()
 grdAttributes.Cols = grdAttributes.Cols + 1
End Sub

Sub meClearGrid_Click ()
```

```
529|  flgIgnoreMovement = True
530|   grdAttributes.FixedRows = 0
531|   grdAttributes.Rows = 1
532|   grdAttributes.Rows = 2
533|   grdAttributes.FixedRows = 1
534|  flgIgnoreMovement = False
535|
536|  End Sub
537|
538|  Sub meColDefinition_Click ()
539|   DefineColumn
540|  End Sub
541|
542|  Sub meCopy_Click ()
543|  Dim Current As Control
544|  Set Current = ActiveControl
545|  If TypeOf Current Is TextBox Then
546|     clipboard.Clear
547|     clipboard.SetText Current
548|  End If
549|  If TypeOf Current Is Grid Then
550|     Dim ClipText As String
551|     ClipText = Current.Clip
552|     clipboard.Clear
553|     clipboard.SetText ClipText
554|  End If
555|  End Sub
556|
557|  Sub meDeleteRow_Click ()
558|
559|     Dim n As Integer, nRows As Integer
560|
561|     flgIgnoreMovement = True
562|    grdAttributes.Col = 0
563|     For n = grdAttributes.SelStartRow To grdAttributes.SelEndRow
564|    grdAttributes.Row = n
565|    If grdAttributes <> "XXX" Then grdAttributes = "D"
566|     Next n
567|     flgChangesWereMade = True
568|     flgIgnoreMovement = False
569|  End Sub
570|
571|  Sub meInsertRow_Click ()
572|
573|
574|    Dim n As Long
575|    flgIgnoreMovement = True
576|    For n = grdAttributes.SelStartRow To grdAttributes.SelEndRow
577|      grdAttributes.AddItem "A"   ', (ActiveControl.Row) 'Chr$(9) & Chr$(9) & CR
578|    Next n
579|    flgChangesWereMade = True
580|    flgIgnoreMovement = False
581|  End Sub
582|
```

```
583| Sub mePaste_Click ()
584| Dim Current As Control
585| Set Current = ActiveControl
586| Dim ClipText As String
587| ClipText = clipboard.GetText()
588|
589| If TypeOf Current Is TextBox Then
590|    Dim Temp As String
591|    Temp = Current.Text
592|    'Check these XXXX use frmTileOptions as example
593|    Temp = Mid$(Temp, 1, Current.SelStart) & ClipText & Mid$(Temp, Current.SelStart +
     Current.SelLength)
594|    Current.Text = Temp
595| End If
596|
597| If TypeOf Current Is Grid Then
598|    Dim nRows As Long, nCols As Long
599|    Dim n As Long
600| ' Find number of rows (delimeted by asc(13))
601|    For n = 1 To Len(ClipText)
602|   If Mid$(ClipText, n, 1) = CR Then
603|      nRows = nRows + 1
604|   End If
605|   If Mid$(ClipText, n, 1) = TabChar Then
606|      nCols = nCols + 1
607|   End If
608|    Next n
609|
610|    If nRows = 0 Then
611|      nRows = 1
612|    End If
613|    If nCols = 0 Then
614|      nCols = 1
615|    End If
616|    nCols = nCols \ nRows
617|    Current.SelEndCol = Current.SelStartCol + nCols - 1
618|    Current.SelEndRow = Current.SelStartRow + nRows - 1
619|    Current.Clip = ClipText
620| End If
621|
622| End Sub
623|
624| Sub meUndoThisRow_Click ()
625|  If grdAttributes.Row <= nRecs Then
626|    flgIgnoreMovement = True
627|    Data1.Recordset.Bookmark = strBookMark(grdAttributes.Row)
628|    For nF = 1 To nFields
629|   grdAttributes.Col = nF
630|   If FieldType(nF) < 11 Then grdAttributes = Format(grdAttributes)
631|    Next nF
632|    grdAttributes.Col = 0
633|    grdAttributes = grdAttributes.Row
634|    flgIgnoreMovement = False
635|   Else
```

```
636|     grdAttributes.RemoveItem grdAttributes.Row
637|   End If
638| End Sub
639|
640| Sub mfAbandonTableChanges_Click ()
641|   flgIgnoreMovement = True
642|
643|   grdAttributes.Col = 0
644|   For nR = 1 To grdAttributes.Rows - 1
645|     grdAttributes = nR
646|   Next nR
647|   grdAttributes.Col = lngCurrentCol
648|   flgIgnoreMovement = False
649|   If lngCurrentRow < grdAttributes.Rows Then grdAttributes.Row = lngCurrentRow
650| End Sub
651|
652| Sub mfCloseDB_Click ()
653|  DB.Close
654|  Set DB = Nothing
655|   Data1.Recordset.Close
656|   cmbQTList.Clear
657|
658| End Sub
659|
660| Sub mfExit_Click ()
661|  ShutDownAndFreeMem
662| End Sub
663|
664| Sub mfNewFile_Click ()
665| Static OutFile As String, OutExt As String, DelimiterChosen As String
666|
667|   'Decide whether it should be Space, tab, or comma delimited.
668|    ReDim OZQuestion(1 To 5)  As typOZQuestion
669|    Dim Accepted As Integer
670|    OZQuestion(1).KindOfControl = CONTROL_LABEL
671|    OZQuestion(1).StringData = "What character do you want separating columns of
     data?"
672|    OZQuestion(2).KindOfControl = CONTROL_OPTIONBUTTON
673|    OZQuestion(2).StringData = "&Tab"
674|    OZQuestion(3).KindOfControl = CONTROL_OPTIONBUTTON
675|    OZQuestion(3).StringData = "&, Comma"
676|    OZQuestion(4).KindOfControl = CONTROL_OPTIONBUTTON
677|    OZQuestion(4).StringData = "&5 spaces"
678|    OZQuestion(5).KindOfControl = CONTROL_OPTIONBUTTON
679|    OZQuestion(5).StringData = "&; Semi-colon"
680| 'Consult the Wizard.
681|    AskOZ OZQuestion(), Accepted
682|    If Accepted Then
683|       If OZQuestion(2).Extra Then DelimiterChosen = Chr$(9): Me.CMDialog1.DefaultExt
     = "TDF"
684|       If OZQuestion(3).Extra Then DelimiterChosen = ",": : Me.CMDialog1.DefaultExt =
     "CSV"
685|       If OZQuestion(4).Extra Then DelimiterChosen = "     ": Me.CMDialog1.DefaultExt =
     "SDF"
```

```
686|       If OZQuestion(5).Extra Then DelimiterChosen = ";": Me.CMDialog1.DefaultExt =
     "TXT"
687|      Else
688|       Exit Sub
689|    End If
690|
691|   'Open a file
692|
693|  Me.CMDialog1.DialogTitle = "Text file to receive output of current grid"
694|  Me.CMDialog1.Filter = "Text|*.txt;*.tdf;*.sdf|Comma Separated Values
     (*.csv)|*.csv|Tab-delimted file(*.tdf)|*.tdf|All files (*.*)|(*.*)"
695|  Me.CMDialog1.Filename = OutFile
696|  Me.CMDialog1.Flags = OFN_OVERWRITEPROMPT Or OFN_PATHMUSTEXIST
697|
698|  Me.CMDialog1.Action = 1
699|  Dim Temp As String
700|
701|  OutFile = UCase$(Me.CMDialog1.Filename)
702|  If OutFile = "" Then Exit Sub
703|  Dim FileNum As Integer
704|  FileNum = FreeFile
705|  Open OutFile For Output As FileNum
706|
707|  'Write it out.
708|  Dim LineOut As String, Qt As String * 1, nR As Long, nC As Long, nRows As Long,
     nCols As Long
709|  nCols = grdAttributes.Cols
710|  nRows = grdAttributes.Rows
711|
712|  Qt = Chr$(34)
713|  flgIgnoreMovement = True
714|   grdAttributes.Row = 0
715|  'For first line, add a SerialNumber label and put quotes around everything
716|   LineOut = Qt & "SerialNumber" & Qt & DelimiterChosen
717|     For nC = 1 To nCols - 1
718|    grdAttributes.Col = nC
719|    LineOut = LineOut & Qt & grdAttributes & Qt & DelimiterChosen
720|     Next nC
721|   LineOut = Left(LineOut, Len(LineOut) - Len(DelimiterChosen))'truncate extra delimiter at
     end  of line
722|   Print #FileNum, LineOut
723|  For nR = 1 To grdAttributes.Rows - 1
724|     grdAttributes.Row = nR
725|  'No quotes around values here.
726|     LineOut = nR & DelimiterChosen
727|     For nC = 1 To nCols - 1
728|       grdAttributes.Col = nC
729|       LineOut = LineOut & grdAttributes & DelimiterChosen
730|     Next nC
731|  LineOut = Left(LineOut, Len(LineOut) - Len(DelimiterChosen))'truncate extra delimiter at
     end  of line
732|  Print #FileNum, LineOut
733|  Next nR
734|  flgIgnoreMovement = False
```

```
735|
736| 'Close the file
737|
738| Close FileNum
739|
740| End Sub
741|
742| Sub mfNewTable_Click ()
743|
744|   Dim NewTable As New TableDef
745|    Dim TempVar As Variant
746|   Dim n As Long, nCols As Long, m As Long, nRows As Long, TableName As String
747|   Dim nC As Long, nR As Long
748|   nCols = grdAttributes.Cols
749| On Error GoTo ERRmfNewTable
750| TableName = "User Points " & Format$(MapDB.TableDefs.Count, "0000")
751| TableName = InputBox$("Enter a name for the table to be saved as.", "Saving Grid to
    Table", TableName)
752|
753| If TableName = "" Then
754|    ' Erase NewField
755|     Exit Sub
756| End If
757|   NewTable.Name = TableName
758| 'Add the serial number field.
759|   Dim LatestField As New field
760|   LatestField.Type = DB_LONG
761|   LatestField.Name = "Serial Number"
762|   NewTable.Fields.Append LatestField
763|   MapDB.TableDefs.Append NewTable
764|   Set LatestField = Nothing
765|   Set NewTable = Nothing
766|
767| 'Get the newly created table
768|  Dim tblOut As Table
769|
770|
771| 'This will take the first row of the grdAttributes grid,
772| ' and use it to define the names and types for an output table.
773| 'No provision is made for being able to append to a table
774| '   or to overwrite a table.
775| 'Command decision by Greg Pouch, 22 June 1994
776|
777|   'ReDim NewField(0 To nCols) As New Field
778|   ReDim FieldNames(0 To nCols) As String
779|
780| Dim Temp As String, tmpFieldName As String, tmpFieldType As Integer
781| Dim strFieldTypeList As String
782| flgIgnoreMovement = True
783| grdAttributes.Row = 0
784|
785|
786| For n = 1 To grdAttributes.Cols - 1
787|    grdAttributes.Col = n
```

```
    Temp = grdAttributes
    If Temp = "" Then
   Temp = InputBox$("Please enter a name for this field, in column " & n, "Saving Table",
"Field" & Format$(n, "00"))
    End If
    tmpFieldName = Left(Temp, Len(Temp) - 3)
    tmpFieldType = Val(Right(Temp, 3))
    If tmpFieldType = 0 Then
   'We have an untagged field, so ask the user whether
   'to skip it or to define the data type.
   ReDim OZQuestion(1 To 2)  As typOZQuestion
   Dim Accepted As Integer
   OZQuestion(1).KindOfControl = CONTROL_LABEL
   OZQuestion(1).StringData = "This field, " & Temp & ", has no data type associated it
with it." & CRLF & "Choose one from the list box."
   OZQuestion(2).KindOfControl = CONTROL_COMBOBOX
   strFieldTypeList = ListOfFieldTypes
   OZQuestion(2).StringData = strFieldTypeList
   AskOZ OZQuestion(), Accepted
   If Accepted Then
     tmpFieldType = OZQuestion(2).Extra
   End If
    End If

  If tmpFieldType <> 0 And tmpFieldName <> "" Then
    'if the above information exists, add the field
    Dim VeryNewField As New field
   VeryNewField.Type = tmpFieldType
   If tmpFieldType = 10 Then VeryNewField.Size = 255
   VeryNewField.Name = Trim$(tmpFieldName)
   MapDB.TableDefs(TableName).Fields.Append VeryNewField
   Set VeryNewField = Nothing
   FieldNames(n) = tmpFieldName
     Else 'Otherwise, mark it to be skipped.
      FieldNames(n) = ""
  End If
 Next n
 'At this point, all the columns have been assigned a field name,
 ' or not if the user doesn't want to keep it, and we should save the table
 ' and start writing to it.
 Set tblOut = MapDB.OpenTable(TableName)
 On Error Resume Next
 For nR = 1 To grdAttributes.Rows
    grdAttributes.Row = nR
    tblOut.AddNew
    tblOut("Serial Number") = nR
    For nC = 1 To nCols
      grdAttributes.Col = nC
      If (FieldNames(nC) <> "") Then
    TempVar = grdAttributes
    tblOut(FieldNames(nC)) = TempVar
      End If
    Next nC
    tblOut.Update
```

```
840| Next nR
841|     Set tblOut = Nothing
842|
843|
844| Exit Sub
845| ERRmfNewTable:
846| Select Case Err
847|  Case 3010 'Table of that name already exists
848|    TableName = InputBox$("Table " & TableName & " already exists.  Please choose a
     new name.", "Saving Grid to Table", "User Table" & Format$(Now))
849|   If TableName = "" Then
850|      Err = 0
851|      Exit Sub
852|      Resume Next
853|   End If
854|    NewTable.Name = TableName
855|    Resume
856|  Case Else
857|    MsgBox "Error #" & Err & " occured in frmGridEdit.mfNewTable." & CRLF & Error
858|    Err = 0
859|    Exit Sub
860|    Resume
861| End Select
862|
863|
864| End Sub
865|
866| Sub mfOpenDB_Click ()
867| Dim DBName As String' May want to change scope of this later to global.
868| CMDialog1.DialogTitle = "Database to Open"
869| CMDialog1.Filter = "MS Access Databases|*.mdb"
870| CMDialog1.Flags = &H1000&    'OFN_FILEMUSTEXIST
871| CMDialog1.Action = 1
872| Dim Temp As String
873|
874| Temp = CMDialog1.Filename
875| If Len(Temp) = 0 Then 'the user did not enter a file name
876|    Else
877|    DBName = Temp
878|    ' Open an MSAccess database for non-exclusive, read-write access.
879|    Set DB = OpenDatabase(DBName, False, False, "") 'Open the desired database
880|    Data1.DatabaseName = DBName
881|    Me.Caption = "Grid-style data editor " & DBName
882|    Call ListQrysTblsInComboBox(DB, cmbQTList)
883|
884|    cmbQTList.SetFocus
885|    SendKeys "%{DOWN}"
886| End If
887|
888| End Sub
889|
890| Sub mfSaveTableChanges_Click ()
891|   SaveGridToDynaset
892| End Sub
```

```
893|
894| Sub SaveGridToDynaset ()
895| On Error GoTo ERRSaveAttributes
896| If Not flgChangesWereMade Then Exit Sub
897| flgIgnoreMovement = True
898|
899| 'Might want to change this so that it doesn't reset row number
900| 'This way, multiple records could be displayed and edited.
901| Dim nRows As Long, nRec As Long, Col0 As String
902| nRows = grdAttributes.Rows - 1
903| For nRec = 1 To nRows
904|    grdAttributes.Row = nRec
905| 'Examine the zeroeth column, which could contain a flag for NEW, EDITED, DELETED,
    etc.
906|    grdAttributes.Col = 0
907|    Col0 = grdAttributes
908|   If Col0 = "E" Then
909|    Data1.Recordset.Bookmark = strBookMark(nRec)
910|    Data1.Recordset.Edit
911|    For nF = 1 To nFields
912|   If FieldType(nF) < 11 Then 'don't update memo and BLOB objects, because the grid
    control can lose them due to chr(0)s
913|       grdAttributes.Col = nF
914|       Data1.Recordset.Fields(nF - 1) = grdAttributes
915|   End If
916|    Next nF
917|    Data1.Recordset.Update
918|   End If
919|
920|   If Col0 = "D" Then
921|     grdAttributes = "XXX"
922|     Data1.Recordset.Bookmark = strBookMark(nRec)
923|     Data1.Recordset.Delete
924|   End If
925|
926|   If Col0 = "A" Then
927|    grdAttributes = ""
928|    Data1.Recordset.AddNew
929|    For nF = 1 To nFields
930|   If FieldType(nF) < 11 Then 'don't update memo and BLOB objects, because the grid
    control can lose them due to chr(0)s
931|       grdAttributes.Col = nF
932|       Data1.Recordset.Fields(nF - 1) = grdAttributes
933|   End If
934|    Next nF
935|    Data1.Recordset.Update
936|   End If
937|
938| Next nRec
939|
940|
941| flgIgnoreMovement = False
942| flgChangesWereMade = False
943|
```

```
Exit Sub
ERRSaveAttributes:

If Err = 3021 Then 'No current record
   Data1.Recordset.AddNew
   Resume Next
End If
If Err = 91 Then 'Object variable not set.  ie, no XYBLOB to update
  Resume Next
End If
If Err = 3058 Then
   grdAttributes.SelStartRow = grdAttributes.Row
   grdAttributes.SelEndRow = grdAttributes.Row
   grdAttributes.SelStartCol = 1
   grdAttributes.SelEndRow = grdAttributes.Cols - 1
   n = MsgBox("You have not entered a value in a field that is part of a primary index." &
CRLF & "This is not allowed.  Do you want to delete this record?", 36)
   If n = 6 Then
      grdAttributes.Col = 0
      grdAttributes = "XXX"
      Resume Next
     Else
      Err = 0
      Exit Sub
      Resume Next
   End If
End If
If Err = 9 Then 'Subscript out of range
   Resume Next 'Try ignoring it
End If



MsgBox "Error number " & Err & " encountered while saving attributes." & Chr$(13) &
Chr$(10) & Error$
Err = 0
Exit Sub
Resume

End Sub

Sub SetIgnoreMovement (ToValue As Long)
 flgIgnoreMovement = ToValue
End Sub

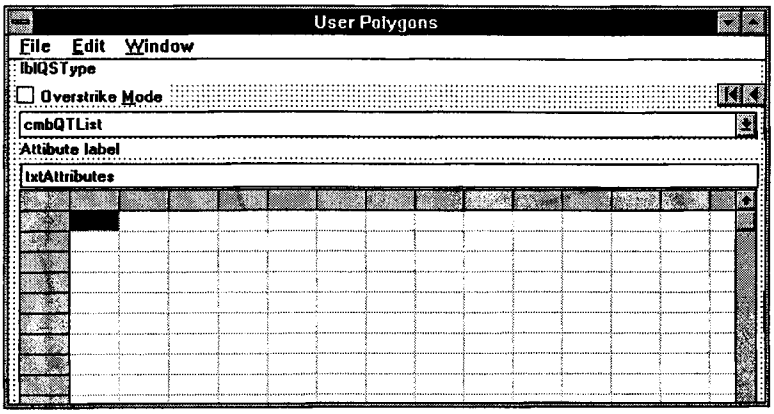
Sub SetQuery (strTblQrySQL As String)
'This subroutine sets the data control to the desired TaBLe, QueRY, or SQL string,
' then sets the labels for the attribute grid.
On Error GoTo ERRSetQuery

'Check whether changes need to be saved.
If flgChangesWereMade Then
     'Present a yes/no/cancel box to the user.
```

```
 996|    Me.Show
 997|    nF = MsgBox("Changes were made to this data set." & CRLF & "Do you wish to save
      them now?", 36)
 998|    If nF = 2 Then Exit Sub'user chose cancel, so clear out.
 999|    If nF = 6 Then 'User chose to save changes
1000|      Call mfSaveTableChanges_Click
1001|    End If
1002|    flgChangesWereMade = False  'If the user chose yes or no, reset the flag.
1003|  End If
1004|
1005|  'Set flag to not display cell contents in text box.
1006|  flgIgnoreMovement = True
1007|
1008|  'Clear the grid
1009|  ClearGrid Me.grdAttributes
1010|
1011|  grdAttributes.RowHeight(0) = 600
1012|  'Set the data control to the desired query, table, or SQL statement.
1013|  Data1.RecordSource = strTblQrySQL
1014|  Data1.Refresh
1015|
1016|  If Data1.Recordset.Bookmarkable Then
1017|     'ReDim strBookMark(1 To nRecs)
1018|     flgBookmarkable = True
1019|     ZUnKillMenuItems
1020|   Else
1021|     mfSaveTableChanges.Enabled = False
1022|     MsgBox "You will not be able to save changes to this table or query because it cannot
      be bookmarked."
1023|     ZKillMenuItems
1024|     flgBookmarkable = False
1025|  End If
1026|
1027|
1028|  flgIgnoreMovement = False
1029|  grdAttributes.Visible = True
1030|
1031|  Exit Sub
1032|
1033|  ERRSetQuery:
1034|  If Err = 3061 Then
1035|     MsgBox "You tried selecting a parameter query, and this application doesn't do those.
      Sorry!"
1036|     Err = 0
1037|     Exit Sub
1038|  End If
1039|
1040|  MsgBox "Error number " & Err & " occurred in SetQuery." & CRLF & Error
1041|  Err = 0
1042|  'Set cursor back to normal.
1043|  mousepointer = 0
1044|  grdAttributes.Visible = True
1045|  Exit Sub
1046|  Resume
```

```
1047| End Sub
1048|
1049| Sub txtAttributes_GotFocus ()
1050|  If chkOverstrike Then
1051|    txtAttributes.SelStart = 0
1052|    txtAttributes.SelLength = 32765
1053|  End If
1054|
1055| End Sub
1056|
1057| Sub txtAttributes_KeyPress (KeyAscii As Integer)
1058|
1059|  If KeyAscii = 13 Then  'For a return key, move focus to the
1060|    KeyAscii = 0        ' grid, then save the changes to the grid.
1061|    'If Not flgBookmarkable Then
1062|    '   Beep
1063|    '   Beep
1064|    '   txtAttributes = grdAttributes
1065|    '   Exit Sub
1066|    'End If
1067|
1068|    flgIgnoreMovement = True
1069|    grdAttributes.Col = 0
1070|    If grdAttributes <> "A" Then grdAttributes = "E"
1071|    grdAttributes.Col = lngCurrentCol
1072|    flgIgnoreMovement = False
1073|    flgChangesWereMade = True
1074|    grdAttributes.SetFocus
1075|  End If
1076|  If KeyAscii = 27 Then   'For an ESCape key, reset the text box to
1077|    txtAttributes = grdAttributes    'the contents of the grid cell.
1078|    KeyAscii = 0
1079|  End If
1080|
1081|
1082| End Sub
1083|
1084| Sub txtAttributes_LostFocus ()
1085|    grdAttributes.Text = txtAttributes
1086| End Sub
1087|
1088| Sub UndoDataChanges ()
1089|  Call Data1_Reposition
1090| End Sub
1091|
1092| Sub ZKillMenuItems ()
1093|   zzEdit.Enabled = False
1094|   mfAbandonTableChanges.Enabled = False
1095|   mfSaveTableChanges.Enabled = False
1096| End Sub
1097|
1098| Sub ZUnKillMenuItems ()
1099|   zzEdit.Enabled = True
1100|   mfAbandonTableChanges.Enabled = True
```

```
1101|    mfSaveTableChanges.Enabled = True
1102| End Sub
1103|
```


User Polygons
File Edit Window
lblQSType
Overstrike Mode
cmbQTList
Attibute label
txtAttributes

```
VERSION 2.00
Begin Form frmUserPolys
  Caption        = "User Polygons"
  ClientHeight   = 4005
  ClientLeft     = 345
  ClientTop      = 990
  ClientWidth    = 8745
  Height         = 4695
  Icon           = USERPOLY.FRX:0000
  Left           = 285
  LinkTopic      = "Form1"
  MDIChild       = -1 'True
  ScaleHeight    = 70.644
  ScaleMode      = 6 'Millimeter
  ScaleWidth     = 154.252
  Top            = 360
  Width          = 8865
  Begin ComboBox cmbQTList
    Height       = 300
    Left         = 120
    Sorted       = -1 'True
    Style        = 2 'Dropdown List
    TabIndex     = 6
    Top          = 600
    Visible      = 0 'False
    Width        = 8535
  End
  Begin CommonDialog CMDialog2
    Left         = 8730
    Top          = 450
  End
  Begin CommonDialog CMDialog1
    Left         = 8730
    Top          = 2700
  End
  Begin Data Data1
    Caption      = "Move record "
    Connect      = ""
    DatabaseName = ""
    Exclusive    = 0 'False
    Height       = 270
    Left         = 8190
    Options      = 0
    ReadOnly     = 0 'False
    RecordSource = ""
    Top          = 270
    Visible      = 0 'False
    Width        = 2445
  End
  Begin CheckBox chkOverstrike
    Caption      = "Overstrike &Mode"
    Height       = 255
    Left         = 90
    TabIndex     = 3
```

```
    Top             = 285
    Width           = 1757
  End
  Begin TextBox txtAttributes
    Height          = 285
    Left            = 135
    TabIndex        = 1
    Text            = "txtAttributes"
    Top             = 1200
    Width           = 8535
  End
  Begin Grid grdAttributes
    Cols            = 15
    Height          = 4515
    Left            = 120
    Rows            = 30
    TabIndex        = 0
    Top             = 1500
    Width           = 8535
  End
  Begin Label lblQSSource
    Caption         = "lblQSSource"
    Height          = 255
    Left            = 120
    TabIndex        = 5
    Top             = 600
    Width           = 8415
  End
  Begin Label lblQSType
    Caption         = "lblQSType"
    Height          = 255
    Left            = 120
    TabIndex        = 4
    Top             = 0
    Width           = 8535
  End
  Begin Label lblAttributes
    Caption         = "Attibute label"
    Height          = 210
    Left            = 135
    TabIndex        = 2
    Top             = 930
    Width           = 8535
  End
  Begin Menu zzzzFile
    Caption         = "&File"
    Begin Menu mfOpenDB
      Caption       = "&Open Database..."
      Visible       = 0  'False
    End
    Begin Menu mfCloseDB
      Caption       = "&Close Database"
      Visible       = 0  'False
    End
```

```
109|     Begin Menu zz10
110|       Caption      = "-"
111|     End
112|     Begin Menu mfNewTable
113|       Caption      = "New &Table..."
114|     End
115|     Begin Menu zzz1
116|       Caption      = "-"
117|     End
118|     Begin Menu mfNewFile
119|       Caption      = "New &File..."
120|     End
121|     Begin Menu zz3
122|       Caption      = "-"
123|       Visible      = 0  'False
124|     End
125|     Begin Menu mfSaveTableChanges
126|       Caption      = "&Save Changes"
127|       Visible      = 0  'False
128|     End
129|     Begin Menu mfAbandonTableChanges
130|       Caption      = "&Abandon Changes"
131|       Visible      = 0  'False
132|     End
133|     Begin Menu zzzzz1
134|       Caption      = "-"
135|     End
136|     Begin Menu mfExit
137|       Caption      = "E&xit"
138|     End
139|   End
140|   Begin Menu zzEdit
141|     Caption      = "&Edit"
142|     Begin Menu meUndoThisRow
143|       Caption      = "&Undo changes to this row"
144|       Shortcut     = %{BKSP}
145|     End
146|     Begin Menu zz6
147|       Caption      = "-"
148|     End
149|     Begin Menu meCopy
150|       Caption      = "&Copy"
151|       Shortcut     = ^{INSERT}
152|     End
153|     Begin Menu mePaste
154|       Caption      = "&Paste"
155|       Shortcut     = +{INSERT}
156|     End
157|     Begin Menu meCut
158|       Caption      = "Cu&t"
159|       Shortcut     = +{DEL}
160|     End
161|     Begin Menu meClearGrid
162|       Caption      = "C&lear"
```

```
163|       Shortcut      = ^X
164|     End
165|     Begin Menu zzzz1
166|       Caption       = "-"
167|     End
168|     Begin Menu meInsertRow
169|       Caption       = "&Append Records"
170|       Shortcut      = ^A
171|     End
172|     Begin Menu meDeleteRow
173|       Caption       = "&Delete Records"
174|       Shortcut      = ^D
175|     End
176|     Begin Menu zzz5
177|       Caption       = "-"
178|     End
179|     Begin Menu meAddColumn
180|       Caption       = "Add Co&lumn"
181|     End
182|     Begin Menu meColDefinition
183|       Caption       = "Column De&finition"
184|     End
185|   End
186|   Begin Menu mnuWindow
187|     Caption       = "&Window"
188|     WindowList    = -1  'True
189|   End
190| End
191| Option Explicit
192|
193| Dim flgChangeRow As Integer, flgIgnoreMovement As Integer
194| Dim flgChangesWereMade As Integer, flgBookmarkable As Integer
195|
196| Dim strBookMark() As String
197| Dim flgSkipMe() As Long
198| Dim nFields As Long, n As Long, nRecs As Long, nR As Long
199| Dim nF As Long, FieldType() As Long
200| Dim lngCurrentRow As Long, lngCurrentCol As Long
201|
202| Dim DB As Database, tblLineData As Table
203| 'Dim snpQTList As Snapshot, dnsCurrentQT As dynaset
204|
205| Dim flgAddingRecord As Integer   'Might want to eliminate this variable.
206| Dim TabChar As String * 1, CR  As String * 1, CRLF  As String * 2
207| Dim ListOfFieldTypes As String
208|
209| Sub cmbQTList_Click ()
210| Dim Temp As String
211| mousepointer = 11
212|
213| Temp = cmbQTList
214| Call SetQuery(Temp)
215| flgIgnoreMovement = True
216| grdAttributes.FixedCols = 0: grdAttributes.Cols = 1
```

```
217| grdAttributes.Cols = 2: grdAttributes.FixedCols = 1
218| grdAttributes.Rows = 2
219| 'Function GetClipBoardTextFromDynaset (dynIn As Dynaset, flgOptions As Long) As
     String
220| Set dynViewInGrid = DB.CreateDynaset(Temp)
221| Call InitializeGridForDynaset(Me!grdAttributes, dynViewInGrid)
222|
223| Temp = GetClipBoardTextFromDynaset(dynViewInGrid, 0)
224| Call SetGridFocusColumn(Me!grdAttributes, 1)
225| Call FillGridFromClipBoardText(Temp, Me!grdAttributes, 12&)
226|
227| grdAttributes.RemoveItem 1
228|
229|
230| 'Call OverwriteGridFromDynaset(Me!grdAttributes, dynViewInGrid)
231| flgIgnoreMovement = False
232| mousepointer = 0
233|
234| End Sub
235|
236| Sub cmdClearGrid_Click ()
237|  flgIgnoreMovement = True
238|  grdAttributes.FixedRows = 0
239|  grdAttributes.Rows = 1
240|  grdAttributes.Rows = 2
241|  grdAttributes.FixedRows = 1
242|  flgIgnoreMovement = False
243| End Sub
244|
245| Sub Data1_Reposition ()
246| 'On Error Resume Next
247| 'DisplayCurrentRecord
248| End Sub
249|
250| Sub DefineColumn ()
251|    flgIgnoreMovement = True
252|    Dim Temp As String, tmpFieldName As String, tmpFieldType As Integer,
     strFieldTypeList As String
253|    grdAttributes.Row = 0
254|    n = grdAttributes.Col
255|    Temp = grdAttributes
256|    If Temp = "" Then
257|   tmpFieldName = "Field" & Format$(n, "00")
258|   tmpFieldType = 10'Default to text
259|     Else
260|   tmpFieldType = Val(Right(Temp, 3))
261|   If tmpFieldType <> 0 Then
262|      tmpFieldName = Left(Temp, Len(Temp) - 3)
263|      Else
264|      tmpFieldName = Temp
265|      tmpFieldType = 10
266|   End If
267|      End If
268|
```

```
269|    ReDim OZQuestion(1 To 4)  As typOZQuestion
270|    Dim Accepted As Integer
271|    'Define the name.
272|    OZQuestion(1).KindOfControl = CONTROL_LABEL
273|    OZQuestion(1).StringData = "This field is currently named: " & Temp & " ." & CRLF &
       "If you wish, you may enter a new name below." '& CRLF & "and droning on and on and
       on about absolutely nothing at all in the world of any possible interest to anyone."
274|    OZQuestion(2).KindOfControl = CONTROL_TEXTBOX
275|    OZQuestion(2).StringData = tmpFieldName
276|
277|    'Define the datafield type.
278|    OZQuestion(3).KindOfControl = CONTROL_LABEL
279|    OZQuestion(3).StringData = "It currently has a datatype of " & tmpFieldType
280|    OZQuestion(4).KindOfControl = CONTROL_COMBOBOX
281|    strFieldTypeList = ListOfFieldTypes
282|    OZQuestion(4).StringData = strFieldTypeList
283|    OZQuestion(4).Extra = tmpFieldType
284|  'Consult the Wizard.
285|    AskOZ OZQuestion(), Accepted
286|
287|    If Accepted Then
288|       tmpFieldType = OZQuestion(4).Extra
289|       tmpFieldName = OZQuestion(2).StringData
290|       grdAttributes = tmpFieldName & " " & Format$(tmpFieldType, "00")
291|    End If
292|    flgIgnoreMovement = False
293|
294|  End Sub
295|
296|  Sub DisplayCurrentRecord ()
297|  On Error GoTo ERRDisplayCurrentRecord
298|  flgIgnoreMovement = True
299|  grdAttributes.Row = 1
300|  Dim Temp As String
301|  For n = 1 To nFields
302|     grdAttributes.Col = n
303|     grdAttributes = Format(Data1.Recordset.Fields(n - 1).Value)
304|  Next n
305|
306|
307|  flgIgnoreMovement = False
308|  Exit Sub
309|
310|  ERRDisplayCurrentRecord:
311|  If Err = 30010 Then 'Invalid number of columns or rows
312|     Err = 0
313|     Exit Sub
314|     Resume Next
315|  End If
316|  MsgBox "Error number " & Err & " encountered while displaying a record." & CRLF &
       Error$
317|  Err = 0
318|  Exit Sub
319|  Resume
```

```
320|
321|
322|
323| End Sub
324|
325| Sub Form_Load ()
326|
327| CR = Chr$(13)
328| TabChar = Chr$(9)
329| CRLF = Chr$(13) & Chr$(10)
330| ListOfFieldTypes = ""
331|
332| 'cmbTypeList.Clear
333| 'cmbTypeList.AddItem "Do not save this field", 0
334| 'cmbTypeList.AddItem "BOOLEAN", 1
335| 'cmbTypeList.AddItem "BYTE", 2
336| 'cmbTypeList.AddItem "Short Integer", 3
337| 'cmbTypeList.AddItem "Long Integer 4-bytes", 4
338| 'cmbTypeList.AddItem "CURRENCY", 5
339| 'cmbTypeList.AddItem "Single Precision REAL*4", 6
340| 'cmbTypeList.AddItem "Double Precision REAL*8", 7
341| 'cmbTypeList.AddItem "DATE", 8
342| 'cmbTypeList.AddItem "Do not Save this field", 9
343| 'cmbTypeList.AddItem "TEXT", 10
344| 'cmbTypeList.AddItem "LONGBINARY", 11
345| 'cmbTypeList.AddItem "MEMO (Long String)", 12
346|
347|
348| ListOfFieldTypes = "Do not save this field" & CRLF & "BOOLEAN" & CRLF & "BYTE" &
    CRLF
349| ListOfFieldTypes = ListOfFieldTypes & "Short Integer" & CRLF & "Long Integer 4-bytes"
    & CRLF
350| ListOfFieldTypes = ListOfFieldTypes & "CURRENCY" & CRLF & "Single Precision
    REAL*4" & CRLF & "Double Precision REAL*8" & CRLF
351| ListOfFieldTypes = ListOfFieldTypes & "DATE" & CRLF & "Do not Save this field" &
    CRLF & "TEXT" & CRLF & "LONGBINARY" & CRLF & "MEMO (Long String)" & CRLF
352|
353| Me!grdAttributes.Cols = 5
354| flgIgnoreMovement = True
355|    Me!grdAttributes.Row = 0
356|
357|    Me!grdAttributes.Col = 1
358|    Me!grdAttributes = "Name 10"
359|    Me!grdAttributes.ColWidth(1) = 1200
360|
361|    Me!grdAttributes.Col = 2
362|    Me!grdAttributes = "XYCoordinatesAsText 12"
363|    Me!grdAttributes.ColWidth(2) = 3600
364|
365|    Me!grdAttributes.Col = 3
366|    Me!grdAttributes = "TimeStamp 08"
367|    Me!grdAttributes.ColWidth(3) = 1800
368|
369|    Me!grdAttributes.Col = 4
```

```
370|     Me!grdAttributes = "SequenceNumber 08"
371|     Me!grdAttributes.ColWidth(4) = 1200
372|
373|     Me!grdAttributes.Rows = 2
374| flgIgnoreMovement = False
375| End Sub
376|
377| Sub Form_Paint ()
378| flgIgnoreMovement = False
379| End Sub
380|
381| Sub Form_QueryUnload (Cancel As Integer, UnloadMode As Integer)
382| 'Check whether changes need to be saved.
383|
384| If flgChangesWereMade Then
385|       'Present a yes/no/cancel box to the user.
386|    nF = MsgBox("Changes were made to this data set." & CRLF & "Do you wish to save
      them now?", 35)
387|    If nF = 2 Then
388|       Cancel = True
389|       Exit Sub'user chose cancel, so clear out.
390|       End If
391|    If nF = 6 Then 'User chose to save changes
392|      Call mfSaveTableChanges_Click
393|      If flgChangesWereMade Then Cancel = True 'Still need to cancel
394|    End If
395|    'flgChangesWereMade = False  'If the user chose yes or no, reset the flag.
396| End If
397|
398| End Sub
399|
400| Sub Form_Resize ()
401| If frmGridDataEdit.WindowState = 1 Then Exit Sub'iconized
402| On Error GoTo ERRResize
403|     lblAttributes.Width = frmGridDataEdit.ScaleWidth - 5
404|     txtAttributes.Width = frmGridDataEdit.ScaleWidth - 5
405|     grdAttributes.Width = frmGridDataEdit.ScaleWidth - 5
406|     grdAttributes.Height = frmGridDataEdit.ScaleHeight - 27.5
407|
408| cmbQTList.Width = Me.ScaleWidth - 10
409|
410| Exit Sub
411|
412| ERRResize:
413| If Err = 380 Then 'invalid width
414|   frmGridDataEdit.Width = frmGridDataEdit.Width * 1.2
415|   Resume
416| End If
417|
418|
419| End Sub
420|
421| Sub GetIgnoreMovement (FromValue As Long)
422|  FromValue = flgIgnoreMovement
```

```
End Sub

Sub grdAttributes_KeyDown (Keycode As Integer, Shift As Integer)
If Keycode > 48 Then
   txtAttributes.SetFocus
End If
End Sub

Sub grdAttributes_KeyPress (KeyAscii As Integer)
   If KeyAscii = 13 Then SendKeys "{DOWN}"
   If KeyAscii > 31 Then SendKeys Chr$(KeyAscii)
End Sub

Sub grdAttributes_RowColChange ()
  'every time a new cell is selected, the text appears
  ' in textbox text1 and text1 receives focus.
  'Updates of text1 automatically appear in the grid.
If flgIgnoreMovement Then Exit Sub

   flgIgnoreMovement = True
   lngCurrentRow = grdAttributes.Row
   lngCurrentCol = grdAttributes.Col
   txtAttributes = grdAttributes.Text
   grdAttributes.Row = 0
   lblAttributes.Caption = grdAttributes.Text
   grdAttributes.Row = lngCurrentRow
   flgIgnoreMovement = False

End Sub

Sub GridClipNRowsNCols (ClipText As String, nRows As Long, nCols As Long)
   nCols = 1
   nRows = 1
   Dim TabChar As String * 1, CR  As String * 1
   CR = Chr$(13)
   TabChar = Chr$(9)
   Dim n As Long
' Find number of rows (delimeted by asc(13))
   For n = 1 To Len(ClipText)
  If Mid$(ClipText, n, 1) = CR Then
     nRows = nRows + 1
     nCols = nCols + 1
  End If
  If Mid$(ClipText, n, 1) = TabChar Then
     nCols = nCols + 1
  End If
   Next n
   nCols = nCols \ nRows
End Sub

Sub lblQSSource_Change ()
If lblQSSource = "CLEAR" Then
   grdAttributes.FixedRows = 1
   flgIgnoreMovement = True
```

```
477|     ClearGrid Me.grdAttributes
478|     flgIgnoreMovement = False
479|    Else
480|     SetQuery CStr(lblQSSource)
481| End If
482| End Sub
483|
484| Sub lblQSType_Change ()
485| If flgChangesWereMade Then
486|       'Present a yes/no box to the user.
487|    Me.Show
488|    nF = MsgBox("Changes were made to this data set." & CRLF & "Do you wish to save
     them now?", 36)
489|    If nF = 6 Then 'User chose to save changes
490|      Call mfSaveTableChanges_Click
491|     Else
492|      flgChangesWereMade = False
493|    End If
494| End If
495|
496| End Sub
497|
498| Sub ListQrysTblsInComboBox (DBIn As Database, ComboOut As ComboBox)
499| ' this subroutine lists all queries and tables in a database
500| '   in a combobox, for later user selection.
501|    ComboOut.Clear
502|    Dim DB_SYSTEMOBJECT As Long
503|    DB_SYSTEMOBJECT = &H80000002
504|
505|    Dim snpQTList As Snapshot 'declared locally so it goes away
506|    Set snpQTList = DBIn.ListTables()
507|
508|    Dim QTName As String
509|    snpQTList.MoveFirst
510|    Do Until snpQTList.EOF 'For each Table and query
511|    If Not ((snpQTList("Attributes") And DB_SYSTEMOBJECT) <> 0) And
     (snpQTList("Attributes") < 6) Then
512|      'Exclude system objects from list of tables and queries presented to user
513|      QTName = snpQTList("Name")
514|      ComboOut.AddItem QTName
515|    End If
516|    snpQTList.MoveNext
517|     Loop
518|     snpQTList.Close
519|     Set snpQTList = Nothing
520| End Sub
521|
522| Sub meAddColumn_Click ()
523|  grdAttributes.Cols = grdAttributes.Cols + 1
524| End Sub
525|
526| Sub meClearGrid_Click ()
527|  flgIgnoreMovement = True
528|   grdAttributes.FixedRows = 0
```

```
529|   grdAttributes.Rows = 1
530|   grdAttributes.Rows = 2
531|   grdAttributes.FixedRows = 1
532|  flgIgnoreMovement = False
533|
534|  End Sub
535|
536|  Sub meColDefinition_Click ()
537|   DefineColumn
538|  End Sub
539|
540|  Sub meCopy_Click ()
541|  Dim Current As Control
542|  Set Current = ActiveControl
543|  If TypeOf Current Is TextBox Then
544|     clipboard.Clear
545|     clipboard.SetText Current
546|  End If
547|  If TypeOf Current Is Grid Then
548|     Dim ClipText As String
549|     ClipText = Current.Clip
550|     clipboard.Clear
551|     clipboard.SetText ClipText
552|  End If
553|  End Sub
554|
555|  Sub meDeleteRow_Click ()
556|
557|     Dim n As Integer, nRows As Integer
558|
559|     flgIgnoreMovement = True
560|    grdAttributes.Col = 0
561|     For n = grdAttributes.SelStartRow To grdAttributes.SelEndRow
562|    grdAttributes.Row = n
563|    If grdAttributes <> "XXX" Then grdAttributes = "D"
564|     Next n
565|     flgChangesWereMade = True
566|     flgIgnoreMovement = False
567|  End Sub
568|
569|  Sub meInsertRow_Click ()
570|
571|
572|    Dim n As Long
573|    flgIgnoreMovement = True
574|    For n = grdAttributes.SelStartRow To grdAttributes.SelEndRow
575|      grdAttributes.AddItem "A"   ', (ActiveControl.Row) 'Chr$(9) & Chr$(9) & CR
576|    Next n
577|    flgChangesWereMade = True
578|    flgIgnoreMovement = False
579|  End Sub
580|
581|  Sub mePaste_Click ()
582|  Dim Current As Control
```

```
583| Set Current = ActiveControl
584| Dim ClipText As String
585| ClipText = clipboard.GetText()
586|
587| If TypeOf Current Is TextBox Then
588|    Dim Temp As String
589|    Temp = Current.Text
590|    'Check these XXXX use frmTileOptions as example
591|    Temp = Mid$(Temp, 1, Current.SelStart) & ClipText & Mid$(Temp, Current.SelStart +
     Current.SelLength)
592|    Current.Text = Temp
593| End If
594|
595| If TypeOf Current Is Grid Then
596|    Dim nRows As Long, nCols As Long
597|    Dim n As Long
598| ' Find number of rows (delimeted by asc(13))
599|    For n = 1 To Len(ClipText)
600|   If Mid$(ClipText, n, 1) = CR Then
601|      nRows = nRows + 1
602|   End If
603|   If Mid$(ClipText, n, 1) = TabChar Then
604|      nCols = nCols + 1
605|   End If
606|    Next n
607|
608|    If nRows = 0 Then
609|      nRows = 1
610|    End If
611|    If nCols = 0 Then
612|      nCols = 1
613|    End If
614|    nCols = nCols \ nRows
615|    Current.SelEndCol = Current.SelStartCol + nCols - 1
616|    Current.SelEndRow = Current.SelStartRow + nRows - 1
617|    Current.Clip = ClipText
618| End If
619|
620| End Sub
621|
622| Sub meUndoThisRow_Click ()
623|  If grdAttributes.Row <= nRecs Then
624|    flgIgnoreMovement = True
625|    Data1.Recordset.Bookmark = strBookMark(grdAttributes.Row)
626|    For nF = 1 To nFields
627|   grdAttributes.Col = nF
628|   If FieldType(nF) < 11 Then grdAttributes = Format(grdAttributes)
629|    Next nF
630|    grdAttributes.Col = 0
631|    grdAttributes = grdAttributes.Row
632|    flgIgnoreMovement = False
633|   Else
634|    grdAttributes.RemoveItem grdAttributes.Row
635|  End If
```

```
End Sub

Sub mfAbandonTableChanges_Click ()
 flgIgnoreMovement = True

 grdAttributes.Col = 0
 For nR = 1 To grdAttributes.Rows - 1
  grdAttributes = nR
 Next nR
 grdAttributes.Col = lngCurrentCol
 flgIgnoreMovement = False
 If lngCurrentRow < grdAttributes.Rows Then grdAttributes.Row = lngCurrentRow
End Sub

Sub mfCloseDB_Click ()
 DB.Close
 Set DB = Nothing
  Data1.Recordset.Close
  cmbQTList.Clear

End Sub

Sub mfExit_Click ()
 ShutDownAndFreeMem
End Sub

Sub mfNewFile_Click ()
Static OutFile As String, OutExt As String, DelimiterChosen As String

  'Decide whether it should be Space, tab, or comma delimited.
   ReDim OZQuestion(1 To 5)  As typOZQuestion
   Dim Accepted As Integer
   OZQuestion(1).KindOfControl = CONTROL_LABEL
   OZQuestion(1).StringData = "What character do you want separating columns of
data?"
   OZQuestion(2).KindOfControl = CONTROL_OPTIONBUTTON
   OZQuestion(2).StringData = "&Tab"
   OZQuestion(3).KindOfControl = CONTROL_OPTIONBUTTON
   OZQuestion(3).StringData = "&, Comma"
   OZQuestion(4).KindOfControl = CONTROL_OPTIONBUTTON
   OZQuestion(4).StringData = "&5 spaces"
   OZQuestion(5).KindOfControl = CONTROL_OPTIONBUTTON
   OZQuestion(5).StringData = "&; Semi-colon"
'Consult the Wizard.
   AskOZ OZQuestion(), Accepted
   If Accepted Then
      If OZQuestion(2).Extra Then DelimiterChosen = Chr$(9): Me.CMDialog1.DefaultExt
= "TDF"
      If OZQuestion(3).Extra Then DelimiterChosen = ",": : Me.CMDialog1.DefaultExt =
"CSV"
      If OZQuestion(4).Extra Then DelimiterChosen = "     ": Me.CMDialog1.DefaultExt =
"SDF"
      If OZQuestion(5).Extra Then DelimiterChosen = ";": Me.CMDialog1.DefaultExt =
"TXT"
```

```
685|       Else
686|        Exit Sub
687|     End If
688|
689|    'Open a file
690|
691|  Me.CMDialog1.DialogTitle = "Text file to receive output of current grid"
692|  Me.CMDialog1.Filter = "Text|*.txt;*.tdf;*.sdf|Comma Separated Values
     (*.csv)|*.csv|Tab-delimted file(*.tdf)|*.tdf|All files (*.*)|(*.*)"
693|  Me.CMDialog1.Filename = OutFile
694|  Me.CMDialog1.Flags = OFN_OVERWRITEPROMPT Or OFN_PATHMUSTEXIST
695|
696|  Me.CMDialog1.Action = 1
697|  Dim Temp As String
698|
699|  OutFile = UCase$(Me.CMDialog1.Filename)
700|  If OutFile = "" Then Exit Sub
701|  Dim FileNum As Integer
702|  FileNum = FreeFile
703|  Open OutFile For Output As FileNum
704|
705|  'Write it out.
706|  Dim LineOut As String, Qt As String * 1, nR As Long, nC As Long, nRows As Long,
     nCols As Long
707|  nCols = grdAttributes.Cols
708|  nRows = grdAttributes.Rows
709|
710|  Qt = Chr$(34)
711|  flgIgnoreMovement = True
712|   grdAttributes.Row = 0
713|  'For first line, add a SerialNumber label and put quotes around everything
714|   LineOut = Qt & "SerialNumber" & Qt & DelimiterChosen
715|      For nC = 1 To nCols - 1
716|     grdAttributes.Col = nC
717|     LineOut = LineOut & Qt & grdAttributes & Qt & DelimiterChosen
718|      Next nC
719|   LineOut = Left(LineOut, Len(LineOut) - Len(DelimiterChosen))'truncate extra delimiter at
     end  of line
720|   Print #FileNum, LineOut
721|  For nR = 1 To grdAttributes.Rows - 1
722|      grdAttributes.Row = nR
723|  'No quotes around values here.
724|      LineOut = nR & DelimiterChosen
725|      For nC = 1 To nCols - 1
726|        grdAttributes.Col = nC
727|        LineOut = LineOut & grdAttributes & DelimiterChosen
728|      Next nC
729|  LineOut = Left(LineOut, Len(LineOut) - Len(DelimiterChosen))'truncate extra delimiter at
     end  of line
730|  Print #FileNum, LineOut
731|  Next nR
732|  flgIgnoreMovement = False
733|
734|  'Close the file
```

```
735|
736| Close FileNum
737|
738| End Sub
739|
740| Sub mfNewTable_Click ()
741|
742|   Dim NewTable As New TableDef
743|    Dim TempVar As Variant
744|   Dim n As Long, nCols As Long, m As Long, nRows As Long, TableName As String
745|   Dim nC As Long, nR As Long
746|   nCols = grdAttributes.Cols
747| On Error GoTo ERRmfNewTable
748| TableName = "User Points " & Format$(MapDB.TableDefs.Count, "0000")
749| TableName = InputBox$("Enter a name for the table to be saved as.", "Saving Grid to
     Table", TableName)
750|
751| If TableName = "" Then
752|    ' Erase NewField
753|     Exit Sub
754| End If
755|   NewTable.Name = TableName
756| 'Add the serial number field.
757|   Dim LatestField As New field
758|   LatestField.Type = DB_LONG
759|   LatestField.Name = "Serial Number"
760|   NewTable.Fields.Append LatestField
761|   MapDB.TableDefs.Append NewTable
762|   Set LatestField = Nothing
763|   Set NewTable = Nothing
764|
765| 'Get the newly created table
766|  Dim tblOut As Table
767|
768|
769| 'This will take the first row of the grdAttributes grid,
770| ' and use it to define the names and types for an output table.
771| 'No provision is made for being able to append to a table
772| '   or to overwrite a table.
773| 'Command decision by Greg Pouch, 22 June 1994
774|
775|   'ReDim NewField(0 To nCols) As New Field
776|   ReDim FieldNames(0 To nCols) As String
777|
778| Dim Temp As String, tmpFieldName As String, tmpFieldType As Integer
779| Dim strFieldTypeList As String
780| flgIgnoreMovement = True
781| grdAttributes.Row = 0
782|
783|
784| For n = 1 To grdAttributes.Cols - 1
785|    grdAttributes.Col = n
786|    Temp = grdAttributes
787|    If Temp = "" Then
```

```
788|    Temp = InputBox$("Please enter a name for this field, in column " & n, "Saving Table",
       "Field" & Format$(n, "00"))
789|     End If
790|     tmpFieldName = Left(Temp, Len(Temp) - 3)
791|     tmpFieldType = Val(Right(Temp, 3))
792|     If tmpFieldType = 0 Then
793|    'We have an untagged field, so ask the user whether
794|    'to skip it or to define the data type.
795|     ReDim OZQuestion(1 To 2)  As typOZQuestion
796|     Dim Accepted As Integer
797|     OZQuestion(1).KindOfControl = CONTROL_LABEL
798|     OZQuestion(1).StringData = "This field, " & Temp & ", has no data type associated it
      with it." & CRLF & "Choose one from the list box."
799|     OZQuestion(2).KindOfControl = CONTROL_COMBOBOX
800|     strFieldTypeList = ListOfFieldTypes
801|     OZQuestion(2).StringData = strFieldTypeList
802|     AskOZ OZQuestion(), Accepted
803|     If Accepted Then
804|       tmpFieldType = OZQuestion(2).Extra
805|     End If
806|      End If
807|
808|   If tmpFieldType <> 0 And tmpFieldName <> "" Then
809|     'if the above information exists, add the field
810|     Dim VeryNewField As New field
811|    VeryNewField.Type = tmpFieldType
812|    If tmpFieldType = 10 Then VeryNewField.Size = 255
813|    VeryNewField.Name = Trim$(tmpFieldName)
814|    MapDB.TableDefs(TableName).Fields.Append VeryNewField
815|    Set VeryNewField = Nothing
816|    FieldNames(n) = tmpFieldName
817|      Else 'Otherwise, mark it to be skipped.
818|       FieldNames(n) = ""
819|   End If
820|  Next n
821|  'At this point, all the columns have been assigned a field name,
822|  ' or not if the user doesn't want to keep it, and we should save the table
823|  ' and start writing to it.
824|   Set tblOut = MapDB.OpenTable(TableName)
825|   On Error Resume Next
826|  For nR = 1 To grdAttributes.Rows
827|     grdAttributes.Row = nR
828|     tblOut.AddNew
829|     tblOut("Serial Number") = nR
830|     For nC = 1 To nCols
831|       grdAttributes.Col = nC
832|       If (FieldNames(nC) <> "") Then
833|     TempVar = grdAttributes
834|     tblOut(FieldNames(nC)) = TempVar
835|       End If
836|     Next nC
837|     tblOut.Update
838|  Next nR
839|       Set tblOut = Nothing
```

```
840|
841|
842| Exit Sub
843| ERRmfNewTable:
844| Select Case Err
845|  Case 3010 'Table of that name already exists
846|    TableName = InputBox$("Table " & TableName & " already exists.  Please choose a
     new name.", "Saving Grid to Table", "User Table" & Format$(Now))
847|   If TableName = "" Then
848|      Err = 0
849|      Exit Sub
850|      Resume Next
851|   End If
852|    NewTable.Name = TableName
853|    Resume
854|  Case Else
855|    MsgBox "Error #" & Err & " occured in frmGridEdit.mfNewTable." & CRLF & Error
856|    Err = 0
857|    Exit Sub
858|    Resume
859| End Select
860|
861|
862| End Sub
863|
864| Sub mfOpenDB_Click ()
865| Dim DBName As String' May want to change scope of this later to global.
866| CMDialog1.DialogTitle = "Database to Open"
867| CMDialog1.Filter = "MS Access Databases|*.mdb"
868| CMDialog1.Flags = &H1000&    'OFN_FILEMUSTEXIST
869| CMDialog1.Action = 1
870| Dim Temp As String
871|
872| Temp = CMDialog1.Filename
873| If Len(Temp) = 0 Then 'the user did not enter a file name
874|    Else
875|     DBName = Temp
876|     ' Open an MSAccess database for non-exclusive, read-write access.
877|     Set DB = OpenDatabase(DBName, False, False, "") 'Open the desired database
878|     Data1.DatabaseName = DBName
879|     Me.Caption = "Grid-style data editor " & DBName
880|     Call ListQrysTblsInComboBox(DB, cmbQTList)
881|
882|     cmbQTList.SetFocus
883|     SendKeys "%{DOWN}"
884| End If
885|
886| End Sub
887|
888| Sub mfSaveTableChanges_Click ()
889|    SaveGridToDynaset
890| End Sub
891|
892| Sub SaveGridToDynaset ()
```

```
893| On Error GoTo ERRSaveAttributes
894| If Not flgChangesWereMade Then Exit Sub
895| flgIgnoreMovement = True
896|
897| 'Might want to change this so that it doesn't reset row number
898| 'This way, multiple records could be displayed and edited.
899| Dim nRows As Long, nRec As Long, Col0 As String
900| nRows = grdAttributes.Rows - 1
901| For nRec = 1 To nRows
902|    grdAttributes.Row = nRec
903| 'Examine the zeroeth column, which could contain a flag for NEW, EDITED, DELETED,
    etc.
904|    grdAttributes.Col = 0
905|    Col0 = grdAttributes
906|   If Col0 = "E" Then
907|    Data1.Recordset.Bookmark = strBookMark(nRec)
908|    Data1.Recordset.Edit
909|    For nF = 1 To nFields
910|   If FieldType(nF) < 11 Then 'don't update memo and BLOB objects, because the grid
    control can lose them due to chr(0)s
911|      grdAttributes.Col = nF
912|      Data1.Recordset.Fields(nF - 1) = grdAttributes
913|   End If
914|    Next nF
915|    Data1.Recordset.Update
916|   End If
917|
918|   If Col0 = "D" Then
919|     grdAttributes = "XXX"
920|     Data1.Recordset.Bookmark = strBookMark(nRec)
921|     Data1.Recordset.Delete
922|   End If
923|
924|   If Col0 = "A" Then
925|    grdAttributes = ""
926|    Data1.Recordset.AddNew
927|    For nF = 1 To nFields
928|   If FieldType(nF) < 11 Then 'don't update memo and BLOB objects, because the grid
    control can lose them due to chr(0)s
929|      grdAttributes.Col = nF
930|      Data1.Recordset.Fields(nF - 1) = grdAttributes
931|   End If
932|    Next nF
933|    Data1.Recordset.Update
934|   End If
935|
936| Next nRec
937|
938|
939| flgIgnoreMovement = False
940| flgChangesWereMade = False
941|
942| Exit Sub
943| ERRSaveAttributes:
```

```
944|
945| If Err = 3021 Then 'No current record
946|    Data1.Recordset.AddNew
947|    Resume Next
948| End If
949| If Err = 91 Then 'Object variable not set.  ie, no XYBLOB to update
950|  Resume Next
951| End If
952| If Err = 3058 Then
953|   grdAttributes.SelStartRow = grdAttributes.Row
954|   grdAttributes.SelEndRow = grdAttributes.Row
955|   grdAttributes.SelStartCol = 1
956|   grdAttributes.SelEndRow = grdAttributes.Cols - 1
957|   n = MsgBox("You have not entered a value in a field that is part of a primary index." &
    CRLF & "This is not allowed.  Do you want to delete this record?", 36)
958|   If n = 6 Then
959|     grdAttributes.Col = 0
960|     grdAttributes = "XXX"
961|     Resume Next
962|    Else
963|     Err = 0
964|     Exit Sub
965|     Resume Next
966|   End If
967| End If
968| If Err = 9 Then 'Subscript out of range
969|    Resume Next 'Try ignoring it
970| End If
971|
972|
973|
974|
975| MsgBox "Error number " & Err & " encountered while saving attributes." & Chr$(13) &
    Chr$(10) & Error$
976| Err = 0
977| Exit Sub
978| Resume
979|
980| End Sub
981|
982| Sub SetIgnoreMovement (ToValue As Long)
983|  flgIgnoreMovement = ToValue
984| End Sub
985|
986| Sub SetQuery (strTblQrySQL As String)
987| 'This subroutine sets the data control to the desired TaBLe, QueRY, or SQL string,
988| ' then sets the labels for the attribute grid.
989| On Error GoTo ERRSetQuery
990|
991| 'Check whether changes need to be saved.
992| If flgChangesWereMade Then
993|     'Present a yes/no/cancel box to the user.
994|   Me.Show
995|   nF = MsgBox("Changes were made to this data set." & CRLF & "Do you wish to save
```

```
      them now?", 36)
 996|    If nF = 2 Then Exit Sub'user chose cancel, so clear out.
 997|    If nF = 6 Then 'User chose to save changes
 998|      Call mfSaveTableChanges_Click
 999|    End If
1000|    flgChangesWereMade = False  'If the user chose yes or no, reset the flag.
1001| End If
1002|
1003| 'Set flag to not display cell contents in text box.
1004| flgIgnoreMovement = True
1005|
1006| 'Clear the grid
1007| ClearGrid Me.grdAttributes
1008|
1009| grdAttributes.RowHeight(0) = 600
1010| 'Set the data control to the desired query, table, or SQL statement.
1011| Data1.RecordSource = strTblQrySQL
1012| Data1.Refresh
1013|
1014| If Data1.Recordset.Bookmarkable Then
1015|    'ReDim strBookMark(1 To nRecs)
1016|    flgBookmarkable = True
1017|    ZUnKillMenuItems
1018|  Else
1019|    mfSaveTableChanges.Enabled = False
1020|    MsgBox "You will not be able to save changes to this table or query because it cannot
      be bookmarked."
1021|    ZKillMenuItems
1022|    flgBookmarkable = False
1023| End If
1024|
1025|
1026| flgIgnoreMovement = False
1027| grdAttributes.Visible = True
1028|
1029| Exit Sub
1030|
1031| ERRSetQuery:
1032| If Err = 3061 Then
1033|    MsgBox "You tried selecting a parameter query, and this application doesn't do those.
      Sorry!"
1034|    Err = 0
1035|    Exit Sub
1036| End If
1037|
1038| MsgBox "Error number " & Err & " occurred in SetQuery." & CRLF & Error
1039| Err = 0
1040| 'Set cursor back to normal.
1041| mousepointer = 0
1042| grdAttributes.Visible = True
1043| Exit Sub
1044| Resume
1045| End Sub
1046|
```

```
Sub txtAttributes_GotFocus ()
 If chkOverstrike Then
   txtAttributes.SelStart = 0
   txtAttributes.SelLength = 32765
 End If

End Sub

Sub txtAttributes_KeyPress (KeyAscii As Integer)

 If KeyAscii = 13 Then  'For a return key, move focus to the
   KeyAscii = 0       ' grid, then save the changes to the grid.
   'If Not flgBookmarkable Then
   '   Beep
   '   Beep
   '   txtAttributes = grdAttributes
   '   Exit Sub
   'End If

   flgIgnoreMovement = True
   grdAttributes.Col = 0
   If grdAttributes <> "A" Then grdAttributes = "E"
   grdAttributes.Col = lngCurrentCol
   flgIgnoreMovement = False
   flgChangesWereMade = True
   grdAttributes.SetFocus
 End If
 If KeyAscii = 27 Then   'For an ESCape key, reset the text box to
   txtAttributes = grdAttributes    'the contents of the grid cell.
   KeyAscii = 0
 End If


End Sub

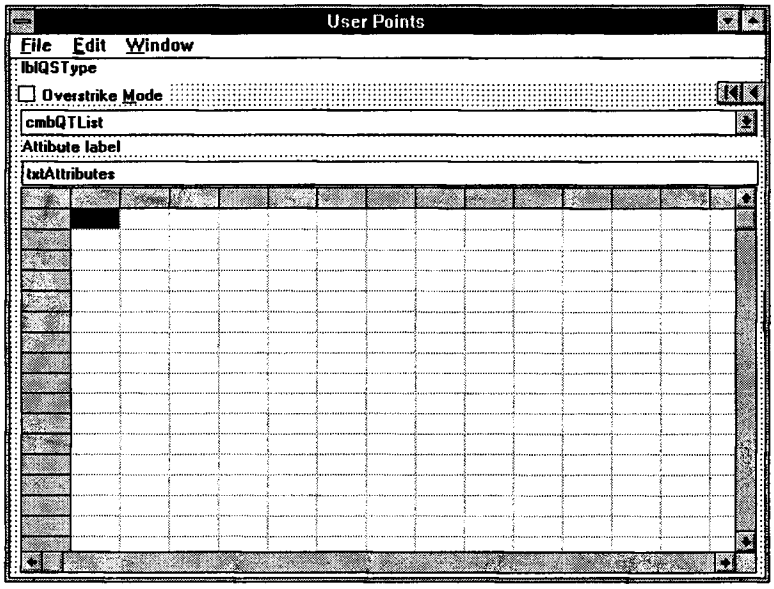
Sub txtAttributes_LostFocus ()
   grdAttributes.Text = txtAttributes
End Sub

Sub UndoDataChanges ()
 Call Data1_Reposition
End Sub

Sub ZKillMenuItems ()
  zzEdit.Enabled = False
  mfAbandonTableChanges.Enabled = False
  mfSaveTableChanges.Enabled = False
End Sub

Sub ZUnKillMenuItems ()
  zzEdit.Enabled = True
  mfAbandonTableChanges.Enabled = True
  mfSaveTableChanges.Enabled = True
End Sub
```

```
1101|
```


User Points
File Edit Window
lblQSType
Overstrike Mode
cmbQTList
Attibute label
txtAttributes

```
VERSION 2.00
Begin Form frmUserPoints
  Caption        = "User Points"
  ClientHeight   = 6060
  ClientLeft     = 495
  ClientTop      = 795
  ClientWidth    = 8745
  Height         = 6750
  Icon           = USERPTS1.FRX:0000
  Left           = 435
  LinkTopic      = "Form1"
  MDIChild       = -1 'True
  ScaleHeight    = 106.892
  ScaleMode      = 6 'Millimeter
  ScaleWidth     = 154.252
  Top            = 165
  Width          = 8865
  Begin ComboBox cmbQTList
    Height       = 300
    Left         = 120
    Sorted       = -1 'True
    Style        = 2 'Dropdown List
    TabIndex     = 6
    Top          = 600
    Visible      = 0 'False
    Width        = 8535
  End
  Begin CommonDialog CMDialog2
    Left         = 8730
    Top          = 450
  End
  Begin CommonDialog CMDialog1
    Left         = 8730
    Top          = 2700
  End
  Begin Data Data1
    Caption      = "Move record "
    Connect      = ""
    DatabaseName = ""
    Exclusive    = 0 'False
    Height       = 270
    Left         = 8190
    Options      = 0
    ReadOnly     = 0 'False
    RecordSource = ""
    Top          = 270
    Visible      = 0 'False
    Width        = 2445
  End
  Begin CheckBox chkOverstrike
    Caption      = "Overstrike &Mode"
    Height       = 255
    Left         = 90
    TabIndex     = 3
```

```
    Top             =  285
    Width           =  1757
  End
  Begin TextBox txtAttributes
    Height          =  285
    Left            =  135
    TabIndex        =  1
    Text            =  "txtAttributes"
    Top             =  1200
    Width           =  8535
  End
  Begin Grid grdAttributes
    Cols            =  15
    Height          =  4515
    Left            =  120
    Rows            =  30
    TabIndex        =  0
    Top             =  1500
    Width           =  8535
  End
  Begin Label lblQSSource
    Caption         =  "lblQSSource"
    Height          =  255
    Left            =  120
    TabIndex        =  5
    Top             =  600
    Width           =  8415
  End
  Begin Label lblQSType
    Caption         =  "lblQSType"
    Height          =  255
    Left            =  120
    TabIndex        =  4
    Top             =  0
    Width           =  8535
  End
  Begin Label lblAttributes
    Caption         =  "Attibute label"
    Height          =  210
    Left            =  135
    TabIndex        =  2
    Top             =  930
    Width           =  8535
  End
  Begin Menu zzzzFile
    Caption         =  "&File"
    Begin Menu mfOpenDB
      Caption       =  "&Open Database..."
      Visible       =  0  'False
    End
    Begin Menu mfCloseDB
      Caption       =  "&Close Database"
      Visible       =  0  'False
    End
```

```
109|     Begin Menu zz10
110|       Caption      = "-"
111|     End
112|     Begin Menu mfNewTable
113|       Caption      = "New &Table..."
114|     End
115|     Begin Menu zzz1
116|       Caption      = "-"
117|     End
118|     Begin Menu mfNewFile
119|       Caption      = "New &File..."
120|     End
121|     Begin Menu zz3
122|       Caption      = "-"
123|       Visible      = 0  'False
124|     End
125|     Begin Menu mfSaveTableChanges
126|       Caption      = "&Save Changes"
127|       Visible      = 0  'False
128|     End
129|     Begin Menu mfAbandonTableChanges
130|       Caption      = "&Abandon Changes"
131|       Visible      = 0  'False
132|     End
133|     Begin Menu zzzzz1
134|       Caption      = "-"
135|     End
136|     Begin Menu mfExit
137|       Caption      = "E&xit"
138|     End
139|   End
140|   Begin Menu zzEdit
141|     Caption        = "&Edit"
142|     Begin Menu meUndoThisRow
143|       Caption      = "&Undo changes to this row"
144|       Shortcut     = %{BKSP}
145|     End
146|     Begin Menu zz6
147|       Caption      = "-"
148|     End
149|     Begin Menu meCopy
150|       Caption      = "&Copy"
151|       Shortcut     = ^{INSERT}
152|     End
153|     Begin Menu mePaste
154|       Caption      = "&Paste"
155|       Shortcut     = +{INSERT}
156|     End
157|     Begin Menu meCut
158|       Caption      = "Cu&t"
159|       Shortcut     = +{DEL}
160|     End
161|     Begin Menu meClearGrid
162|       Caption      = "C&lear"
```

```
163|       Shortcut      =  ^X
164|     End
165|     Begin Menu zzzz1
166|       Caption       =  "-"
167|     End
168|     Begin Menu meInsertRow
169|       Caption       =  "&Append Records"
170|       Shortcut      =  ^A
171|     End
172|     Begin Menu meDeleteRow
173|       Caption       =  "&Delete Records"
174|       Shortcut      =  ^D
175|     End
176|     Begin Menu zzz5
177|       Caption       =  "-"
178|     End
179|     Begin Menu meAddColumn
180|       Caption       =  "Add Co&lumn"
181|     End
182|     Begin Menu meColDefinition
183|       Caption       =  "Column De&finition"
184|     End
185|   End
186|   Begin Menu mnuWindow
187|     Caption       =  "&Window"
188|     WindowList    =  -1  'True
189|   End
190| End
191| Option Explicit
192|
193| Dim flgChangeRow As Integer, flgIgnoreMovement As Integer
194| Dim flgChangesWereMade As Integer, flgBookmarkable As Integer
195|
196| Dim strBookMark() As String
197| Dim flgSkipMe() As Long
198| Dim nFields As Long, n As Long, nRecs As Long, nR As Long
199| Dim nF As Long, FieldType() As Long
200| Dim lngCurrentRow As Long, lngCurrentCol As Long
201|
202| Dim DB As Database, tblLineData As Table
203| 'Dim snpQTList As Snapshot, dnsCurrentQT As dynaset
204|
205| Dim flgAddingRecord As Integer   'Might want to eliminate this variable.
206| Dim TabChar As String * 1, CR  As String * 1, CRLF  As String * 2
207| Dim ListOfFieldTypes As String
208|
209| Sub cmbQTList_Click ()
210| Dim Temp As String
211| mousepointer = 11
212|
213| Temp = cmbQTList
214| Call SetQuery(Temp)
215| flgIgnoreMovement = True
216| grdAttributes.FixedCols = 0: grdAttributes.Cols = 1
```

```
grdAttributes.Cols = 2: grdAttributes.FixedCols = 1
grdAttributes.Rows = 2
'Function GetClipBoardTextFromDynaset (dynIn As Dynaset, flgOptions As Long) As
String
Set dynViewInGrid = DB.CreateDynaset(Temp)
Call InitializeGridForDynaset(Me!grdAttributes, dynViewInGrid)

Temp = GetClipBoardTextFromDynaset(dynViewInGrid, 0)
Call SetGridFocusColumn(Me!grdAttributes, 1)
Call FillGridFromClipBoardText(Temp, Me!grdAttributes, 12&)

grdAttributes.RemoveItem 1


'Call OverwriteGridFromDynaset(Me!grdAttributes, dynViewInGrid)
flgIgnoreMovement = False
mousepointer = 0

End Sub

Sub cmdClearGrid_Click ()
 flgIgnoreMovement = True
  grdAttributes.FixedRows = 0
  grdAttributes.Rows = 1
  grdAttributes.Rows = 2
  grdAttributes.FixedRows = 1
 flgIgnoreMovement = False
End Sub

Sub Data1_Reposition ()
'On Error Resume Next
'DisplayCurrentRecord
End Sub

Sub DefineColumn ()
    flgIgnoreMovement = True
    Dim Temp As String, tmpFieldName As String, tmpFieldType As Integer,
strFieldTypeList As String
    grdAttributes.Row = 0
    n = grdAttributes.Col
    Temp = grdAttributes
    If Temp = "" Then
   tmpFieldName = "Field" & Format$(n, "00")
   tmpFieldType = 10'Default to text
      Else
   tmpFieldType = Val(Right(Temp, 3))
   If tmpFieldType <> 0 Then
      tmpFieldName = Left(Temp, Len(Temp) - 3)
     Else
      tmpFieldName = Temp
      tmpFieldType = 10
   End If
     End If

```

```
269|    ReDim OZQuestion(1 To 4)  As typOZQuestion
270|    Dim Accepted As Integer
271|    'Define the name.
272|    OZQuestion(1).KindOfControl = CONTROL_LABEL
273|    OZQuestion(1).StringData = "This field is currently named: " & Temp & " ." & CRLF &
       "If you wish, you may enter a new name below." '& CRLF & "and droning on and on and
       on about absolutely nothing at all in the world of any possible interest to anyone."
274|    OZQuestion(2).KindOfControl = CONTROL_TEXTBOX
275|    OZQuestion(2).StringData = tmpFieldName
276|
277|    'Define the datafield type.
278|    OZQuestion(3).KindOfControl = CONTROL_LABEL
279|    OZQuestion(3).StringData = "It currently has a datatype of " & tmpFieldType
280|    OZQuestion(4).KindOfControl = CONTROL_COMBOBOX
281|    strFieldTypeList = ListOfFieldTypes
282|    OZQuestion(4).StringData = strFieldTypeList
283|    OZQuestion(4).Extra = tmpFieldType
284|  'Consult the Wizard.
285|    AskOZ OZQuestion(), Accepted
286|
287|    If Accepted Then
288|       tmpFieldType = OZQuestion(4).Extra
289|       tmpFieldName = OZQuestion(2).StringData
290|       grdAttributes = tmpFieldName & " " & Format$(tmpFieldType, "00")
291|    End If
292|    flgIgnoreMovement = False
293|
294|  End Sub
295|
296|  Sub DisplayCurrentRecord ()
297|  On Error GoTo ERRDisplayCurrentRecord
298|  flgIgnoreMovement = True
299|  grdAttributes.Row = 1
300|  Dim Temp As String
301|  For n = 1 To nFields
302|     grdAttributes.Col = n
303|     grdAttributes = Format(Data1.Recordset.Fields(n - 1).Value)
304|  Next n
305|
306|
307|  flgIgnoreMovement = False
308|  Exit Sub
309|
310|  ERRDisplayCurrentRecord:
311|  If Err = 30010 Then 'Invalid number of columns or rows
312|     Err = 0
313|     Exit Sub
314|     Resume Next
315|  End If
316|  MsgBox "Error number " & Err & " encountered while displaying a record." & CRLF &
       Error$
317|  Err = 0
318|  Exit Sub
319|  Resume
```

```
320|
321|
322|
323| End Sub
324|
325| Sub Form_Load ()
326|
327| CR = Chr$(13)
328| TabChar = Chr$(9)
329| CRLF = Chr$(13) & Chr$(10)
330| ListOfFieldTypes = ""
331|
332| 'cmbTypeList.Clear
333| 'cmbTypeList.AddItem "Do not save this field", 0
334| 'cmbTypeList.AddItem "BOOLEAN", 1
335| 'cmbTypeList.AddItem "BYTE", 2
336| 'cmbTypeList.AddItem "Short Integer", 3
337| 'cmbTypeList.AddItem "Long Integer 4-bytes", 4
338| 'cmbTypeList.AddItem "CURRENCY", 5
339| 'cmbTypeList.AddItem "Single Precision REAL*4", 6
340| 'cmbTypeList.AddItem "Double Precision REAL*8", 7
341| 'cmbTypeList.AddItem "DATE", 8
342| 'cmbTypeList.AddItem "Do not Save this field", 9
343| 'cmbTypeList.AddItem "TEXT", 10
344| 'cmbTypeList.AddItem "LONGBINARY", 11
345| 'cmbTypeList.AddItem "MEMO (Long String)", 12
346|
347|
348| ListOfFieldTypes = "Do not save this field" & CRLF & "BOOLEAN" & CRLF & "BYTE" &
    CRLF
349| ListOfFieldTypes = ListOfFieldTypes & "Short Integer" & CRLF & "Long Integer 4-bytes"
    & CRLF
350| ListOfFieldTypes = ListOfFieldTypes & "CURRENCY" & CRLF & "Single Precision
    REAL*4" & CRLF & "Double Precision REAL*8" & CRLF
351| ListOfFieldTypes = ListOfFieldTypes & "DATE" & CRLF & "Do not Save this field" &
    CRLF & "TEXT" & CRLF & "LONGBINARY" & CRLF & "MEMO (Long String)" & CRLF
352|
353|
354| Me!grdAttributes.Cols = 6
355| flgIgnoreMovement = True
356|   Me!grdAttributes.Row = 0
357|
358|   Me!grdAttributes.Col = 1
359|   Me!grdAttributes = "Name 10"
360|   Me!grdAttributes.ColWidth(1) = 1200
361|
362|   Me!grdAttributes.Col = 2
363|   Me!grdAttributes = "X_Coord 06"
364|   Me!grdAttributes.ColWidth(2) = 1200
365|
366|   Me!grdAttributes.Col = 3
367|   Me!grdAttributes = "Y_Coord 06"
368|   Me!grdAttributes.ColWidth(3) = 1200
369|
```

```
370|
371|    Me!grdAttributes.Col = 4
372|    Me!grdAttributes = "TimeStamp 08"
373|    Me!grdAttributes.ColWidth(4) = 1800
374|
375|    Me!grdAttributes.Col = 5
376|    Me!grdAttributes = "SequenceNumber 08"
377|    Me!grdAttributes.ColWidth(5) = 1200
378|
379|    Me!grdAttributes.Rows = 2
380| flgIgnoreMovement = False
381|
382|
383| End Sub
384|
385| Sub Form_Paint ()
386| flgIgnoreMovement = False
387| End Sub
388|
389| Sub Form_QueryUnload (Cancel As Integer, UnloadMode As Integer)
390| 'Check whether changes need to be saved.
391|
392| If flgChangesWereMade Then
393|      'Present a yes/no/cancel box to the user.
394|   nF = MsgBox("Changes were made to this data set." & CRLF & "Do you wish to save
    them now?", 35)
395|   If nF = 2 Then
396|     Cancel = True
397|     Exit Sub'user chose cancel, so clear out.
398|     End If
399|   If nF = 6 Then 'User chose to save changes
400|     Call mfSaveTableChanges_Click
401|     If flgChangesWereMade Then Cancel = True 'Still need to cancel
402|   End If
403|   'flgChangesWereMade = False  'If the user chose yes or no, reset the flag.
404| End If
405|
406| End Sub
407|
408| Sub Form_Resize ()
409| If frmGridDataEdit.WindowState = 1 Then Exit Sub'iconized
410| On Error GoTo ERRResize
411|    lblAttributes.Width = frmGridDataEdit.ScaleWidth - 5
412|    txtAttributes.Width = frmGridDataEdit.ScaleWidth - 5
413|    grdAttributes.Width = frmGridDataEdit.ScaleWidth - 5
414|    grdAttributes.Height = frmGridDataEdit.ScaleHeight - 27.5
415|
416| cmbQTList.Width = Me.ScaleWidth - 10
417|
418| Exit Sub
419|
420| ERRResize:
421| If Err = 380 Then 'invalid width
422|   frmGridDataEdit.Width = frmGridDataEdit.Width * 1.2
```

```
423|   Resume
424|  End If
425|
426|
427|  End Sub
428|
429|  Sub GetIgnoreMovement (FromValue As Long)
430|   FromValue = flgIgnoreMovement
431|  End Sub
432|
433|  Sub grdAttributes_KeyDown (Keycode As Integer, Shift As Integer)
434|  If Keycode > 48 Then
435|     txtAttributes.SetFocus
436|  End If
437|  End Sub
438|
439|  Sub grdAttributes_KeyPress (KeyAscii As Integer)
440|     If KeyAscii = 13 Then SendKeys "{DOWN}"
441|     If KeyAscii > 31 Then SendKeys Chr$(KeyAscii)
442|  End Sub
443|
444|  Sub grdAttributes_RowColChange ()
445|   'every time a new cell is selected, the text appears
446|   ' in textbox text1 and text1 receives focus.
447|   'Updates of text1 automatically appear in the grid.
448|  If flgIgnoreMovement Then Exit Sub
449|
450|     flgIgnoreMovement = True
451|     lngCurrentRow = grdAttributes.Row
452|     lngCurrentCol = grdAttributes.Col
453|     txtAttributes = grdAttributes.Text
454|     grdAttributes.Row = 0
455|     lblAttributes.Caption = grdAttributes.Text
456|     grdAttributes.Row = lngCurrentRow
457|     flgIgnoreMovement = False
458|
459|  End Sub
460|
461|  Sub GridClipNRowsNCols (ClipText As String, nRows As Long, nCols As Long)
462|     nCols = 1
463|     nRows = 1
464|     Dim TabChar As String * 1, CR  As String * 1
465|     CR = Chr$(13)
466|     TabChar = Chr$(9)
467|     Dim n As Long
468|  ' Find number of rows (delimeted by asc(13))
469|     For n = 1 To Len(ClipText)
470|    If Mid$(ClipText, n, 1) = CR Then
471|        nRows = nRows + 1
472|        nCols = nCols + 1
473|    End If
474|    If Mid$(ClipText, n, 1) = TabChar Then
475|        nCols = nCols + 1
476|    End If
```

```
    Next n
    nCols = nCols \ nRows
End Sub

Sub lblQSSource_Change ()
If lblQSSource = "CLEAR" Then
    grdAttributes.FixedRows = 1
    flgIgnoreMovement = True
    ClearGrid Me.grdAttributes
    flgIgnoreMovement = False
   Else
    SetQuery CStr(lblQSSource)
End If
End Sub

Sub lblQSType_Change ()
If flgChangesWereMade Then
     'Present a yes/no box to the user.
   Me.Show
   nF = MsgBox("Changes were made to this data set." & CRLF & "Do you wish to save
them now?", 36)
   If nF = 6 Then 'User chose to save changes
    Call mfSaveTableChanges_Click
    Else
    flgChangesWereMade = False
   End If
End If

End Sub

Sub ListQrysTblsInComboBox (DBIn As Database, ComboOut As ComboBox)
' this subroutine lists all queries and tables in a database
'  in a combobox, for later user selection.
    ComboOut.Clear
    Dim DB_SYSTEMOBJECT As Long
    DB_SYSTEMOBJECT = &H80000002

    Dim snpQTList As Snapshot 'declared locally so it goes away
    Set snpQTList = DBIn.ListTables()

    Dim QTName As String
    snpQTList.MoveFirst
    Do Until snpQTList.EOF 'For each Table and query
   If Not ((snpQTList("Attributes") And DB_SYSTEMOBJECT) <> 0) And
(snpQTList("Attributes") < 6) Then
     'Exclude system objects from list of tables and queries presented to user
     QTName = snpQTList("Name")
     ComboOut.AddItem QTName
   End If
   snpQTList.MoveNext
    Loop
    snpQTList.Close
    Set snpQTList = Nothing
End Sub
```

```
529|
530|  Sub meAddColumn_Click ()
531|   grdAttributes.Cols = grdAttributes.Cols + 1
532|  End Sub
533|
534|  Sub meClearGrid_Click ()
535|   flgIgnoreMovement = True
536|    grdAttributes.FixedRows = 0
537|    grdAttributes.Rows = 1
538|    grdAttributes.Rows = 2
539|    grdAttributes.FixedRows = 1
540|   flgIgnoreMovement = False
541|
542|  End Sub
543|
544|  Sub meColDefinition_Click ()
545|    DefineColumn
546|  End Sub
547|
548|  Sub meCopy_Click ()
549|  Dim Current As Control
550|  Set Current = ActiveControl
551|  If TypeOf Current Is TextBox Then
552|     clipboard.Clear
553|     clipboard.SetText Current
554|  End If
555|  If TypeOf Current Is Grid Then
556|     Dim ClipText As String
557|     ClipText = Current.Clip
558|     clipboard.Clear
559|     clipboard.SetText ClipText
560|  End If
561|  End Sub
562|
563|  Sub meDeleteRow_Click ()
564|
565|     Dim n As Integer, nRows As Integer
566|
567|     flgIgnoreMovement = True
568|    grdAttributes.Col = 0
569|     For n = grdAttributes.SelStartRow To grdAttributes.SelEndRow
570|    grdAttributes.Row = n
571|    If grdAttributes <> "XXX" Then grdAttributes = "D"
572|     Next n
573|     flgChangesWereMade = True
574|     flgIgnoreMovement = False
575|  End Sub
576|
577|  Sub meInsertRow_Click ()
578|
579|
580|    Dim n As Long
581|    flgIgnoreMovement = True
582|    For n = grdAttributes.SelStartRow To grdAttributes.SelEndRow
```

```
583|      grdAttributes.AddItem "A"   ', (ActiveControl.Row) 'Chr$(9) & Chr$(9) & CR
584|    Next n
585|    flgChangesWereMade = True
586|    flgIgnoreMovement = False
587|  End Sub
588|
589|  Sub mePaste_Click ()
590|  Dim Current As Control
591|  Set Current = ActiveControl
592|  Dim ClipText As String
593|  ClipText = clipboard.GetText()
594|
595|  If TypeOf Current Is TextBox Then
596|     Dim Temp As String
597|     Temp = Current.Text
598|     'Check these XXXX use frmTileOptions as example
599|     Temp = Mid$(Temp, 1, Current.SelStart) & ClipText & Mid$(Temp, Current.SelStart +
     Current.SelLength)
600|     Current.Text = Temp
601|  End If
602|
603|  If TypeOf Current Is Grid Then
604|     Dim nRows As Long, nCols As Long
605|     Dim n As Long
606|  ' Find number of rows (delimeted by asc(13))
607|     For n = 1 To Len(ClipText)
608|    If Mid$(ClipText, n, 1) = CR Then
609|       nRows = nRows + 1
610|    End If
611|    If Mid$(ClipText, n, 1) = TabChar Then
612|       nCols = nCols + 1
613|    End If
614|     Next n
615|
616|     If nRows = 0 Then
617|       nRows = 1
618|     End If
619|     If nCols = 0 Then
620|       nCols = 1
621|     End If
622|     nCols = nCols \ nRows
623|     Current.SelEndCol = Current.SelStartCol + nCols - 1
624|     Current.SelEndRow = Current.SelStartRow + nRows - 1
625|     Current.Clip = ClipText
626|  End If
627|
628|  End Sub
629|
630|  Sub meUndoThisRow_Click ()
631|   If grdAttributes.Row <= nRecs Then
632|     flgIgnoreMovement = True
633|     Data1.Recordset.Bookmark = strBookMark(grdAttributes.Row)
634|     For nF = 1 To nFields
635|    grdAttributes.Col = nF
```

```
636|    If FieldType(nF) < 11 Then grdAttributes = Format(grdAttributes)
637|     Next nF
638|     grdAttributes.Col = 0
639|     grdAttributes = grdAttributes.Row
640|     flgIgnoreMovement = False
641|    Else
642|     grdAttributes.RemoveItem grdAttributes.Row
643|   End If
644|  End Sub
645|
646|  Sub mfAbandonTableChanges_Click ()
647|   flgIgnoreMovement = True
648|
649|   grdAttributes.Col = 0
650|   For nR = 1 To grdAttributes.Rows - 1
651|    grdAttributes = nR
652|   Next nR
653|   grdAttributes.Col = lngCurrentCol
654|   flgIgnoreMovement = False
655|   If lngCurrentRow < grdAttributes.Rows Then grdAttributes.Row = lngCurrentRow
656|  End Sub
657|
658|  Sub mfCloseDB_Click ()
659|   DB.Close
660|   Set DB = Nothing
661|   Data1.Recordset.Close
662|   cmbQTList.Clear
663|
664|  End Sub
665|
666|  Sub mfExit_Click ()
667|   ShutDownAndFreeMem
668|  End Sub
669|
670|  Sub mfNewFile_Click ()
671|  Static OutFile As String, OutExt As String, DelimiterChosen As String
672|
673|   'Decide whether it should be Space, tab, or comma delimited.
674|    ReDim OZQuestion(1 To 5)  As typOZQuestion
675|    Dim Accepted As Integer
676|    OZQuestion(1).KindOfControl = CONTROL_LABEL
677|    OZQuestion(1).StringData = "What character do you want separating columns of
      data?"
678|    OZQuestion(2).KindOfControl = CONTROL_OPTIONBUTTON
679|    OZQuestion(2).StringData = "&Tab"
680|    OZQuestion(3).KindOfControl = CONTROL_OPTIONBUTTON
681|    OZQuestion(3).StringData = "&, Comma"
682|    OZQuestion(4).KindOfControl = CONTROL_OPTIONBUTTON
683|    OZQuestion(4).StringData = "&5 spaces"
684|    OZQuestion(5).KindOfControl = CONTROL_OPTIONBUTTON
685|    OZQuestion(5).StringData = "&; Semi-colon"
686|  'Consult the Wizard.
687|    AskOZ OZQuestion(), Accepted
688|    If Accepted Then
```

```
689|       If OZQuestion(2).Extra Then DelimiterChosen = Chr$(9): Me.CMDialog1.DefaultExt
   = "TDF"
690|       If OZQuestion(3).Extra Then DelimiterChosen = ",": : Me.CMDialog1.DefaultExt =
   "CSV"
691|       If OZQuestion(4).Extra Then DelimiterChosen = "     ": Me.CMDialog1.DefaultExt =
   "SDF"
692|       If OZQuestion(5).Extra Then DelimiterChosen = ";": Me.CMDialog1.DefaultExt =
   "TXT"
693|      Else
694|       Exit Sub
695|    End If
696|
697|   'Open a file
698|
699|  Me.CMDialog1.DialogTitle = "Text file to receive output of current grid"
700|  Me.CMDialog1.Filter = "Text|*.txt;*.tdf;*.sdf|Comma Separated Values
   (*.csv)|*.csv|Tab-delimted file(*.tdf)|*.tdf|All files (*.*)|(*.*)"
701|  Me.CMDialog1.Filename = OutFile
702|  Me.CMDialog1.Flags = OFN_OVERWRITEPROMPT Or OFN_PATHMUSTEXIST
703|
704|  Me.CMDialog1.Action = 1
705|  Dim Temp As String
706|
707|  OutFile = UCase$(Me.CMDialog1.Filename)
708|  If OutFile = "" Then Exit Sub
709|  Dim FileNum As Integer
710|  FileNum = FreeFile
711|  Open OutFile For Output As FileNum
712|
713|  'Write it out.
714|  Dim LineOut As String, Qt As String * 1, nR As Long, nC As Long, nRows As Long,
   nCols As Long
715|  nCols = grdAttributes.Cols
716|  nRows = grdAttributes.Rows
717|
718|  Qt = Chr$(34)
719|  flgIgnoreMovement = True
720|   grdAttributes.Row = 0
721|  'For first line, add a SerialNumber label and put quotes around everything
722|   LineOut = Qt & "SerialNumber" & Qt & DelimiterChosen
723|     For nC = 1 To nCols - 1
724|    grdAttributes.Col = nC
725|    LineOut = LineOut & Qt & grdAttributes & Qt & DelimiterChosen
726|     Next nC
727|   LineOut = Left(LineOut, Len(LineOut) - Len(DelimiterChosen))'truncate extra delimiter at
   end  of line
728|   Print #FileNum, LineOut
729|  For nR = 1 To grdAttributes.Rows - 1
730|     grdAttributes.Row = nR
731|  'No quotes around values here.
732|     LineOut = nR & DelimiterChosen
733|     For nC = 1 To nCols - 1
734|       grdAttributes.Col = nC
735|       LineOut = LineOut & grdAttributes & DelimiterChosen
```

```
736|     Next nC
737| LineOut = Left(LineOut, Len(LineOut) - Len(DelimiterChosen))'truncate extra delimiter at
     end  of line
738| Print #FileNum, LineOut
739| Next nR
740| flgIgnoreMovement = False
741|
742| 'Close the file
743|
744| Close FileNum
745|
746| End Sub
747|
748| Sub mfNewTable_Click ()
749|
750|   Dim NewTable As New TableDef
751|    Dim TempVar As Variant
752|   Dim n As Long, nCols As Long, m As Long, nRows As Long, TableName As String
753|   Dim nC As Long, nR As Long
754|   nCols = grdAttributes.Cols
755| On Error GoTo ERRmfNewTable
756| TableName = "User Table" & Format$(MapDB.TableDefs.Count, "0000")
757| TableName = InputBox$("Enter a name for the table to be saved as.", "Saving Grid to
     Table", TableName)
758|
759| If TableName = "" Then
760|    ' Erase NewField
761|    Exit Sub
762| End If
763|   NewTable.Name = TableName
764| 'Add the serial number field.
765|   Dim LatestField As New field
766|   LatestField.Type = DB_LONG
767|   LatestField.Name = "Serial Number"
768|   NewTable.Fields.Append LatestField
769|   MapDB.TableDefs.Append NewTable
770|   Set LatestField = Nothing
771|   Set NewTable = Nothing
772|
773| 'Get the newly created table
774|  Dim tblOut As Table
775|
776|
777| 'This will take the first row of the grdAttributes grid,
778| ' and use it to define the names and types for an output table.
779| 'No provision is made for being able to append to a table
780| '  or to overwrite a table.
781| 'Command decision by Greg Pouch, 22 June 1994
782|
783|   'ReDim NewField(0 To nCols) As New Field
784|   ReDim FieldNames(0 To nCols) As String
785|
786| Dim Temp As String, tmpFieldName As String, tmpFieldType As Integer
787| Dim strFieldTypeList As String
```

```
788| flgIgnoreMovement = True
789| grdAttributes.Row = 0
790|
791|
792| For n = 1 To grdAttributes.Cols - 1
793|    grdAttributes.Col = n
794|    Temp = grdAttributes
795|    If Temp = "" Then
796|   Temp = InputBox$("Please enter a name for this field, in column " & n, "Saving Table",
    "Field" & Format$(n, "00"))
797|    End If
798|    tmpFieldName = Left(Temp, Len(Temp) - 3)
799|    tmpFieldType = Val(Right(Temp, 3))
800|    If tmpFieldType = 0 Then
801|   'We have an untagged field, so ask the user whether
802|   'to skip it or to define the data type.
803|   ReDim OZQuestion(1 To 2)  As typOZQuestion
804|   Dim Accepted As Integer
805|   OZQuestion(1).KindOfControl = CONTROL_LABEL
806|   OZQuestion(1).StringData = "This field, " & Temp & ", has no data type associated it
    with it." & CRLF & "Choose one from the list box."
807|   OZQuestion(2).KindOfControl = CONTROL_COMBOBOX
808|   strFieldTypeList = ListOfFieldTypes
809|   OZQuestion(2).StringData = strFieldTypeList
810|   AskOZ OZQuestion(), Accepted
811|   If Accepted Then
812|     tmpFieldType = OZQuestion(2).Extra
813|   End If
814|    End If
815|
816|  If tmpFieldType <> 0 And tmpFieldName <> "" Then
817|    'if the above information exists, add the field
818|    Dim VeryNewField As New field
819|   VeryNewField.Type = tmpFieldType
820|   If tmpFieldType = 10 Then VeryNewField.Size = 255
821|   VeryNewField.Name = Trim$(tmpFieldName)
822|   MapDB.TableDefs(TableName).Fields.Append VeryNewField
823|   Set VeryNewField = Nothing
824|   FieldNames(n) = tmpFieldName
825|     Else 'Otherwise, mark it to be skipped.
826|      FieldNames(n) = ""
827|  End If
828| Next n
829| 'At this point, all the columns have been assigned a field name,
830| ' or not if the user doesn't want to keep it, and we should save the table
831| ' and start writing to it.
832|  Set tblOut = MapDB.OpenTable(TableName)
833|  On Error Resume Next
834| For nR = 1 To grdAttributes.Rows
835|    grdAttributes.Row = nR
836|    tblOut.AddNew
837|    tblOut("Serial Number") = nR
838|    For nC = 1 To nCols
839|      grdAttributes.Col = nC
```

```
840|       If (FieldNames(nC) <> "") Then
841|    TempVar = grdAttributes
842|    tblOut(FieldNames(nC)) = TempVar
843|       End If
844|    Next nC
845|    tblOut.Update
846| Next nR
847|      Set tblOut = Nothing
848|
849|
850| Exit Sub
851| ERRmfNewTable:
852| Select Case Err
853|  Case 3010 'Table of that name already exists
854|    TableName = InputBox$("Table " & TableName & " already exists.  Please choose a
     new name.", "Saving Grid to Table", "User Table" & Format$(Now))
855|   If TableName = "" Then
856|      Err = 0
857|      Exit Sub
858|      Resume Next
859|   End If
860|    NewTable.Name = TableName
861|    Resume
862|  Case Else
863|    MsgBox "Error #" & Err & " occured in frmGridEdit.mfNewTable." & CRLF & Error
864|    Err = 0
865|    Exit Sub
866|    Resume
867| End Select
868|
869|
870| End Sub
871|
872| Sub mfOpenDB_Click ()
873| Dim DBName As String' May want to change scope of this later to global.
874| CMDialog1.DialogTitle = "Database to Open"
875| CMDialog1.Filter = "MS Access Databases|*.mdb"
876| CMDialog1.Flags = &H1000&    'OFN_FILEMUSTEXIST
877| CMDialog1.Action = 1
878| Dim Temp As String
879|
880| Temp = CMDialog1.Filename
881| If Len(Temp) = 0 Then 'the user did not enter a file name
882|    Else
883|     DBName = Temp
884|     ' Open an MSAccess database for non-exclusive, read-write access.
885|     Set DB = OpenDatabase(DBName, False, False, "") 'Open the desired database
886|     Data1.DatabaseName = DBName
887|     Me.Caption = "Grid-style data editor " & DBName
888|     Call ListQrysTblsInComboBox(DB, cmbQTList)
889|
890|     cmbQTList.SetFocus
891|     SendKeys "%{DOWN}"
892| End If
```

```
893|
894| End Sub
895|
896| Sub mfSaveTableChanges_Click ()
897|    SaveGridToDynaset
898| End Sub
899|
900| Sub SaveGridToDynaset ()
901| On Error GoTo ERRSaveAttributes
902| If Not flgChangesWereMade Then Exit Sub
903| flgIgnoreMovement = True
904|
905| 'Might want to change this so that it doesn't reset row number
906| 'This way, multiple records could be displayed and edited.
907| Dim nRows As Long, nRec As Long, Col0 As String
908| nRows = grdAttributes.Rows - 1
909| For nRec = 1 To nRows
910|    grdAttributes.Row = nRec
911| 'Examine the zeroeth column, which could contain a flag for NEW, EDITED, DELETED,
    etc.
912|    grdAttributes.Col = 0
913|    Col0 = grdAttributes
914|   If Col0 = "E" Then
915|    Data1.Recordset.Bookmark = strBookMark(nRec)
916|    Data1.Recordset.Edit
917|    For nF = 1 To nFields
918|   If FieldType(nF) < 11 Then 'don't update memo and BLOB objects, because the grid
    control can lose them due to chr(0)s
919|      grdAttributes.Col = nF
920|      Data1.Recordset.Fields(nF - 1) = grdAttributes
921|   End If
922|    Next nF
923|    Data1.Recordset.Update
924|   End If
925|
926|   If Col0 = "D" Then
927|     grdAttributes = "XXX"
928|     Data1.Recordset.Bookmark = strBookMark(nRec)
929|     Data1.Recordset.Delete
930|   End If
931|
932|   If Col0 = "A" Then
933|    grdAttributes = ""
934|    Data1.Recordset.AddNew
935|    For nF = 1 To nFields
936|   If FieldType(nF) < 11 Then 'don't update memo and BLOB objects, because the grid
    control can lose them due to chr(0)s
937|      grdAttributes.Col = nF
938|      Data1.Recordset.Fields(nF - 1) = grdAttributes
939|   End If
940|    Next nF
941|    Data1.Recordset.Update
942|   End If
943|
```

```
944|  Next nRec
945|
946|
947|  flgIgnoreMovement = False
948|  flgChangesWereMade = False
949|
950|  Exit Sub
951|  ERRSaveAttributes:
952|
953|  If Err = 3021 Then 'No current record
954|     Data1.Recordset.AddNew
955|     Resume Next
956|  End If
957|  If Err = 91 Then 'Object variable not set.  ie, no XYBLOB to update
958|   Resume Next
959|  End If
960|  If Err = 3058 Then
961|    grdAttributes.SelStartRow = grdAttributes.Row
962|    grdAttributes.SelEndRow = grdAttributes.Row
963|    grdAttributes.SelStartCol = 1
964|    grdAttributes.SelEndRow = grdAttributes.Cols - 1
965|    n = MsgBox("You have not entered a value in a field that is part of a primary index." &
     CRLF & "This is not allowed.  Do you want to delete this record?", 36)
966|    If n = 6 Then
967|       grdAttributes.Col = 0
968|       grdAttributes = "XXX"
969|       Resume Next
970|      Else
971|       Err = 0
972|       Exit Sub
973|       Resume Next
974|    End If
975|  End If
976|  If Err = 9 Then 'Subscript out of range
977|     Resume Next 'Try ignoring it
978|  End If
979|
980|
981|
982|
983|  MsgBox "Error number " & Err & " encountered while saving attributes." & Chr$(13) &
     Chr$(10) & Error$
984|  Err = 0
985|  Exit Sub
986|  Resume
987|
988|  End Sub
989|
990|  Sub SetIgnoreMovement (ToValue As Long)
991|   flgIgnoreMovement = ToValue
992|  End Sub
993|
994|  Sub SetQuery (strTblQrySQL As String)
995|  'This subroutine sets the data control to the desired TaBLe, QueRY, or SQL string,
```

```
 996| ' then sets the labels for the attribute grid.
 997| On Error GoTo ERRSetQuery
 998|
 999| 'Check whether changes need to be saved.
1000| If flgChangesWereMade Then
1001|     'Present a yes/no/cancel box to the user.
1002|   Me.Show
1003|   nF = MsgBox("Changes were made to this data set." & CRLF & "Do you wish to save
      them now?", 36)
1004|   If nF = 2 Then Exit Sub'user chose cancel, so clear out.
1005|   If nF = 6 Then 'User chose to save changes
1006|     Call mfSaveTableChanges_Click
1007|   End If
1008|   flgChangesWereMade = False  'If the user chose yes or no, reset the flag.
1009| End If
1010|
1011| 'Set flag to not display cell contents in text box.
1012| flgIgnoreMovement = True
1013|
1014| 'Clear the grid
1015| ClearGrid Me.grdAttributes
1016|
1017| grdAttributes.RowHeight(0) = 600
1018| 'Set the data control to the desired query, table, or SQL statement.
1019| Data1.RecordSource = strTblQrySQL
1020| Data1.Refresh
1021|
1022| If Data1.Recordset.Bookmarkable Then
1023|   'ReDim strBookMark(1 To nRecs)
1024|   flgBookmarkable = True
1025|   ZUnKillMenuItems
1026|  Else
1027|   mfSaveTableChanges.Enabled = False
1028|   MsgBox "You will not be able to save changes to this table or query because it cannot
      be bookmarked."
1029|   ZKillMenuItems
1030|   flgBookmarkable = False
1031| End If
1032|
1033|
1034| flgIgnoreMovement = False
1035| grdAttributes.Visible = True
1036|
1037| Exit Sub
1038|
1039| ERRSetQuery:
1040| If Err = 3061 Then
1041|   MsgBox "You tried selecting a parameter query, and this application doesn't do those.
      Sorry!"
1042|   Err = 0
1043|   Exit Sub
1044| End If
1045|
1046| MsgBox "Error number " & Err & " occurred in SetQuery." & CRLF & Error
```

```
Err = 0
'Set cursor back to normal.
mousepointer = 0
grdAttributes.Visible = True
Exit Sub
Resume
End Sub

Sub txtAttributes_GotFocus ()
 If chkOverstrike Then
   txtAttributes.SelStart = 0
   txtAttributes.SelLength = 32765
 End If

End Sub

Sub txtAttributes_KeyPress (KeyAscii As Integer)

 If KeyAscii = 13 Then  'For a return key, move focus to the
   KeyAscii = 0       '  grid, then save the changes to the grid.
   'If Not flgBookmarkable Then
   '   Beep
   '   Beep
   '   txtAttributes = grdAttributes
   '   Exit Sub
   'End If

   flgIgnoreMovement = True
   grdAttributes.Col = 0
   If grdAttributes <> "A" Then grdAttributes = "E"
   grdAttributes.Col = lngCurrentCol
   flgIgnoreMovement = False
   flgChangesWereMade = True
   grdAttributes.SetFocus
 End If
 If KeyAscii = 27 Then   'For an ESCape key, reset the text box to
   txtAttributes = grdAttributes    'the contents of the grid cell.
   KeyAscii = 0
 End If

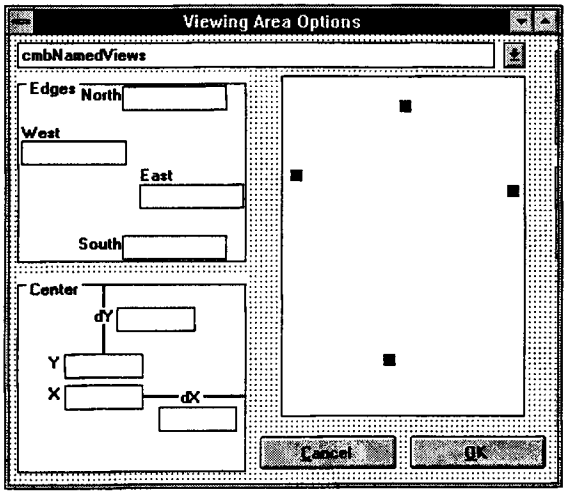

End Sub

Sub txtAttributes_LostFocus ()
   grdAttributes.Text = txtAttributes
End Sub

Sub UndoDataChanges ()
 Call Data1_Reposition
End Sub

Sub ZKillMenuItems ()
   zzEdit.Enabled = False
   mfAbandonTableChanges.Enabled = False
```

```
1101|    mfSaveTableChanges.Enabled = False
1102|  End Sub
1103|
1104|  Sub ZUnKillMenuItems ()
1105|    zzEdit.Enabled = True
1106|    mfAbandonTableChanges.Enabled = True
1107|    mfSaveTableChanges.Enabled = True
1108|  End Sub
1109|
```


Viewing Area Options
cmbNamedViews
Edges
North
West
East
South
Center
dY
Y
X
dX
Cancel
OK

```
VERSION 2.00
Begin Form frmViewPortOptions
  Caption        = "Viewing Area Options"
  ClientHeight   = 5220
  ClientLeft     = 2025
  ClientTop      = 825
  ClientWidth    = 6330
  ControlBox     = 0  'False
  Height         = 5625
  Left           = 1965
  LinkTopic      = "Form1"
  ScaleHeight    = 92.075
  ScaleMode      = 6  'Millimeter
  ScaleWidth     = 111.654
  Top            = 480
  Width          = 6450
  Begin Frame zFrameViewBounds
    Caption      = "View Bounds"
    Height       = 1185
    Left         = 6300
    TabIndex     = 3
    Top          = 1440
    Width        = 2265
    Begin Label zlblVXMin
      Caption    = "XMin"
      Height     = 195
      Left       = 90
      TabIndex   = 4
      Top        = 270
      Width      = 555
    End
    Begin Label zlblVXMax
      Caption    = "XMax"
      Height     = 195
      Left       = 90
      TabIndex   = 5
      Top        = 450
      Width      = 465
    End
    Begin Label zlblVYMin
      Caption    = "YMin"
      Height     = 195
      Left       = 90
      TabIndex   = 6
      Top        = 720
      Width      = 465
    End
    Begin Label zlblVYMax
      Caption    = "YMax"
      Height     = 195
      Left       = 90
      TabIndex   = 7
      Top        = 900
      Width      = 465
```

```
   55|     End
   56|     Begin Label lblVXMin
   57|       Caption       = "lblVXMin"
   58|       FontBold      = 0  'False
   59|       FontItalic    = 0  'False
   60|       FontName      = "MS Sans Serif"
   61|       FontSize      = 8.25
   62|       FontStrikethru = 0  'False
   63|       FontUnderline = 0  'False
   64|       Height        = 195
   65|       Left          = 630
   66|       TabIndex      = 12
   67|       Top           = 270
   68|       Width         = 1545
   69|     End
   70|     Begin Label lblVXMax
   71|       Caption       = "lblVXMax"
   72|       FontBold      = 0  'False
   73|       FontItalic    = 0  'False
   74|       FontName      = "MS Sans Serif"
   75|       FontSize      = 8.25
   76|       FontStrikethru = 0  'False
   77|       FontUnderline = 0  'False
   78|       Height        = 195
   79|       Left          = 630
   80|       TabIndex      = 13
   81|       Top           = 450
   82|       Width         = 1545
   83|     End
   84|     Begin Label lblVYMin
   85|       Caption       = "lblVYMin"
   86|       FontBold      = 0  'False
   87|       FontItalic    = 0  'False
   88|       FontName      = "MS Sans Serif"
   89|       FontSize      = 8.25
   90|       FontStrikethru = 0  'False
   91|       FontUnderline = 0  'False
   92|       Height        = 195
   93|       Left          = 630
   94|       TabIndex      = 14
   95|       Top           = 720
   96|       Width         = 1545
   97|     End
   98|     Begin Label lblVYMax
   99|       Caption       = "lblVYMax"
  100|       FontBold      = 0  'False
  101|       FontItalic    = 0  'False
  102|       FontName      = "MS Sans Serif"
  103|       FontSize      = 8.25
  104|       FontStrikethru = 0  'False
  105|       FontUnderline = 0  'False
  106|       Height        = 195
  107|       Left          = 630
  108|       TabIndex      = 15
```

```
109|       Top          =  900
110|       Width        =  1545
111|     End
112|   End
113|   Begin Frame zzFrameDataBounds
114|     Caption      =  "Data Bounds"
115|     Height       =  1185
116|     Left         =  6300
117|     TabIndex     =  16
118|     Top          =  90
119|     Width        =  2265
120|     Begin Label lblDYMax
121|       Caption       =  "lblDYMax"
122|       FontBold      =  0  'False
123|       FontItalic    =  0  'False
124|       FontName      =  "MS Sans Serif"
125|       FontSize      =  8.25
126|       FontStrikethru =  0  'False
127|       FontUnderline =  0  'False
128|       Height        =  195
129|       Left          =  630
130|       TabIndex      =  17
131|       Top           =  900
132|       Width         =  1545
133|     End
134|     Begin Label lblDYMin
135|       Caption       =  "lblDYMin"
136|       FontBold      =  0  'False
137|       FontItalic    =  0  'False
138|       FontName      =  "MS Sans Serif"
139|       FontSize      =  8.25
140|       FontStrikethru =  0  'False
141|       FontUnderline =  0  'False
142|       Height        =  195
143|       Left          =  630
144|       TabIndex      =  37
145|       Top           =  720
146|       Width         =  1545
147|     End
148|     Begin Label lblDXMax
149|       Caption       =  "lblDXMax"
150|       FontBold      =  0  'False
151|       FontItalic    =  0  'False
152|       FontName      =  "MS Sans Serif"
153|       FontSize      =  8.25
154|       FontStrikethru =  0  'False
155|       FontUnderline =  0  'False
156|       Height        =  195
157|       Left          =  630
158|       TabIndex      =  38
159|       Top           =  450
160|       Width         =  1545
161|     End
162|     Begin Label lblDXMin
```

```
163|       Caption      = "lblDXMin"
164|       FontBold     = 0  'False
165|       FontItalic   = 0  'False
166|       FontName     = "MS Sans Serif"
167|       FontSize     = 8.25
168|       FontStrikethru = 0  'False
169|       FontUnderline  = 0  'False
170|       Height       = 195
171|       Left         = 630
172|       TabIndex     = 39
173|       Top          = 270
174|       Width        = 1545
175|     End
176|     Begin Label zlblDYMax
177|       Caption      = "YMax"
178|       Height       = 195
179|       Left         = 90
180|       TabIndex     = 40
181|       Top          = 900
182|       Width        = 465
183|     End
184|     Begin Label zlblDYMin
185|       Caption      = "YMin"
186|       Height       = 195
187|       Left         = 90
188|       TabIndex     = 41
189|       Top          = 720
190|       Width        = 465
191|     End
192|     Begin Label zlblDXMax
193|       Caption      = "XMax"
194|       Height       = 195
195|       Left         = 90
196|       TabIndex     = 42
197|       Top          = 450
198|       Width        = 465
199|     End
200|     Begin Label zlblDXMin
201|       Caption      = "XMin"
202|       Height       = 195
203|       Left         = 90
204|       TabIndex     = 43
205|       Top          = 270
206|       Width        = 555
207|     End
208|   End
209|   Begin ComboBox cmbNamedViews
210|     Height       = 300
211|     Left         = 90
212|     Sorted       = -1 'True
213|     TabIndex     = 36
214|     Text         = "cmbNamedViews"
215|     Top          = 90
216|     Width        = 5865
```

```
End
Begin Frame zframeMargins
  Caption      = "Edges"
  Height       = 2175
  Left         = 90
  TabIndex     = 27
  Top          = 480
  Width        = 2625
  Begin TextBox txtBottomMargin
    Height       = 285
    Left         = 1200
    TabIndex     = 31
    Top          = 1860
    Width        = 1200
  End
  Begin TextBox txtTopMargin
    Height       = 285
    Left         = 1200
    TabIndex     = 30
    Top          = 120
    Width        = 1200
  End
  Begin TextBox txtRightMargin
    Height       = 285
    Left         = 1400
    TabIndex     = 29
    Top          = 1260
    Width        = 1200
  End
  Begin TextBox txtLeftMargin
    Height       = 285
    Left         = 45
    TabIndex     = 28
    Top          = 750
    Width        = 1200
  End
  Begin Label lblNorth
    Caption      = "North"
    Height       = 195
    Left         = 720
    TabIndex     = 35
    Top          = 120
    Width        = 555
  End
  Begin Label lblSouth
    Caption      = "South"
    Height       = 195
    Left         = 690
    TabIndex     = 34
    Top          = 1860
    Width        = 555
  End
  Begin Label lblEast
    Caption      = "East"
```

```
271|       Height      = 195
272|       Left        = 1400
273|       TabIndex    = 33
274|       Top         = 1050
275|       Width       = 645
276|     End
277|     Begin Label lblWest
278|       Caption     = "West"
279|       Height      = 195
280|       Left        = 45
281|       TabIndex    = 32
282|       Top         = 550
283|       Width       = 510
284|     End
285|   End
286|   Begin Frame zframeCenterPoint
287|     Caption       = "Center"
288|     Height        = 2265
289|     Left          = 90
290|     TabIndex      = 18
291|     Top           = 2820
292|     Width         = 2625
293|     Begin TextBox txtDY
294|       Height      = 285
295|       Left        = 1140
296|       TabIndex    = 26
297|       Top         = 360
298|       Width       = 900
299|     End
300|     Begin TextBox txtDX
301|       Height      = 285
302|       Left        = 1620
303|       TabIndex    = 25
304|       Top         = 1520
305|       Width       = 900
306|     End
307|     Begin TextBox txtYCenter
308|       Height      = 285
309|       Left        = 540
310|       TabIndex    = 22
311|       Top         = 900
312|       Width       = 900
313|     End
314|     Begin TextBox txtXCenter
315|       Height      = 285
316|       Left        = 540
317|       TabIndex    = 20
318|       Top         = 1260
319|       Width       = 900
320|     End
321|     Begin Label lbldX
322|       Alignment   = 2 'Center
323|       Caption     = "dX"
324|       Height      = 195
```

```
325|       Left        = 1850
326|       TabIndex     = 24
327|       Top         = 1310
328|       Width       = 285
329|     End
330|     Begin Line zLinedX
331|       BorderWidth  = 2
332|       X1          = 2610
333|       X2          = 1440
334|       Y1          = 1400
335|       Y2          = 1400
336|     End
337|     Begin Label lbldY
338|       Caption     = "dY"
339|       Height      = 195
340|       Left        = 870
341|       TabIndex    = 23
342|       Top         = 360
343|       Width       = 240
344|     End
345|     Begin Line zLinedY
346|       BorderWidth  = 2
347|       X1          = 990
348|       X2          = 990
349|       Y1          = 120
350|       Y2          = 900
351|     End
352|     Begin Label lblYCenter
353|       Caption     = "Y"
354|       Height      = 195
355|       Left        = 360
356|       TabIndex    = 21
357|       Top         = 900
358|       Width       = 195
359|     End
360|     Begin Label lblXCenter
361|       Caption     = "X"
362|       Height      = 195
363|       Left        = 360
364|       TabIndex    = 19
365|       Top         = 1260
366|       Width       = 195
367|     End
368|   End
369|   Begin PictureBox picPageLayout
370|     Height        = 3975
371|     Left          = 3120
372|     ScaleHeight   = 69.585
373|     ScaleMode     = 6 'Millimeter
374|     ScaleWidth    = 49.477
375|     TabIndex      = 2
376|     Top           = 480
377|     Width         = 2835
378|     Begin PictureBox picBottom
```

```
379|       BackColor      = &H00000000&
380|       DragMode       = 1 'Automatic
381|       Height         = 142
382|       Left           = 1170
383|       ScaleHeight    = 105
384|       ScaleWidth     = 105
385|       TabIndex       = 11
386|       Top            = 3240
387|       Width          = 142
388|     End
389|     Begin PictureBox picRight
390|       BackColor      = &H00000000&
391|       DragMode       = 1 'Automatic
392|       Height         = 142
393|       Left           = 2610
394|       ScaleHeight    = 105
395|       ScaleWidth     = 105
396|       TabIndex       = 10
397|       Top            = 1260
398|       Width          = 142
399|     End
400|     Begin PictureBox picTop
401|       BackColor      = &H00000000&
402|       DragMode       = 1 'Automatic
403|       Height         = 142
404|       Left           = 1350
405|       ScaleHeight    = 105
406|       ScaleWidth     = 105
407|       TabIndex       = 9
408|       Top            = 270
409|       Width          = 142
410|     End
411|     Begin PictureBox picLeft
412|       BackColor      = &H00000000&
413|       DragMode       = 1 'Automatic
414|       Height         = 142
415|       Left           = 90
416|       ScaleHeight    = 105
417|       ScaleWidth     = 105
418|       TabIndex       = 8
419|       Top            = 1080
420|       Width          = 142
421|     End
422|   End
423|   Begin CommandButton cmdCancel
424|     Cancel           = -1 'True
425|     Caption          = "&Cancel"
426|     Height           = 375
427|     Left             = 2880
428|     TabIndex         = 1
429|     Top              = 4680
430|     Width            = 1575
431|   End
432|   Begin CommandButton cmdOK
```

```
433|      Caption      =  "&OK"
434|      Height       =  375
435|      Left         =  4620
436|      TabIndex     =  0
437|      Top          =  4680
438|      Width        =  1575
439|    End
440|    Begin Label lblTrigger
441|      BackColor    =  &H0000FF00&
442|      Caption      =  "Trigger"
443|      Height       =  195
444|      Left         =  6930
445|      TabIndex     =  44
446|      Top          =  2880
447|      Visible      =  0  'False
448|      Width        =  1185
449|    End
450| End
451| Option Explicit
452| Dim ViewArea As RealRect, DataArea As RealRect
453|
454| Sub AdjustPICBoundaries (NewBoundaries As RealRect)
455|    picRight.Left = NewBoundaries.Right
456|    picLeft.Left = NewBoundaries.Left - picLeft.Width
457|    Dim VertMiddle As Single
458|    VertMiddle = .5 * (NewBoundaries.Bottom + NewBoundaries.Top)
459|    picRight.Top = VertMiddle
460|    picLeft.Top = VertMiddle
461|
462|    picTop.Top = NewBoundaries.Top + picTop.Height
463|    picBottom.Top = NewBoundaries.Bottom
464|    Dim HorzMiddle As Single
465|    HorzMiddle = .5 * (NewBoundaries.Left + NewBoundaries.Right)
466|    picTop.Left = HorzMiddle
467|    picBottom.Left = HorzMiddle
468|
469|    picPageLayout_Paint
470|
471| End Sub
472|
473| Sub cmbNamedViews_Click ()
474|     'NEW 950418
475|     tblViews.Index = "PrimaryKey"
476|     tblViews.Seek "=", Me!cmbNamedViews.ItemData(Me!cmbNamedViews.ListIndex)
477|      Me!txtLeftMargin = tblViews("MinX")
478|      Me!txtBottomMargin = tblViews("MinY")
479|      Me!txtRightMargin = tblViews("MaxX")
480|      Me!txtTopMargin = tblViews("MaxY")
481| End Sub
482|
483| Sub cmbNamedViews_KeyPress (KeyAscii As Integer)
484|  'NEW 950418
485|  If KeyAscii <> 13 Then Exit Sub
486|    'The user has hit return, which means we should save the
```

```
487|    ' current view with the specified name.
488|    Dim NewView As String
489|    NewView = Trim$(cmbNamedViews)
490|    If NewView = "" Then Exit Sub
491|    tblViews.AddNew
492|      tblViews("MinX") = Me!txtLeftMargin
493|      tblViews("MinY") = Me!txtBottomMargin
494|      tblViews("MaxX") = Me!txtRightMargin
495|      tblViews("MaxY") = Me!txtTopMargin
496|      tblViews("View Created") = Now
497|      tblViews("ViewName") = NewView
498|      tblViews("ViewSerialNumber") = Me!cmbNamedViews.Tag
499|    tblViews.Update
500|    Me!cmbNamedViews.AddItem NewView
501|    Me!cmbNamedViews.ItemData(Me!cmbNamedViews.NewIndex) =
     Me!cmbNamedViews.Tag
502|    Me!cmbNamedViews.Tag = Me!cmbNamedViews.Tag + 1
503| End Sub
504|
505| Sub cmdCancel_Click ()
506| Me.cmdCancel.SetFocus
507| Me.Hide
508| End Sub
509|
510| Sub cmdOK_Click ()
511| Me.cmdOK.SetFocus
512| Me.Hide
513| Me.cmdOK.FontItalic = True
514| End Sub
515|
516| Sub Form_Paint ()
517|  cmdOK.FontItalic = False
518| End Sub
519|
520| Sub lblDXMax_Click ()
521|  Clipboard.Clear
522|  Clipboard.SetText lblDXMax.Caption
523| End Sub
524|
525| Sub lblDXMin_Click ()
526|  Clipboard.Clear
527|  Clipboard.SetText lblDXMin.Caption
528| End Sub
529|
530| Sub lblDYMax_Click ()
531|  Clipboard.Clear
532|  Clipboard.SetText lblDYMax.Caption
533| End Sub
534|
535| Sub lblDYMin_Click ()
536|  Clipboard.Clear
537|  Clipboard.SetText lblDYMin.Caption
538| End Sub
539|
```

```
Sub lblTrigger_Change ()
'This event occurs everytime the form is shown, so that the viewport
' and datawindow are kept up to date.
'The calling routine is responsible for triggering this event, perhaps
' by setting the caption property to the current time.
DataArea.Left = lblDXMin
DataArea.Right = lblDXMax
DataArea.Top = lblDYMax
DataArea.Bottom = lblDYMin

ViewArea.Left = lblVXMin
ViewArea.Right = lblVXMax
ViewArea.Top = lblVYMax
ViewArea.Bottom = lblVYMin



picPageLayout.Height = picPageLayout.Width * (DataArea.Top - DataArea.Bottom) /
(DataArea.Right - DataArea.Left)
picPageLayout.Scale (DataArea.Left, DataArea.Top)-(DataArea.Right,
DataArea.Bottom)

AdjustPICBoundaries ViewArea
UpdateTextBoxes ViewArea

End Sub

Sub lblVXMax_Click ()
 Clipboard.Clear
 Clipboard.SetText lblVXMax.Caption
End Sub

Sub lblVXMin_Click ()
 Clipboard.Clear
 Clipboard.SetText lblVXMin.Caption
End Sub

Sub lblVYMax_Click ()
 Clipboard.Clear
 Clipboard.SetText lblVYMax.Caption
End Sub

Sub lblVYMin_Click ()
 Clipboard.Clear
 Clipboard.SetText lblVYMin.Caption
End Sub

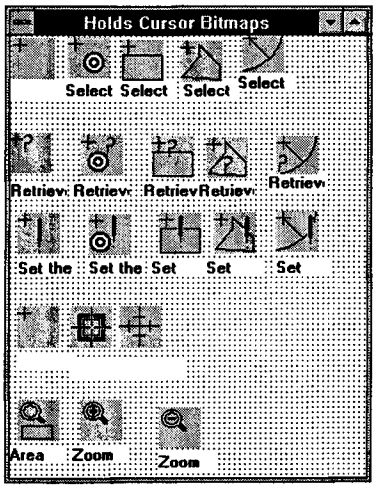
Sub picPageLayout_DragDrop (Source As Control, X As Single, Y As Single)
 Dim NewX As Single, NewY As Single
 NewX = X
 NewY = Y

 If Source Is picLeft Then ViewArea.Left = NewX + .5 * picLeft.Width
 If Source Is picRight Then ViewArea.Right = NewX
```

```
592| If Source Is picTop Then ViewArea.Top = NewY - .5 * picTop.Height
593| If Source Is picBottom Then ViewArea.Bottom = NewY
594| AdjustPICBoundaries ViewArea
595| UpdateTextBoxes ViewArea
596|
597| End Sub
598|
599| Sub picPageLayout_DragOver (Source As Control, X As Single, Y As Single, State As
     Integer)
600| Exit Sub
601| Dim NewX As Single, NewY As Single
602| NewX = X
603| NewY = Y
604|
605| If Source Is picLeft Then ViewArea.Left = NewX
606| If Source Is picRight Then ViewArea.Right = NewX
607| If Source Is picTop Then ViewArea.Top = NewY
608| 'If Source Is picBottom Then picViewArea.Height = NewY
609| AdjustPICBoundaries ViewArea
610|
611|
612| End Sub
613|
614| Sub picPageLayout_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As
     Single)
615| ';Me.Caption = "Mouse is in picPageLayout. X=" & X & ";   Y=" & Y
616| End Sub
617|
618| Sub picPageLayout_Paint ()
619|
620| If ViewArea.Left <> ViewArea.Right Then
621|    picPageLayout.Cls
622|    picPageLayout.Line (ViewArea.Left, ViewArea.Top)-(ViewArea.Right,
   ViewArea.Bottom), RGB(0, 128, 0), BF
623| End If
624| End Sub
625|
626| Sub txtBottomMargin_KeyPress (KeyAscii As Integer)
627|   If KeyAscii = 13 Then
628|      ViewArea.Bottom = txtBottomMargin
629|      UpdateViewDisplay
630|      KeyAscii = 0
631|   End If
632| End Sub
633|
634| Sub txtDX_KeyPress (KeyAscii As Integer)
635|   If KeyAscii = 13 Then
636|      ViewArea.Left = Val(txtXCenter) - Val(txtDX)
637|      ViewArea.Right = Val(txtXCenter) + Val(txtDX)
638|      UpdateViewDisplay
639|      KeyAscii = 0
640|   End If
641| End Sub
642|
```

```
643| Sub txtDY_KeyPress (KeyAscii As Integer)
644|   If KeyAscii = 13 Then
645|     ViewArea.Bottom = Val(txtYCenter) - Val(txtDY)
646|     ViewArea.Top = Val(txtYCenter) + Val(txtDY)
647|     UpdateViewDisplay
648|     KeyAscii = 0
649|   End If
650| End Sub
651|
652| Sub txtLeftMargin_KeyPress (KeyAscii As Integer)
653|   If KeyAscii = 13 Then
654|     ViewArea.Left = txtLeftMargin
655|     UpdateViewDisplay
656|     KeyAscii = 0
657|   End If
658| End Sub
659|
660| Sub txtRightMargin_KeyPress (KeyAscii As Integer)
661|   If KeyAscii = 13 Then
662|     ViewArea.Right = txtRightMargin
663|     UpdateViewDisplay
664|     KeyAscii = 0
665|   End If
666| End Sub
667|
668| Sub txtTopMargin_KeyPress (KeyAscii As Integer)
669|   If KeyAscii = 13 Then
670|     ViewArea.Top = txtTopMargin
671|     UpdateViewDisplay
672|     KeyAscii = 0
673|   End If
674| End Sub
675|
676| Sub txtXCenter_KeyPress (KeyAscii As Integer)
677|   If KeyAscii = 13 Then
678|     ViewArea.Left = Val(txtXCenter) - Val(txtDX)
679|     ViewArea.Right = Val(txtXCenter) + Val(txtDX)
680|     UpdateViewDisplay
681|     KeyAscii = 0
682|   End If
683| End Sub
684|
685| Sub txtYCenter_KeyPress (KeyAscii As Integer)
686|   If KeyAscii = 13 Then
687|     ViewArea.Bottom = Val(txtYCenter) - Val(txtDY)
688|     ViewArea.Top = Val(txtYCenter) + Val(txtDY)
689|     UpdateViewDisplay
690|     KeyAscii = 0
691|   End If
692| End Sub
693|
694| Sub UpdateTextBoxes (NewBoundaries As RealRect)
695|   txtLeftMargin = NewBoundaries.Left
696|   txtRightMargin = NewBoundaries.Right
```

```
697|   txtTopMargin = NewBoundaries.Top
698|   txtBottomMargin = NewBoundaries.Bottom
699|   txtXCenter = (NewBoundaries.Left + NewBoundaries.Right) / 2
700|   txtDX = (NewBoundaries.Right - NewBoundaries.Left) / 2
701|   txtYCenter = (NewBoundaries.Top + NewBoundaries.Bottom) / 2
702|   txtDY = (NewBoundaries.Top - NewBoundaries.Bottom) / 2
703| End Sub
704|
705| Sub UpdateViewDisplay ()
706|  AdjustPICBoundaries ViewArea
707|  UpdateTextBoxes ViewArea
708| End Sub
709|
```


Holds Cursor Bitmaps
Select
Select
Select
Select
Retriev
Retriev
Retriev
Retriev
Retriev
Set the
Set the
Set
Set
Set
Area
Zoom
Zoom

```
VERSION 2.00
Begin Form frmHoldCursorBitMaps
  Caption        = "Holds Cursor Bitmaps"
  ClientHeight   = 5145
  ClientLeft     = 480
  ClientTop      = 1500
  ClientWidth    = 4110
  Height         = 5550
  Icon           = WHTCURS2.FRX:0000
  Left           = 420
  LinkTopic      = "Form1"
  ScaleHeight    = 5145
  ScaleWidth     = 4110
  Tag            = "Query by polygon"
  Top            = 1155
  Width          = 4230
  Begin Label Label1
    Caption      = "Zoom out 40% centered on point"
    Height       = 195
    Index        = 20
    Left         = 1710
    TabIndex     = 20
    Top          = 4860
    Width        = 645
  End
  Begin Label Label1
    Caption      = "Zoom in 30% centered on point"
    Height       = 195
    Index        = 19
    Left         = 720
    TabIndex     = 19
    Top          = 4770
    Width        = 645
  End
  Begin Label Label1
    Caption      = "Area viewed approximates the area selected."
    Height       = 195
    Index        = 18
    Left         = 0
    TabIndex     = 18
    Top          = 4770
    Width        = 645
  End
  Begin Label Label1
    Height       = 195
    Index        = 17
    Left         = 1350
    TabIndex     = 17
    Top          = 3780
    Width        = 645
  End
  Begin Label Label1
    Height       = 195
    Index        = 16
```

```
    Left        = 720
    TabIndex    = 16
    Top         = 3780
    Width       = 645
  End
  Begin Label Label1
    Height      = 195
    Index       = 15
    Left        = 90
    TabIndex    = 15
    Top         = 3690
    Width       = 645
  End
  Begin Label Label1
    Caption     = "Set value(s) for all points within radius"
    Height      = 195
    Index       = 14
    Left        = 3060
    TabIndex    = 14
    Top         = 2610
    Width       = 645
  End
  Begin Label Label1
    Caption     = "Set values for all points within polygon"
    Height      = 195
    Index       = 13
    Left        = 2250
    TabIndex    = 13
    Top         = 2610
    Width       = 645
  End
  Begin Label Label1
    Caption     = "Set values for all points within box"
    Height      = 195
    Index       = 12
    Left        = 1620
    TabIndex    = 12
    Top         = 2610
    Width       = 645
  End
  Begin Label Label1
    Caption     = "Set the value(s) for the closest point for a direct hit"
    Height      = 195
    Index       = 11
    Left        = 900
    TabIndex    = 11
    Top         = 2610
    Width       = 645
  End
  Begin Label Label1
    Caption     = "Set the value(s) for the closest point"
    Height      = 195
    Index       = 10
    Left        = 90
```

```
109|     TabIndex     = 10
110|     Top          = 2610
111|     Width        = 645
112|   End
113|   Begin Label Label1
114|     Caption      = "Retrieve information on all points within a user-specified radius of a
      given point."
115|     Height       = 195
116|     Index        = 9
117|     Left         = 2970
118|     TabIndex     = 9
119|     Top          = 1620
120|     Width        = 645
121|   End
122|   Begin Label Label1
123|     Caption      = "Retrieve information on all points within a polygon"
124|     Height       = 195
125|     Index        = 8
126|     Left         = 2160
127|     TabIndex     = 8
128|     Top          = 1710
129|     Width        = 645
130|   End
131|   Begin Label Label1
132|     Caption      = "Retrieve information on all points within box"
133|     Height       = 195
134|     Index        = 7
135|     Left         = 1530
136|     TabIndex     = 7
137|     Top          = 1710
138|     Width        = 645
139|   End
140|   Begin Label Label1
141|     Caption      = "Retrieve information on closest point within a threshold radius"
142|     Height       = 195
143|     Index        = 6
144|     Left         = 720
145|     TabIndex     = 6
146|     Top          = 1710
147|     Width        = 645
148|   End
149|   Begin Label Label1
150|     Caption      = "Retrieve information on closest point"
151|     Height       = 195
152|     Index        = 5
153|     Left         = 0
154|     TabIndex     = 5
155|     Top          = 1710
156|     Width        = 645
157|   End
158|   Begin Label Label1
159|     Caption      = "Select all points within a user-specified radius"
160|     Height       = 195
161|     Index        = 4
```

```
   Left        = 2610
   TabIndex    = 4
   Top         = 450
   Width       = 645
End
Begin Label Label1
   Caption     = "Select all points inside user-drawn polygon"
   Height      = 195
   Index       = 3
   Left        = 1980
   TabIndex    = 3
   Top         = 540
   Width       = 645
End
Begin Label Label1
   Caption     = "Select all objects within box"
   Height      = 195
   Index       = 2
   Left        = 1260
   TabIndex    = 2
   Top         = 540
   Width       = 645
End
Begin Label Label1
   Caption     = "Select only very close to point"
   Height      = 195
   Index       = 1
   Left        = 630
   TabIndex    = 1
   Top         = 540
   Width       = 645
End
Begin Label Label1
   Height      = 195
   Index       = 0
   Left        = 0
   TabIndex    = 0
   Top         = 540
   Width       = 645
End
Begin Image image1
   Height      = 480
   Index       = 18
   Left        = 90
   Picture     = WHTCURS2.FRX:0302
   Tag         = "Zoom by box"
   Top         = 4230
   Width       = 480
End
Begin Image image1
   Height      = 480
   Index       = 19
   Left        = 810
   Picture     = WHTCURS2.FRX:057C
```

```
216|     Tag          = "Zoom in"
217|     Top          = 4230
218|     Width        = 480
219|   End
220|   Begin Image image1
221|     Height       = 480
222|     Index        = 20
223|     Left         = 1710
224|     Picture      = WHTCURS2.FRX:07F6
225|     Tag          = "Zoom out"
226|     Top          = 4320
227|     Width        = 480
228|   End
229|   Begin Image image1
230|     Height       = 480
231|     Index        = 17
232|     Left         = 1260
233|     Picture      = WHTCURS2.FRX:0A70
234|     Tag          = "Gunsight cursor 2"
235|     Top          = 3150
236|     Width        = 480
237|   End
238|   Begin Image image1
239|     Height       = 480
240|     Index        = 15
241|     Left         = 90
242|     Picture      = WHTCURS2.FRX:0CEA
243|     Tag          = "Small cursor"
244|     Top          = 3150
245|     Width        = 480
246|   End
247|   Begin Image image1
248|     Height       = 480
249|     Index        = 16
250|     Left         = 690
251|     Picture      = WHTCURS2.FRX:0F64
252|     Tag          = "Gunsight cursor 1"
253|     Top          = 3150
254|     Width        = 480
255|   End
256|   Begin Image image1
257|     Height       = 480
258|     Index        = 12
259|     Left         = 1710
260|     Picture      = WHTCURS2.FRX:11DE
261|     Tag          = "Set within box"
262|     Top          = 2070
263|     Width        = 480
264|   End
265|   Begin Image image1
266|     Height       = 480
267|     Index        = 9
268|     Left         = 3060
269|     Picture      = WHTCURS2.FRX:1458
```

```
270|     Tag         = "Query by radius"
271|     Top         = 1170
272|     Width       = 480
273|   End
274|   Begin Image image1
275|     Height      = 480
276|     Index       = 4
277|     Left        = 2670
278|     Picture     = WHTCURS2.FRX:16D2
279|     Tag         = "Select within radius"
280|     Top         = -45
281|     Width       = 480
282|   End
283|   Begin Image image1
284|     Height      = 480
285|     Index       = 13
286|     Left        = 2340
287|     Picture     = WHTCURS2.FRX:194C
288|     Tag         = "Set within polygon"
289|     Top         = 2070
290|     Width       = 480
291|   End
292|   Begin Image image1
293|     Height      = 480
294|     Index       = 14
295|     Left        = 3060
296|     Picture     = WHTCURS2.FRX:1BC6
297|     Tag         = "Set within radius"
298|     Top         = 2055
299|     Width       = 480
300|   End
301|   Begin Image image1
302|     Height      = 480
303|     Index       = 11
304|     Left        = 900
305|     Picture     = WHTCURS2.FRX:1E40
306|     Tag         = "Set by direct hit"
307|     Top         = 2085
308|     Width       = 480
309|   End
310|   Begin Image image1
311|     Height      = 480
312|     Index       = 10
313|     Left        = 90
314|     Picture     = WHTCURS2.FRX:20BA
315|     Tag         = "Set by point"
316|     Top         = 2070
317|     Width       = 480
318|   End
319|   Begin Image image1
320|     Height      = 480
321|     Index       = 0
322|     Left        = 30
323|     Picture     = WHTCURS2.FRX:2334
```

```
324|     Tag          = "Select by point"
325|     Top          = 30
326|     Width        = 480
327|   End
328|   Begin Image image1
329|     Height       = 480
330|     Index        = 1
331|     Left         = 660
332|     Picture      = WHTCURS2.FRX:25AE
333|     Tag          = "Select on point"
334|     Top          = 15
335|     Width        = 480
336|   End
337|   Begin Image image1
338|     Height       = 480
339|     Index        = 2
340|     Left         = 1275
341|     Picture      = WHTCURS2.FRX:2828
342|     Tag          = "Select all within box"
343|     Top          = 45
344|     Width        = 480
345|   End
346|   Begin Image image1
347|     Height       = 480
348|     Index        = 3
349|     Left         = 1950
350|     Picture      = WHTCURS2.FRX:2AA2
351|     Tag          = "Select polygon"
352|     Top          = 60
353|     Width        = 480
354|   End
355|   Begin Image image1
356|     Height       = 480
357|     Index        = 5
358|     Left         = -15
359|     Picture      = WHTCURS2.FRX:2D1C
360|     Tag          = "Query by point"
361|     Top          = 1125
362|     Width        = 480
363|   End
364|   Begin Image image1
365|     Height       = 480
366|     Index        = 6
367|     Left         = 810
368|     Picture      = WHTCURS2.FRX:2F96
369|     Tag          = "Query close point"
370|     Top          = 1155
371|     Width        = 480
372|   End
373|   Begin Image image1
374|     Height       = 480
375|     Index        = 7
376|     Left         = 1620
377|     Picture      = WHTCURS2.FRX:3210
```

```
      Tag          =  "Query by box"
      Top          =  1170
      Width        =  480
    End
    Begin Image image1
      Height       =  480
      Index        =  8
      Left         =  2250
      Picture      =  WHTCURS2.FRX:348A
      Tag          =  "Query by polygon"
      Top          =  1185
      Width        =  480
    End
End
Option Explicit

Sub Form_Load ()
 LoadWheatCursors
End Sub

Sub LoadWheatCursors ()
Dim junk As Integer
junk = AddNewCursor("WhtCurs2.DLL", 1001&, "PickPoint01")
Call SetToolPick(TOOL_PlainPointer, junk, "Pick a point", TOOL_PlainPointer)
junk = AddNewCursor("WhtCurs2.DLL", 1002&, "Gunsight1")
Call SetToolPick(TOOL_GunSight1, junk, "GunSight Pointer 1", TOOL_GunSight1)
junk = AddNewCursor("WhtCurs2.DLL", 1003&, "Gunsight2")
Call SetToolPick(TOOL_GunSight2, junk, "GunSight Pointer 2", TOOL_GunSight2)
junk = AddNewCursor("WhtCurs2.DLL", 1004&, "Gunsight3")
junk = AddNewCursor("WhtCurs2.DLL", 1005&, "Invert1")
junk = AddNewCursor("WhtCurs2.DLL", 1006&, "HighLight001")
junk = AddNewCursor("WhtCurs2.DLL", 1007&, "HighLight002")
junk = AddNewCursor("WhtCurs2.DLL", 1102&, "PickPointBullsEye")
junk = AddNewCursor("WhtCurs2.DLL", 1150&, "QueryByPoint")
Call SetToolPick(TOOL_QueryByPoint, junk, "Find Closest Point", TOOL_QueryByPoint)
junk = AddNewCursor("WhtCurs2.DLL", 1151&, "QueryPntBullsI")
Call SetToolPick(TOOL_QueryByPointBullsEye, junk, "Find Close Point",
    TOOL_QueryByPointBullsEye)
junk = AddNewCursor("WhtCurs2.DLL", 1175&, "SetByPoint1")
junk = AddNewCursor("WhtCurs2.DLL", 1176&, "SetByPointBullsEye")
junk = AddNewCursor("WhtCurs2.DLL", 1202&, "PickBox")
junk = AddNewCursor("WhtCurs2.DLL", 1250&, "QueryByBox1")
Call SetToolPick(TOOL_QueryByBox1, junk, "Find all points within box",
    TOOL_QueryByBox1)
junk = AddNewCursor("WhtCurs2.DLL", 1275&, "SetByBox1")
junk = AddNewCursor("WhtCurs2.DLL", 1301&, "QueryByClosestLine")
Call SetToolPick(TOOL_QueryByClosestLine, junk, "Find closest line",
    TOOL_QueryByClosestLine)
junk = AddNewCursor("WhtCurs2.DLL", 1302&, "PickPolygon")
junk = AddNewCursor("WhtCurs2.DLL", 1350&, "QueryByPolygon1")
Call SetToolPick(TOOL_QueryByPolygon1, junk, "Find all points within polygon",
    TOOL_QueryByPolygon1)
junk = AddNewCursor("WhtCurs2.DLL", 1351&, "QueryInMultiPolygon")
Call SetToolPick(TOOL_QueryInMultiPolygon, junk, "Find all polygons including point",
```

```
    TOOL_QueryInMultiPolygon)
428| junk = AddNewCursor("WhtCurs2.DLL", 1375&, "SetByPolygon1")
429| junk = AddNewCursor("WhtCurs2.DLL", 1402&, "PickRadius")
430| junk = AddNewCursor("WhtCurs2.DLL", 1450&, "QueryWithinRadius1")
431| Call SetToolPick(TOOL_QueryWithinRadius1, junk, "Find all points within radius",
    TOOL_QueryWithinRadius1)
432| junk = AddNewCursor("WhtCurs2.DLL", 1475&, "SetWithinRadius1")
433| junk = AddNewCursor("WhtCurs2.DLL", 1500&, "ZoomByBox")
434| Call SetToolPick(TOOL_ZoomByBox, junk, "Zoom within rectangle ",
    TOOL_ZoomByBox)
435| junk = AddNewCursor("WhtCurs2.DLL", 1501&, "Zoomin")
436| junk = AddNewCursor("WhtCurs2.DLL", 1502&, "ZoomOut")
437| junk = AddNewCursor("WhtCurs2.DLL", 1600&, "WaitV")
438| junk = AddNewCursor("WhtCurs2.DLL", 1601&, "WaitH")
439| junk = AddNewCursor("WhtCurs2.DLL", 1610&, "WaitCoffeV")
440| junk = AddNewCursor("WhtCurs2.DLL", 1611&, "WaitCoffeH")
441| junk = AddNewCursor("WhtCurs2.DLL", 1620&, "WaitDinnerV")
442| junk = AddNewCursor("WhtCurs2.DLL", 1621&, "WaitDinnerH")
443| junk = AddNewCursor("WhtCurs2.DLL", 1622&, "Wait_GoEatV")
444| junk = AddNewCursor("WhtCurs2.DLL", 1623&, "Wait_GoEatH")
445| junk = AddNewCursor("WhtCurs2.DLL", 1630&, "WaitSleepV")
446| junk = AddNewCursor("WhtCurs2.DLL", 1631&, "WaitSleepH")
447| junk = AddNewCursor("WhtCurs2.DLL", 1701&, "QueryByTile001")
448| Call SetToolPick(TOOL_QueryByTile, junk, "Find tile enclosing point ",
    TOOL_QueryByTile)
449| junk = AddNewCursor("WhtCurs2.DLL", 2006&, "BOX06")
450| junk = AddNewCursor("WhtCurs2.DLL", 2010&, "BOX10")
451| Call SetToolPick(TOOL_BOX10, junk, "Find feature near cursor ", TOOL_BOX10)
452| junk = AddNewCursor("WhtCurs2.DLL", 2020&, "BOX20")
453| junk = AddNewCursor("WhtCurs2.DLL", 2031&, "BOX31")
454|
455| 'junk = AddNewCursor("WhtCurs2.DLL", -1&, "Unload the DLL by using the -1 flag")
456| End Sub
457|
458| Sub SetToolPick (ToolIDNum As Integer, CursorIndex As Integer, CmdName As String,
    CmdIDNum As Integer)
459| 'This sets the specified ToolPick structure (as IDNum) to the values specified in
460| 'CursorIndex is the location of the appropriate Cursor and Icon in the CursorTable array.
461| ' CmdName and CmdIDNum are for user use.
462|
463| 'Type ToolDescription
464| ' CmdName As String
465| ' CmdIDNum As Integer
466| ' hCursor As Integer
467| ' hIcon As Integer
468| 'End Type
469|  ToolPick(ToolIDNum).CmdName = CmdName
470|  ToolPick(ToolIDNum).CmdIDNum = CmdIDNum
471|  ToolPick(ToolIDNum).hCursor = CursorTable(CursorIndex).hCursor
472|  ToolPick(ToolIDNum).hIcon = CursorTable(CursorIndex).hIcon
473|
474| End Sub
475|
```

```
Option Explicit
Dim I As Long
Type typOZQuestion
   KindOfControl As Long
    '0 label
    '1 Text Box
    '2 drop down Combobox
    '3 picturebox
    '4 command button
   'Left As Single     'These all need to be in millimeters.
   'Top As Single
   'Height As Single
   'Width As Single
   'Index As Single
   StringData As String
   Extra As Variant
End Type

Global Const CONTROL_LABEL = 0
Global Const CONTROL_TEXTBOX = 1
Global Const CONTROL_COMBOBOX = 2
Global Const CONTROL_PICTUREBOX = 3
Global Const CONTROL_COMMANDBUTTON = 4
Global Const CONTROL_CHECKBOX = 5
Global Const CONTROL_OPTIONBUTTON = 6

Sub AskOZ (ListOfControls() As typOZQuestion, UserChoice As Integer)
'UserChoice will be +1 if the user chose the Next> button
'        will be -1 if the user chose the <Back button
'              0 if the user chose the cancel button
'             -2                StartOver
'             +2                 Take all defaults
'
'
On Error GoTo ERRAskOZ
 On Error GoTo 0
 Dim CurrentTop As Single
 HideOldControls

 Dim nControls As Long, N As Long
 nControls = UBound(ListOfControls) - LBound(ListOfControls) + 1
 Dim TopAt As Single
 I = 0
For N = LBound(ListOfControls) To UBound(ListOfControls)
   I = I + 1
   Select Case ListOfControls(N).KindOfControl
      Case CONTROL_LABEL:       LoadLabel ListOfControls(N), TopAt
      Case CONTROL_TEXTBOX:     LoadTextBox ListOfControls(N), TopAt
      Case CONTROL_COMBOBOX:    LoadComboBox ListOfControls(N), TopAt
      Case CONTROL_PICTUREBOX:  LoadPictureBox ListOfControls(N), TopAt
      Case CONTROL_CHECKBOX:    LoadCheckBox ListOfControls(N), TopAt
      Case CONTROL_OPTIONBUTTON: LoadOption ListOfControls(N), TopAt
   End Select
Next N
```

```
'Show the form modally (can't shift focus to elsewhere in the application.
frmOZ.Show 1
UserChoice = Val(frmOZ.lblUserChoice(0).Caption)
'If frmOZ.ActiveControl Is frmOZ.cmdOK(0) Then
'    cmdOK = True
' Else
'    cmdOK = False
'End If

'Get the user's answers, and unload the controls.
I = 0

For N = LBound(ListOfControls) To UBound(ListOfControls)
   I = I + 1 ' ListOfControls(N).Index
   Select Case ListOfControls(N).KindOfControl
      Case CONTROL_LABEL
        ListOfControls(N).StringData = frmOZ.Label1(I).Caption
        Unload frmOZ.Label1(I)
      Case CONTROL_TEXTBOX
        ListOfControls(N).StringData = frmOZ.Text1(I).Text
        Unload frmOZ.Text1(I)
      Case CONTROL_COMBOBOX
        ListOfControls(N).StringData = frmOZ.Combo1(I)
        ListOfControls(N).Extra = frmOZ.Combo1(I).ListIndex
        Unload frmOZ.Combo1(I)
      Case CONTROL_PICTUREBOX
        ListOfControls(N).Extra = frmOZ.Picture1(I)
        Unload frmOZ.Picture1(I)
      Case CONTROL_COMMANDBUTTON
        ListOfControls(N).Extra = frmOZ.Command1(I).FontItalic
        Unload frmOZ.Command1(I)
      Case CONTROL_CHECKBOX
        ListOfControls(N).Extra = frmOZ.Check1(I).Value
        Unload frmOZ.Check1(I)
      Case CONTROL_OPTIONBUTTON
        ListOfControls(N).Extra = frmOZ.Option1(I).Value
        Unload frmOZ.Option1(I)
   End Select
Next N
Unload frmOZ
Exit Sub

ERRAskOZ:
Select Case Err
 Case Else
  MsgBox "Error #" & Err & " occurred in AskOZ." & CRLF & Error
  Err = 0
  Exit Sub
  Resume
End Select


End Sub
```

```
109|
110| Function GetColorWithCmnDialog (ByVal ColorIsNow As Long) As Long
111|    Dim Temp As Long
112|    Temp = ColorIsNow
113|
114|    If ColorIsNow = 0 Then ColorIsNow = 128
115|    frmOZ.CMDialog1(0).Color = ColorIsNow
116|    frmOZ.CMDialog1(0).CancelError = True
117| On Error Resume Next 'This will only skip the color setting, so we're safe.
118|    frmOZ.CMDialog1(0).Flags = CC_RGBINIT Or CC_FULLOPEN 'Show the full dialog
    box (allowing custom colors) and initialize the value.
119|    frmOZ.CMDialog1(0).Action = 3 ' so open the color common dialog box.
120|    Temp = frmOZ.CMDialog1(0).Color'Only executed if user chose OK.
121|    frmOZ.CMDialog1(0).CancelError = False
122|    GetColorWithCmnDialog = Temp
123| End Function
124|
125| Sub HideOldControls ()
126| '940622
127| 'On Error Resume Next 'Cheap way out
128| 'This subroutine removes indexed controls on a form
129| ' and makes the 0 index controls invisible.
130| 'This is used to clear previous controls off the GREAT OZ!
131|
132| Dim N As Long, nControls As Long
133| nControls = frmOZ.Controls.Count
134| For N = frmOZ.Controls.Count - 1 To 0 Step -1
135|  If frmOZ(N).Index <> 0 Then
136|       Unload frmOZ(N)
137|     Else
138|       If TypeOf frmOZ(N) Is CommonDialog Then
139|         'do nothing
140|       Else
141|         frmOZ(N).Visible = False
142|       End If
143|  End If
144| Next N
145| frmOZ.cmdCancel(0).Visible = True
146| frmOZ.cmdAllDefaults(0).Visible = True
147| frmOZ.cmdBack(0).Visible = True
148| frmOZ.cmdNext(0).Visible = True
149| frmOZ.cmdStartOver(0).Visible = True
150| End Sub
151|
152| Sub LoadCheckBox (myData As typOZQuestion, TopAt As Single)
153|  Load frmOZ.Check1(I)
154|  frmOZ.Check1(I).Top = TopAt
155|  frmOZ.Check1(I).Left = 2 'All controls will be in 2 mm from left
156|  frmOZ.Check1(I).Height = frmOZ.TextHeight(myData.StringData)
157|  frmOZ.Check1(I).Width = frmOZ.Combo1(0).Width
158|  frmOZ.Check1(I).Caption = myData.StringData
159|  frmOZ.Check1(I).Visible = True
160| 'Update the starting top location.
161|  TopAt = TopAt + frmOZ.Check1(I).Height + .5
```

```
End Sub

Sub LoadComboBox (myData As typOZQuestion, TopAt As Single)
  Load frmOZ.Combo1(I)
  frmOZ.Combo1(I).Clear
  frmOZ.Combo1(I).Top = TopAt
  Call DlmtdStngsToComboBox(CRLF, myData.StringData, frmOZ.Combo1(I))
  frmOZ.Combo1(I).ListIndex = myData.Extra
  frmOZ.Combo1(I).Visible = True
'Update the starting top location.
  TopAt = TopAt + frmOZ.Combo1(I).Height + .5

End Sub

Sub LoadLabel (myData As typOZQuestion, TopAt As Single)
  Load frmOZ.Label1(I)
  frmOZ.Label1(I).Top = TopAt
  frmOZ.Label1(I).Left = 2 'All controls will be in 2 mm from left
  'frmOZ.Label1(I).Height = frmOZ.TextHeight(myData.StringData)
  frmOZ.Label1(I).Width = frmOZ.Combo1(0).Width
                          'frmOZ.TextWidth(myData.StringData)
  frmOZ.Label1(I).Caption = myData.StringData
  frmOZ.Label1(I).Visible = True
'Update the starting top location.
  TopAt = TopAt + frmOZ.Label1(I).Height + .5
End Sub

Sub LoadOption (myData As typOZQuestion, TopAt As Single)
  Load frmOZ.Option1(I)
  frmOZ.Option1(I).Top = TopAt
  frmOZ.Option1(I).Left = 2 'All controls will be in 2 mm from left
  frmOZ.Option1(I).Height = frmOZ.TextHeight(myData.StringData)
  frmOZ.Option1(I).Width = frmOZ.Combo1(0).Width
  frmOZ.Option1(I).Caption = myData.StringData
  frmOZ.Option1(I).Visible = True
'Update the starting top location.
  TopAt = TopAt + frmOZ.Option1(I).Height + 2

End Sub

Sub LoadPictureBox (myData As typOZQuestion, TopAt As Single)
  Load frmOZ.Picture1(I)
  frmOZ.Picture1(I).Top = TopAt
  frmOZ.Picture1(I).Left = 2 'All controls will be in 2 mm from left
  frmOZ.Picture1(I).Height = frmOZ.TextHeight(myData.StringData)
  frmOZ.Picture1(I).Width = frmOZ.TextWidth(myData.StringData)

  frmOZ.Picture1(I).Visible = True
'Update the starting top location.
  TopAt = TopAt + frmOZ.Picture1(I).Height + .5
  If (myData.StringData <> "") Then
     frmOZ.Picture1(I) = LoadPicture(myData.StringData)
       Else If myData.Extra <> 0 Then frmOZ.Picture1(I) = myData.Extra
  End If
```

```
216|
217| End Sub
218|
219| Sub LoadTextBox (myData As typOZQuestion, TopAt As Single)
220|   Load frmOZ.Text1(I)
221|   frmOZ.Text1(I).Top = TopAt
222|   frmOZ.Text1(I).Left = 2 'All controls will be in 2 mm from left
223|   frmOZ.Text1(I).Height = frmOZ.TextHeight(myData.StringData)
224|   'If I = 1 Then
225|   '  frmOZ.Text1(I).Width = 4 * frmOZ.TextWidth(myData.StringData)
226|   ' Else
227|     frmOZ.Text1(I).Width = frmOZ.Combo1(0).Width
228|   'End If
229|   frmOZ.Text1(I).Text = myData.StringData
230|   frmOZ.Text1(I).Visible = True
231| 'Update the starting top location.
232|   TopAt = TopAt + frmOZ.Text1(I).Height + .5
233|
234| End Sub
235|
```

```
 1| Option Explicit
 2| Type CursorSet
 3|  hCursor As Integer
 4|  hIcon As Integer
 5|  IDInFile As Long
 6|  Name As String
 7| End Type
 8|
 9| 'Type APIPoint
10| '   X As Integer
11| '   Y As Integer
12| 'End Type
13| Declare Sub GetCursorPos Lib "User" (lpPoint As APIPOINT)
14| Declare Sub SetCursorPos Lib "User" (ByVal X As Integer, ByVal Y As Integer)
15|
16|
17|
18| Global CursorTable() As CursorSet
19| Global nCursors As Integer
20|
21| 'Declarations for Cursor, ClassWord, and Icon functions, used to pull resources out of a
    DLL or EXE.
22| Declare Function LoadLibrary Lib "kernel" (ByVal LibName As String) As Integer
23| Declare Sub FreeLibrary Lib "Kernel" (ByVal hLibModule As Integer)
24| Declare Function LoadIcon Lib "User" (ByVal hInstance As Integer, ByVal lpIconName
    As Any) As Integer
25| Declare Function LoadCursor Lib "user" (ByVal hInstance As Integer, ByVal
    CursorName As Any) As Integer
26| Declare Function SetClassWord Lib "User" (ByVal hWnd As Integer, ByVal nIndex As
    Integer, ByVal wNewWord As Integer) As Integer
27| Declare Function GetClassword Lib "User" (ByVal hWnd As Integer, ByVal nIndex As
    Integer) As Integer
28| Declare Function DestroyCursor Lib "User" (ByVal hCursor As Integer) As Integer
29| Declare Function DestroyIcon Lib "User" (ByVal hIcon As Integer) As Integer
30| Global Const GCW_HCURSOR = (-12)
31|
32| Dim DummyInt As Integer
33| Dim DummyLong As Long
34| Dim DummySingle As Single
35| Dim DummyDouble As Double
36| Dim DummyString As String
37|
38| Dim CursorID1Int4 As Long
39| Dim CursorID2Int4 As Long
40| Global Const IDC_ARROW = 1038
41| ' TOOL_Cmd constants for use in setting cursors for graphical display of function
42| '  in frmMap.picMap.
43| 'Utility Pointers and such, TOOL_ < 100
44| Global Const TOOL_ArrowPointer = -1
45| Global Const TOOL_PlainPointer = 0
46|
47| Global Const TOOL_ZoomByBox = 1
48| Global Const TOOL_ZoomIn = 2
49| Global Const TOOL_ZoomOut = 3
```

```
50|          'XXX
51| '        'Query pointers TOOL_ between 100 and 200 This coding is a good idea, but I
    need to figure out how to do the lookup and not use this as an array index.
52| 'Global Const TOOL_QueryByPoint = 100
53| 'Global Const TOOL_QueryByPointBullsEye = 101
54| 'Global Const TOOL_QueryWithinRadius1 = 110
55| 'Global Const TOOL_QueryByBox1 = 120
56| 'Global Const TOOL_QueryByPolygon1 = 130
57| 'Global Const TOOL_QueryInMultiPolygon = 131
58| 'Global Const TOOL_QueryByClosestLine = 140
59|
60| Global Const TOOL_QueryByPoint = 4
61| Global Const TOOL_QueryByPointBullsEye = 5
62| Global Const TOOL_QueryWithinRadius1 = 6
63| Global Const TOOL_QueryByBox1 = 7
64| Global Const TOOL_QueryByPolygon1 = 8
65| Global Const TOOL_QueryInMultiPolygon = 9
66| Global Const TOOL_QueryByClosestLine = 10
67| Global Const TOOL_QueryByTile = 11
68| Global Const TOOL_BOX10 = 12
69|
70| Global Const TOOL_GunSight1 = 20
71| Global Const TOOL_GunSight2 = 21
72|
73| Global Const TOOL_DrawPoint = 30
74| Global Const TOOL_DrawLine = 32
75| Global Const TOOL_DrawPolygon = 36
76|
77| Function AddNewCursor (DLLName As String, IDInFile As Long, CallIt As String) As
    Integer
78|  Static DLLInstance As Integer, nCount  As Long
79|  Static Temp As Integer
80| 'This function will load a new cursor to memory from a Dynamic Link Library (DLL)
81| '    or even probably from an .exe.
82| 'If it is called with the argument IDInFile=-1, it will unload the module.
83| If IDInFile = -1 Then
84|    GoSub KillDLL
85|    AddNewCursor = -1
86|    Exit Function
87| End If
88| nCount = nCount + 1
89| 'First, load the library (or EXE) containing the Cursor/Icon resources
90|  DLLInstance = LoadLibrary(ByVal DLLName)
91| 'Next, try loading the Cursor
92|  Temp = LoadCursor(DLLInstance, IDInFile)
93| If Temp <> 0 Then  'successfully loaded the cursor, so increment number of
94|                    ' cursors, store the handle and name and ID# and icon handle in the
    CursorTable array
95|   nCursors = nCursors + 1
96|   AddNewCursor = nCursors 'Return the index of this Cursor/Icon resource
97|   ReDim Preserve CursorTable(nCursors)
98|   CursorTable(nCursors).hCursor = Temp   'store the cursor handle
99| 'Commented the following line out in the hopes of freeing some memory
100| '  GWPouch 940831
```

```
101|  'CursorTable(nCursors).hIcon = LoadIcon(DLLInstance, IDInFile)  'the icon handle,
      which will usually be zero
102|   CursorTable(nCursors).IDInFile = IDInFile       'the ID# in the file
103|   CursorTable(nCursors).Name = CallIt             ' and a name.
104|
105|
106|  Else          'Could not find cursor, so do not increment the array, and return a zero
107|    AddNewCursor = 0
108|  End If
109|  Exit Function
110|
111|
112|  KillDLL:
113|  Dim n As Long
114|  'This loops until it  removes the DLL
115|  For n = 1 To nCount: FreeLibrary (DLLInstance): Next n: nCount = 0
116|  End Function
117|
118|  Function GetHCursorByName (CursorName As String) As Integer
119|  Dim nCurs As Integer
120|  For nCurs = 1 To nCursors
121|   If (CursorTable(nCurs).Name = CursorName) Then
122|     GetHCursorByName = CursorTable(nCurs).hCursor
123|   End If
124|  Next nCurs
125|
126|  End Function
127|
128|  Function SetCursorByHCURSOR (TargetControl As Control, hCursor As Integer) As
      Integer
129|  'Return value is previous cursor's HCURSOR handle.
130|  'Use this to restore cursor shape to correct value.
131|
132|    SetCursorByHCURSOR = SetClassWord(TargetControl.hWnd, GCW_HCURSOR,
      hCursor)
133|  End Function
134|
135|  Function SetCursorByIndex (TargetControl As Control, CursorIndex As Integer) As
      Integer 'Note that this returns the last cursor's hCursor handle
136|  'Return value is previous cursor's HCURSOR handle
137|   SetCursorByIndex = SetClassWord(TargetControl.hWnd, GCW_HCURSOR,
      CursorTable(CursorIndex).hCursor)
138|  End Function
139|
140|  Function SetCursorByName (TargetControl As Control, CursorName As String) As
      Integer
141|  'This function sets the cursor of the target control to that specified by CursorName.
142|  'The return value is the HCursor of the previous cursor, used to reset the cursor later.
143|  ' SAVE THE RETURN VALUE TO RESTORE THE CURSOR.
144|  Dim nCurs As Integer
145|  For nCurs = 1 To nCursors
146|   If (CursorTable(nCurs).Name = CursorName) Then
147|     SetCursorByName = SetCursorByIndex(TargetControl, nCurs)
148|   End If
```

```
149|  Next nCurs
150|  End Function
151|
152|  Sub ShiftCursor (XShift As Integer, YShift As Integer)
153|     Static CursorPos As APIPOINT
154|     Call GetCursorPos(CursorPos)
155|     Call SetCursorPos(CursorPos.X + XShift, CursorPos.Y + YShift)
156|  End Sub
157|
158|  Sub UnloadDLL (hInstance As Integer)
159|
160|  End Sub
161|
162|  Sub ZapCursors ()
163|   Dim n As Integer
164|   For n = nCursors To 0 Step -1
165|    DummyInt = DestroyCursor(CursorTable(n).hCursor)
166|   Next n
167|  End Sub
168|
169|  Sub ZapIcons ()
170|   Dim n As Integer
171|   For n = nCursors To 0 Step -1
172|    DummyInt = DestroyIcon(CursorTable(n).hIcon)
173|   Next n
174|  End Sub
175|
```

```
 1| Option Explicit
 2| Option Compare Text
 3| Dim XNull As Integer
 4| Dim XDiv0 As Integer
 5| Dim strNULL As String
 6|
 7| Global CRLF As String * 2, CR As String * 1, LF As String * 1
 8| Global TabChar As String * 1, DoubleQuote As String * 1, SingleQuote As String * 1
 9|
10|
11| ' Data Access constants
12| '
13|
14| ' Option argument values (CreateDynaset, etc)
15| Global Const DB_DENYWRITE = &H1
16| Global Const DB_DENYREAD = &H2
17| Global Const DB_READONLY = &H4
18| Global Const DB_APPENDONLY = &H8
19| Global Const DB_INCONSISTENT = &H10
20| Global Const DB_CONSISTENT = &H20
21| Global Const DB_SQLPASSTHROUGH = &H40
22|
23| ' SetDataAccessOption
24| Global Const DB_OPTIONINIPATH = 1
25|
26| ' Field Attributes
27| Global Const DB_FIXEDFIELD = &H1
28| Global Const DB_VARIABLEFIELD = &H2
29| Global Const DB_AUTOINCRFIELD = &H10
30| Global Const DB_UPDATABLEFIELD = &H20
31|
32| ' Field Data Types
33| Global Const DB_BOOLEAN = 1
34| Global Const DB_BYTE = 2
35| Global Const DB_INTEGER = 3
36| Global Const DB_LONG = 4
37| Global Const DB_CURRENCY = 5
38| Global Const DB_SINGLE = 6
39| Global Const DB_DOUBLE = 7
40| Global Const DB_DATE = 8
41| Global Const DB_TEXT = 10
42| Global Const DB_LONGBINARY = 11
43| Global Const DB_MEMO = 12
44|
45| ' TableDef Attributes
46| Global Const DB_ATTACHEXCLUSIVE = &H10000
47| Global Const DB_ATTACHSAVEPWD = &H20000
48| Global Const DB_SYSTEMOBJECT = &H80000002
49| Global Const DB_ATTACHEDTABLE = &H40000000
50| Global Const DB_ATTACHEDODBC = &H20000000
51|
52| ' ListTables TableType
53| Global Const DB_TABLE = 1
54| Global Const DB_QUERYDEF = 5
```

```
55|
56| ' ListTables Attributes (for QueryDefs)
57| Global Const DB_QACTION = &HF0
58| Global Const DB_QCROSSTAB = &H10
59| Global Const DB_QDELETE = &H20
60| Global Const DB_QUPDATE = &H30
61| Global Const DB_QAPPEND = &H40
62| Global Const DB_QMAKETABLE = &H50
63|
64| ' ListIndexes IndexAttributes values
65| Global Const DB_UNIQUE = 1
66| Global Const DB_PRIMARY = 2
67| Global Const DB_PROHIBITNULL = 4
68| Global Const DB_IGNORENULL = 8
69| ' ListIndexes FieldAttributes value
70| Global Const DB_DESCENDING = 1  'For each field in Index
71|
72| ' CreateDatabase and CompactDatabase Language constants
73| Global Const DB_LANG_GENERAL = ";LANGID=0x0809;CP=1252;COUNTRY=0"
74| Global Const DB_LANG_SPANISH = ";LANGID=0x040A;CP=1252;COUNTRY=0"
75| Global Const DB_LANG_DUTCH = ";LANGID=0x0413;CP=1252;COUNTRY=0"
76| Global Const DB_LANG_SWEDFIN = ";LANGID=0x040C;CP=1252;COUNTRY=0"
    'VB3 and Access 1.1 Databases
77| Global Const DB_LANG_NORWDAN = ";LANGID=0x0414;CP=1252;COUNTRY=0"
    'VB3 and Access 1.1 Databases
78| Global Const DB_LANG_ICELANDIC = ";LANGID=0x040F;CP=1252;COUNTRY=0"
    'VB3 and Access 1.1 Databases
79| Global Const DB_LANG_NORDIC = ";LANGID=0x041D;CP=1252;COUNTRY=0"
    'Access 1.0 Databases only
80|
81| ' CreateDatabase and CompactDatabase options
82| Global Const DB_VERSION10 = 1       ' Microsoft Access Version 1.0
83| Global Const DB_ENCRYPT = 2         ' Make database encrypted.
84| Global Const DB_DECRYPT = 4         ' Decrypt database while compacting.
85|
86| 'Collating order values
87| Global Const DB_SORTGENERAL = 256   ' Sort by EFGPI rules (English, French,
    German,Portuguese, Italian)
88| Global Const DB_SORTSPANISH = 258   ' Sort by Spanish rules
89| Global Const DB_SORTDUTCH = 259     ' Sort by Dutch rules
90| Global Const DB_SORTSWEDFIN = 260   ' Sort by Swedish, Finnish rules
91| Global Const DB_SORTNORWDAN = 261   ' Sort by Norwegian, Danish rules
92| Global Const DB_SORTICELANDIC = 262 ' Sort by Icelandic rules
93| Global Const DB_SORTPDXINTL = 4096  ' Sort by Paradox international rules
94| Global Const DB_SORTPDXSWE = 4097   ' Sort by Paradox Swedish, Finnish rules
95| Global Const DB_SORTPDXNOR = 4098   ' Sort by Paradox Norwegian, Danish rules
96| Global Const DB_SORTUNDEFINED = -1  ' Sort rules are undefined or unknown
97| Global FldTypeIDNums(0 To 12) As String 'Holds the name of the datatypes
98|                         ' available in Access
99|
100|
101|
102|
103| Declare Function ShellExecute Lib "SHELL" (ByVal hwnd As Integer, ByVal lpszOp As
```

```
      String, ByVal lpszFile As String, ByVal lpszParams As String, ByVal lpszDir As String,
      ByVal fsShowCmd As Integer) As Integer
104   Declare Function GetDesktopWindow Lib "USER" () As Integer
105
106   Const SW_SHOWNORMAL = 1
107
108   'Parameter   Description
109   '
110   '-------------------------------------------------------------------------
111   'hwnd%       Identifies the parent window. This window receives any message
112   '           boxes an application produces (for example, for error
113   '           reporting).
114   '
115   'lpszOp$     Points to a null-terminated string specifying the operation to
116   '           perform. This string can be "open" or "print." If this
117   '           parameter is NULL, "open" is the default value.
118   '
119   'lpszFile$   Points to a null-terminated string specifying the file to open.
120   '
121   'lpszParams$ Points to a null-terminated string specifying parameters
122   '           passed to the application when the lpszFile parameter
123   '           specifies an executable file. If lpszFile points to a
124   '           string specifying a document file, this parameter is NULL.
125   '
126   'lpszDir$    Points to a null-terminated string specifying the default
127   '           directory.
128   '
129   'fsShowCmd%  Specifies whether the application window is to be shown when
130   '           the application is opened. This parameter can be one of the
131   '           values described in the API ShowWindow().
132
133   Sub AddAndHighlightItem (StringIn As String, CMB As ComboBox)
134    CMB.AddItem StringIn
135    CMB.ListIndex = CMB.ListCount - 1
136   End Sub
137
138   Function AddBracketsAsNeeded (NameIn As String) As String
139    If (InStr(NameIn, " ") <> 0 And Not InStr(NameIn, "[")) Then
140       AddBracketsAsNeeded = "[" & NameIn & "]"
141      Else
142       AddBracketsAsNeeded = NameIn
143    End If
144   End Function
145
146   Function AddVarNameAsNeeded (ByVal NameIn As String, ByVal AsVariableName As
      String) As String
147    Dim Temp As String, iWasField As Integer
148    Temp = Trim$(NameIn)
149    If Left$(Temp, 1) = "[" Then
150     Temp = Mid$(Temp, 2, Len(Temp) - 2)
151     iWasField = True
152    Else
153     iWasField = False
154    End If
```

```
155|
156|  If Temp = AsVariableName Then
157|    AddVarNameAsNeeded = AsVariableName
158|   Else
159|    AddVarNameAsNeeded = NameIn & " AS " & AsVariableName
160|  End If
161|
162| End Function
163|
164| Function And2 (I1 As Integer, I2 As Integer) As Integer
165|   If (I1 And I2) <> 0 Then
166|     And2 = True
167|    Else
168|     And2 = False
169|   End If
170| End Function
171|
172| Function AND4 (I1 As Long, I2 As Long) As Long
173|   If (I1 And I2) <> 0 Then
174|     AND4 = True
175|    Else
176|     AND4 = False
177|   End If
178| End Function
179|
180| Sub ClearGrid (Target As Grid)
181| Dim nFixedRows As Long, nFixedCols As Long, nCols As Long
182| nFixedRows = Target.FixedRows
183| nFixedCols = Target.FixedCols
184| nCols = Target.Cols
185|  Target.FixedRows = 0
186|  Target.FixedCols = 0
187|  Target.Rows = 1
188|  Target.Cols = 1
189|  Target = ""
190|  Target.Rows = nFixedRows + 1
191|  Target.FixedRows = nFixedRows
192|  Target.Cols = nFixedCols + 1
193|  Target.FixedCols = nFixedCols
194|  Target.Cols = nCols
195| End Sub
196|
197| Sub ClipTextToXYPair (strClipIn As String, XOut As Single, YOut As Single)
198| Dim n As Long, Temp As String, ThisWord As String
199| On Error Resume Next
200| 'First, replace all spaces and tabs and such with spaces
201| Temp = strClipIn
202| For n = 1 To Len(Temp)
203|   If Asc(Mid$(Temp, n, 1)) < 33 Then Mid$(Temp, n, 1) = " "
204| Next n
205|
206|   ThisWord = GetNextWord(Temp)
207|   XOut = Val(ThisWord)
208|   Temp = Trim$(GetRestOfStringAfter(Temp, ThisWord))
```

```
    ThisWord = GetNextWord(Temp)
    YOut = Val(ThisWord)
End Sub

Function CountOccurrences (ByVal MamaString As String, ByVal BabyString As String)
    As Long
Dim n As Long
Dim BabyLength As Long
Dim NextBabyAt As Long
BabyLength = Len(BabyString)

If BabyLength < 1 Then Exit Function

Dim CurrentPos As Long
CurrentPos = 1
n = 0
NextBabyAt = InStr(CurrentPos, MamaString, BabyString, 1)
Do Until NextBabyAt = 0
 n = n + 1
 CurrentPos = NextBabyAt + BabyLength
 NextBabyAt = InStr(CurrentPos, MamaString, BabyString, 1)
Loop
CountOccurrences = n
End Function

Sub CreateTable (strName As String)
'GWPouch940807 On Error GoTo Err_CreateTable
'GWPouch940807 Dim Temp As String, Prompt As String, Title As String
'GWPouch940807 Dim NewTable As New TableDef, fldBLOBXY As New Field,
    fldLineID As New Field, fldSerial As New Field
'GWPouch940807 Dim fldMinX As New Field, fldMinY As New Field, fldMaxX As New
    Field, fldMaxY As New Field
'GWPouch940807 Dim fldNPoints As New Field
'GWPouch940807 Dim MinX As Single, MinY As Single, MaxX As Single, MaxY As Single
'GWPouch940807
'GWPouch940807
'GWPouch940807    strName = Temp
'GWPouch940807    NewTable.Name = strName 'Name the table
'GWPouch940807 'Define the fields
'GWPouch940807    fldBLOBXY.Name = "BLOBXY"
'GWPouch940807    fldBLOBXY.Type = 12'OLE/LongBinary object
'GWPouch940807
'GWPouch940807    fldLineID.Name = "LineIDNumber"
'GWPouch940807    fldLineID.Type = 4'Long Integer aka Integer*4 aka LONG aka long
    int
'GWPouch940807
'GWPouch940807    fldSerial.Type = 4
'GWPouch940807    fldSerial.Name = "SerialNumber"
'GWPouch940807
'GWPouch940807    fldMinX.Name = "MinX"
'GWPouch940807    fldMinX.Type = 6
'GWPouch940807
'GWPouch940807    fldMinY.Name = "MinY"
'GWPouch940807    fldMinY.Type = 6
```

```
'GWPouch940807
'GWPouch940807    fldMaxX.Name = "MaxX"
'GWPouch940807    fldMaxX.Type = 6
'GWPouch940807
'GWPouch940807    fldMaxY.Name = "MaxY"
'GWPouch940807    fldMaxY.Type = 6
'GWPouch940807
'GWPouch940807    fldNPoints.Name = "nPoints"
'GWPouch940807    fldNPoints.Type = 4
'GWPouch940807
'GWPouch940807
'GWPouch940807 'Append the fields to the table
'GWPouch940807
'GWPouch940807
'GWPouch940807    NewTable.Fields.Append fldSerial
'GWPouch940807    NewTable.Fields.Append fldLineID
'GWPouch940807    NewTable.Fields.Append fldBLOBXY
'GWPouch940807    NewTable.Fields.Append fldMinX
'GWPouch940807    NewTable.Fields.Append fldMaxX
'GWPouch940807    NewTable.Fields.Append fldMinY
'GWPouch940807    NewTable.Fields.Append fldMaxY
'GWPouch940807    NewTable.Fields.Append fldNPoints
'GWPouch940807
'GWPouch940807 ' Append the table to the table defs collection.
'GWPouch940807    MapDB.TableDefs.Append NewTable
'GWPouch940807    'txtTableTo = NewTable.Name
'GWPouch940807
'GWPouch940807 Exit Sub
'GWPouch940807 Err_CreateTable:
'GWPouch940807 If Err = 616 Or Err = 3010 Then 'Table already exists, append not
allowed
'GWPouch940807    NewTable.Name = strName & Format$(Now)
'GWPouch940807    Resume
'GWPouch940807  Else
'GWPouch940807  MsgBox "Error number " & Err & " occurred while trying to create a
table in DBUTIL01.BAS." & CRLF & Error
'GWPouch940807   Resume Next
'GWPouch940807 End If
End Sub

Function DCur (XIn As Variant) As Currency
 Dim Temp As Currency
 On Error GoTo ErrDCur
 Temp = XIn
 DCur = Temp
Exit Function
ErrDCur:
Select Case Err
  Case 11 'Division by zero
    Temp = XDiv0
  Case 6 'Overflow
    Temp = XDiv0
  Case 94 'Invalid use of null
    Temp = XNull
```

```
End Select
Resume Next
End Function

Function DDbl (XIn As Variant) As Double
 Dim Temp As Double
 On Error GoTo ErrDDbl
 Temp = XIn
 DDbl = Temp
Exit Function
ErrDDbl:
Select Case Err
  Case 11 'Division by zero
    Temp = XDiv0
  Case 6 'Overflow
    Temp = XDiv0
  Case 94 'Invalid use of null
    Temp = XNull
End Select
Resume Next
End Function

Function DInt (XIn As Variant) As Integer
 Dim Temp As Integer
 On Error GoTo ErrDInt
 Temp = XIn
 DInt = Temp
Exit Function
ErrDInt:
Select Case Err
  Case 11 'Division by zero
    Temp = XDiv0
  Case 6 'Overflow
    Temp = XDiv0
  Case 94 'Invalid use of null
    Temp = XNull
End Select
Resume Next
End Function

Sub DlmtdStngsToComboBox (Delimiter As String, StringIn As String, ComboOut As
ComboBox)
Dim nStrings As Long, n As Long
nStrings = CountOccurrences(StringIn, Delimiter)
For n = 1 To nStrings
   ComboOut.AddItem ExtractDelimitedString(StringIn, Delimiter, n)
Next n

End Sub

Function DLng (XIn As Variant) As Long
 Dim Temp As Long
 On Error GoTo ErrDLng
 Temp = XIn
```

```
364|  DLng = Temp
365|  Exit Function
366|  ErrDLng:
367|  Select Case Err
368|   Case 11 'Division by zero
369|     Temp = XDiv0
370|   Case 6 'Overflow
371|     Temp = XDiv0
372|   Case 94 'Invalid use of null
373|     Temp = XNull
374|  End Select
375|  Resume Next
376|  End Function
377|
378|  Function DSng (XIn As Variant) As Single
379|   Dim Temp As Single
380|   On Error GoTo ErrDSng
381|   Temp = XIn
382|   DSng = Temp
383|  Exit Function
384|  ErrDSng:
385|  Select Case Err
386|   Case 11 'Division by zero
387|     Temp = XDiv0
388|   Case 6 'Overflow
389|     Temp = XDiv0
390|   Case 94 'Invalid use of null
391|     Temp = XNull
392|  End Select
393|  Resume Next
394|  End Function
395|
396|  Function DStr (VarIn As Variant) As String
397|  Dim Temp As String
398|  On Error GoTo ErrDStr
399|  Temp = VarIn
400|  DStr = Temp
401|  ErrDStr:
402|   Temp = strNULL
403|  Resume Next
404|  End Function
405|
406|  Function DVar (FieldIn As Field) As Variant
407|  Static FieldType As Integer
408|  FieldType = FieldIn.Type
409|  Select Case FieldType
410|   Case 1 To 3 'Boolean, Byte, and Integer*2
411|    DVar = DInt(FieldIn)
412|   Case 4  'Integer*4
413|    DVar = DLng(FieldIn)
414|   Case 5  'Currency.-> double
415|    DVar = DDbl(FieldIn)
416|   Case 6 'real*4
417|    DVar = DSng(FieldIn)
```

```
418|  Case 7 'real*8
419|   DVar = DDbl(FieldIn)
420|  Case 8 ' Date/time.  Can only be stored in a variant
421|   DVar = FieldIn
422|  Case 10 'String
423|   DVar = DStr(FieldIn)
424| End Select
425| 'Note that this ignores types 11 (memo/BLOB) and 12 Long-binary(OLE object)
426| End Function
427|
428| Function ExtractDelimitedString (ByVal InString As String, ByVal Delimiter As String,
     ByVal nth As Long) As String
429| '  Pulls the nth Delimiter-delimited string out of InString.
430| 'Greg Pouch KGS 13 January 1994
431|
432|
433|  Dim Temp1 As String, ThisField As String
434|  Dim n As Long
435| Dim DelimLength As Long, NextDelimiter As Long
436|
437| If nth < 1 Then Exit Function
438| DelimLength = Len(Delimiter)
439| If DelimLength < 1 Then Exit Function
440| If CountOccurrences(InString, Delimiter) < nth Then Exit Function
441|
442| Temp1 = InString
443| n = 0
444| Do
445|    NextDelimiter = InStr(Temp1, Delimiter)
446|    ThisField = Mid$(Temp1, 1, NextDelimiter - 1)
447|    Temp1 = Mid$(Temp1, NextDelimiter + DelimLength)
448|    n = n + 1
449| Loop While (n < nth)
450| ExtractDelimitedString = ThisField
451| End Function
452|
453| Sub FillArrayWithDelimitedStrings (StringIn As String, Delimiter As String, strArrayOut()
     As String)
454| Dim nStrings As Long, n As Long
455| nStrings = CountOccurrences(StringIn, Delimiter)
456| ReDim strArrayOut(nStrings)
457| For n = 1 To nStrings
458|    strArrayOut(n) = ExtractDelimitedString(StringIn, Delimiter, n)
459| Next n
460|
461| End Sub
462|
463| Sub FillCMBFromSGSTLST (strSearchIn() As String, ByVal strLookFor As String, ByVal
     strShowAnyway As String, CMB As ComboBox)
464| 'Takes a bunch of strings, in strSearchIn(), and adds whichever of those contain strings
     in strLookFor to the CombBox.
465| 'It always adds the strings in strAddAnyway.
466| 'For instance, if strSearchIn contained New Jersey, Texas, New Mexico, X in UTM13, X in
     SPCS
```

```
467| ' and strLookFor contained "X", and strAddAnyway contained "3", "8/2", then
468| '  the combobox would end up containing Texas, New Mexico ,X in UTM13, "3", "8/2",
     and X in SPCS.
469| 'if strLookFor contained Jersery, Mexico, the combobox would get
470| ' New Jersey, New Mexico, "3", "8/2"
471| Dim n As Long, m As Long, l As Long, nLookFor As Long
472|
473| strLookFor = UCase$(strLookFor)
474| nLookFor = 0
475| ReDim LookFor(nLookFor) As String
476| FillArrayWithDelimitedStrings strLookFor, " ", LookFor()
477| nLookFor = UBound(LookFor)
478| ReDim Used(LBound(strSearchIn) To UBound(strSearchIn))  As Integer
479| Dim Temp As String
480| For m = 1 To nLookFor
481|    For n = LBound(strSearchIn) To UBound(strSearchIn)
482|       If Not Used(n) Then
483|        If InStr(strSearchIn(n), LookFor(m)) <> 0 Then
484|           CMB.AddItem strSearchIn(n)
485|           Used(n) = True
486|        End If
487|       End If
488|    Next n
489| Next m
490| 'Add the constant items
491| DlmtdStngsToComboBox " ", strShowAnyway, CMB
492|
493| End Sub
494|
495| Sub FillGridFromClipBoardText (TextIn As String, grdOut As Grid, flgHowTo As Long)
496| Const FG_GetFromClipboard = 1&
497| Const FG_OverWrite = 2&  'Ignored for now
498| Const FG_AutoExpandCols = 4&
499| Const FG_AppendRows = 8&
500| Const FG_Transpose = 16&  'Ignored for now
501|
502| On Error GoTo Err_FillGridFromClipboardText
503|
504| Dim nRows As Long, nCols As Long, nR As Long, nC As Long, n As Long
505| Dim Temp As String
506| If AND4(flgHowTo, FG_GetFromClipboard) Then TextIn = Clipboard.GetText()
507| Temp = TextIn
508|
509| 'Preserve information about initial state of the target grid control.
510| Dim gCols As Long, gRows As Long, gStartCol As Long, gStartRow As Long, gEndCol
     As Long, gEndRow As Long
511| gCols = grdOut.Cols
512| gRows = grdOut.Rows
513|
514| gStartCol = grdOut.SelStartCol
515| gStartRow = grdOut.SelStartRow
516| gEndCol = grdOut.SelEndCol
517| gEndRow = grdOut.SelEndRow
518|
```

```
'Find number of rows (delimeted by asc(13))
'Count the number of tabs and LineFeeds in the string
   For n = 1 To Len(Temp)
      If Mid$(Temp, n, 1) = TabChar Then
         nCols = nCols + 1
      End If
      If Mid$(Temp, n, 1) = CR Then
         nRows = nRows + 1
         nCols = nCols + 1
         If Mid$(Temp, n + 1, 1) = LF Then Mid$(Temp, n + 1, 1) = " "
      End If
   Next n

   If nCols = 0 Then
     nCols = 1
     Temp = Temp & TabChar
   End If
   If nRows = 0 Then
     nRows = 1
     Temp = Temp & CR
   End If
   nCols = nCols \ nRows

 Temp = Temp & TabChar & CR 'This avoids problems of too few characters.

'Dimension a string array of the correct size to hold this
ReDim GridText(nRows, nCols) As String
'Load the text into the array
For nR = 1 To nRows
   For nC = 1 To nCols - 1
      GridText(nR, nC) = Left$(Temp, InStr(Temp, TabChar) - 1)
      ""GridText(nR, nC) = Trim$(Left$(Temp, InStr(Temp, TabChar) - 1))
      'The immediately preceding line will help line things up, but takes longer.
      'Probably best to worry about trimming spaces only on saving the data.

      Temp = Mid$(Temp, InStr(Temp, TabChar) + 1)
   Next nC
   GridText(nR, nCols) = Left$(Temp, InStr(Temp, CR) - 1)
   Temp = Mid$(Temp, InStr(Temp, CR) + 1)

Next nR
'If flgTranspose Then Call TransposeText2DArray(GridText())
    'Load the text into the grid
If AND4(flgHowTo, FG_AppendRows) Then
   For n = 1 To nRows
      grdOut.Row = grdOut.Rows - 1
      grdOut.AddItem ""
   Next n
gStartRow = gRows 'Start after last of old rows.
grdOut.Col = gStartCol
End If

If AND4(flgHowTo, FG_AutoExpandCols) Then
   If gCols < gStartCol + nCols Then grdOut.Cols = gStartCol + nCols
```

```
573|  End If
574|
575|  'Put the text into the target grid.
576|  For nR = 0 To nRows - 1
577|   grdOut.Row = gStartRow + nR
578|   For nC = 0 To nCols - 1
579|     grdOut.Col = gStartCol + nC
580|
581|     grdOut = GridText(nR + 1, nC + 1)
582|   Next nC
583|  Next nR
584|
585|
586|
587|  Exit Sub
588|  Err_FillGridFromClipboardText:
589|  Err = 0
590|  MsgBox "Could not successfully complete the requested pasting operation."
591|  Exit Sub
592|  Resume Next
593|
594|  End Sub
595|
596|  Sub FillGridFromDynaset (grdOut As Grid, dynIn As Dynaset)
597|   'Unique to MapVuQry frmMap.lblPercentDone = "Filling grid with Data from Dynaset"
598|
599|    'Unique to MapVuQry frmMap.pnl3dPercentDone.FloodPercent = 0
600|
601|  Dim nRecs As Long, nRec As Long, nFields  As Long, nFld As Long
602|  On Error GoTo ErrFillGridFromDynaset
603|    dynIn.MoveFirst
604|    dynIn.MoveLast
605|    dynIn.MoveFirst
606|    nRecs = dynIn.RecordCount
607|    nFields = dynIn.Fields.Count
608|  If grdOut.Rows < nRecs + 1 Then grdOut.Rows = nRecs + 1
609|  If grdOut.Cols < nFields + 1 Then grdOut.Cols = nFields + 1
610|  For nRec = 1 To nRecs
611|     grdOut.Row = nRec
612|     For nFld = 1 To nFields
613|        grdOut.Col = nFld
614|          'If dynIn.Fields(nFld - 1).Type < 11 Then
615|        grdOut = Format(dynIn(nFld - 1))
616|          'end if
617|     Next nFld
618|     dynIn.MoveNext
619|     'Unique to MapVuQry frmMap.pnl3dPercentDone.FloodPercent = nRec / nRecs * 100
620|  Next nRec
621|  'Unique to MapVuQry frmMap.pnl3dPercentDone.FloodPercent = 100
622|  'Unique to MapVuQry frmMap.lblPercentDone = "Done filling grid"
623|  ErrFillGridFromDynaset:
624|  If Err = 3021 Then 'nothing in dynaset
625|     grdOut.Rows = 1
626|     grdOut.Rows = 2
```

```
627I    grdOut.FixedRows = 1
628I    Resume Next
629I End If
630I End Sub
631I
632I Function GetClipboardDataTypesFromDynaset (dynIn As Dynaset) As String
633I Dim nFields As Long, nF As Long, Temp As String, nType As Long
634I
635I nFields = dynIn.Fields.Count
636I For nF = 0 To nFields - 1
637I    nType = dynIn.Fields(nF).Type
638I    Temp = Temp & Format$(nType, "00") & "  " & FldTypeIDNums(nType) & TabChar
639I Next nF
640I                              'Trim the trailing tab.
641I GetClipboardDataTypesFromDynaset = Left$(Temp, Len(Temp) - 1) & CRLF
642I                              'Left$(Temp, Len(Temp) - 1)
643I End Function
644I
645I Function GetClipboardLabelsFromDynaset (dynIn As Dynaset) As String
646I Dim nFields As Long, nF As Long, Temp As String
647I nFields = dynIn.Fields.Count
648I For nF = 0 To nFields - 1
649I    Temp = Temp & dynIn.Fields(nF).Name & Format$(dynIn.Fields(nF).Type, "  00") &
     Chr$(9)
650I Next nF
651I
652I GetClipboardLabelsFromDynaset = Left$(Temp, Len(Temp) - 1) & CRLF
653I End Function
654I
655I Function GetClipBoardTextFromDynaset (dynIn As Dynaset, flgOptions As Long) As
     String
656I Dim nRecs As Long, nRec As Long, nFields  As Long, nFld As Long
657I Dim Temp As String
658I On Error GoTo Err_ClipboardTextFromDynaset
659I  dynIn.MoveFirst
660I  dynIn.MoveLast
661I  dynIn.MoveFirst
662I  nRecs = dynIn.RecordCount
663I  nFields = dynIn.Fields.Count
664I Temp = ""
665I If flgOptions And 1 Then Temp = GetClipboardLabelsFromDynaset(dynIn) & CRLF
666I
667I For nRec = 1 To nRecs
668I    For nFld = 0 To nFields - 1
669I       If dynIn(nFld).Type < 11 Then
670I         Temp = Temp & dynIn(nFld) & Chr$(9)
671I        Else
672I         Temp = Temp & Chr$(9)
673I       End If
674I    Next nFld
675I    Temp = Left$(Temp, Len(Temp) - 1) & CRLF
676I    dynIn.MoveNext
677I Next nRec
678I GetClipBoardTextFromDynaset = Temp
```

```
679| Exit Function
680| Err_ClipboardTextFromDynaset:
681| If Err = 3021 Then 'nothing in dynaset
682|    Resume Next
683| End If
684|
685| End Function
686|
687| Function GetFieldForOrderBy (ByVal StringIn As String) As String
688|  StringIn = Trim$(StringIn)
689|  Select Case Left$(StringIn, 1)
690|     Case "["
691|      GetFieldForOrderBy = Left$(StringIn, InStr(StringIn, "]"))
692|     Case Else
693|      GetFieldForOrderBy = ""
694|   End Select
695|
696| End Function
697|
698| Function GetGoodValueField (ByVal StringIn As String) As String
699|  StringIn = Trim$(StringIn)
700|  Select Case Left$(StringIn, 1)
701|     Case "["
702|      GetGoodValueField = Left$(StringIn, InStr(StringIn, "]"))
703|     Case Chr$(34)
704|      GetGoodValueField = Left$(StringIn, InStr(2, StringIn, Chr$(34)))
705|     Case Else
706|      GetGoodValueField = Format(Val(StringIn))
707|   End Select
708| End Function
709|
710| Function GetIndexOfStringInCOMBOBOX (ByVal StringIn As String, COMBOLIST As
     ComboBox) As Long
711| Dim nItems As Long, nI As Long, Answer As Long, StringImLookingFor As String
712| StringImLookingFor = UCase$(StringIn)
713| nItems = COMBOLIST.ListCount
714| Answer = -1
715| For nI = 0 To nItems - 1
716|  If UCase$(COMBOLIST.List(nI)) = StringImLookingFor Then
717|    Answer = nI
718|    Exit For
719|  End If
720| Next nI
721| GetIndexOfStringInCOMBOBOX = Answer
722| End Function
723|
724| Function GetIndexOfSubStringInCOMBOBOX (ByVal StringIn As String, COMBOLIST As
     ComboBox) As Long
725| Dim nItems As Long, nI As Long, Answer As Long, StringImLookingFor As String
726| StringImLookingFor = UCase$(StringIn)
727| nItems = COMBOLIST.ListCount
728| Answer = -1
729| For nI = 0 To nItems - 1
730|  If InStr(UCase$(COMBOLIST.List(nI)), StringImLookingFor) <> 0 Then
```

```
731|     Answer = nl
732|     Exit For
733|   End If
734| Next nl
735| GetIndexOfSubStringInCOMBOBOX = Answer
736|
737| End Function
738|
739| Sub GetMeanStdDev (FldName As String, TblQry As String, Criterion As String, Mean As
     Single, StdDev As Single, DB As Database)
740| Dim SSQQLL As String, snpStatsOnOneVariable As Snapshot
741| Dim LeftBracket As Integer
742| Dim RightBracket As Integer
743| Dim Fld As String
744| Fld = FldName
745|
746|
747| SSQQLL = "SELECT  DISTINCTROW Avg(" & Fld & ") AS Mean, "
748| SSQQLL = SSQQLL & "StDev(" & Fld & ") AS StandardDeviation "
749| SSQQLL = SSQQLL & "FROM " & AddBracketsAsNeeded(CStr(TblQry))
750| If (Criterion <> "") Then SSQQLL = SSQQLL & " WHERE " & Criterion
751| SSQQLL = SSQQLL & ";"
752|
753| Set snpStatsOnOneVariable = DB.CreateSnapshot(SSQQLL)
754| snpStatsOnOneVariable.MoveFirst
755| Mean = snpStatsOnOneVariable("Mean")
756| StdDev = snpStatsOnOneVariable("StandardDeviation")
757| snpStatsOnOneVariable.Close : Set snpStatsOnOneVariable = Nothing
758| End Sub
759|
760| Sub GetMinMax (FldName As String, TblQry As String, Criterion As String, Min As Single,
     Max As Single, DB As Database)
761| Dim SSQQLL As String
762| Dim LeftBracket As Integer
763| Dim RightBracket As Integer
764| Dim Fld As String
765| Dim snpStatsOnOneVariable As Snapshot
766| Fld = FldName
767|
768| SSQQLL = "SELECT  DISTINCTROW "
769| SSQQLL = SSQQLL & "Min(" & Fld & ") AS Min, "
770| SSQQLL = SSQQLL & "Max(" & Fld & ") AS Max "
771| SSQQLL = SSQQLL & "FROM " & AddBracketsAsNeeded(CStr(TblQry))
772| If (Criterion <> "") Then SSQQLL = SSQQLL & " WHERE " & Criterion
773| SSQQLL = SSQQLL & ";"
774|
775| Set snpStatsOnOneVariable = DB.CreateSnapshot(SSQQLL)
776| snpStatsOnOneVariable.MoveFirst
777| Min = snpStatsOnOneVariable("Min")
778| Max = snpStatsOnOneVariable("Max")
779| snpStatsOnOneVariable.Close : Set snpStatsOnOneVariable = Nothing
780|
781| End Sub
782|
```

```
Function GetstrNULL () As String
GetstrNULL = strNULL
End Function

Function GetXDiv0 () As Integer
GetXDiv0 = XDiv0
End Function

Function GetXNull () As Integer
GetXNull = XNull
End Function

Function IndexOfStringInCOMBOBOX (ByVal StringIn As String, COMBOLIST As
ComboBox) As Long

' i=IndexOfStringInCOMBOBOX ( StringIn , COMBOLIST)
Dim nItems As Long, nl As Long, Answer As Long, StringImLookingFor As String
StringImLookingFor = UCase$(StringIn)
nItems = COMBOLIST.ListCount
Answer = -1
For nl = 0 To nItems - 1
  If UCase$(COMBOLIST.List(nl)) = StringImLookingFor Then
    Answer = nl
    Exit For
  End If
Next nl
IndexOfStringInCOMBOBOX = Answer
End Function

Function IndexOfSubStringInCOMBOBOX (ByVal StringIn As String, COMBOLIST As
ComboBox) As Long
'IndexOfSubStringInCOMBOBOX ( StringIn, COMBOLIST )
Dim nItems As Long, nl As Long, Answer As Long, StringImLookingFor As String
StringImLookingFor = UCase$(StringIn)
nItems = COMBOLIST.ListCount
Answer = -1
For nl = 0 To nItems - 1
  If InStr(UCase$(COMBOLIST.List(nl)), StringImLookingFor) <> 0 Then
    Answer = nl
    Exit For
  End If
Next nl
IndexOfSubStringInCOMBOBOX = Answer

End Function

Sub InitializeGridForDynaset (Target As Grid, dynIn As Dynaset)
   Dim ColumnLabel As String
   Target.Cols = 2
'First, add the field names as column labels.
   Target.FixedRows = 0
   Call SetGridFocusRow(Target, 0)
   Call SetGridFocusColumn(Target, 1)
   ColumnLabel = GetClipboardLabelsFromDynaset(dynIn)
```

```
835|          'ColumnLabel = Left(ColumnLabel, Len(ColumnLabel) - 2) & TabChar & "X
    LastPoint" & TabChar & "Y LastPoint" & TabChar & "Date/Time" & CRLF
836|    Call FillGridFromClipBoardText(ColumnLabel, Target, 4&)
837|    Call SetGridCellText(Target, 0, 0, "Field Name")
838|    Target.FixedRows = 1
839| 'GWPouch 940722 ' Next add the data types in the second row of the grid.
840| 'GWPouch 940722    Call SetGridFocusRow(Target, 1)
841| 'GWPouch 940722    ColumnLabel = GetClipboardDataTypesFromDynaset(dynIn)
842| 'GWPouch 940722          'ColumnLabel = Left(ColumnLabel, Len(ColumnLabel) - 2) &
    TabChar & "06 Real*4" & TabChar & "06 Real*4" & TabChar & "08 Date/Time" & CRLF
843| 'GWPouch 940722    Call SetGridFocusColumn(Target, 1)
844| 'GWPouch 940722    Call FillGridFromClipBoardText(ColumnLabel, Target, 4&)
845| 'GWPouch 940722    Call SetGridCellText(Target, 0, 1, "Field Type")
846|
847| End Sub
848|
849| Sub LabelGridFromDynaset (grdOut As Grid, dynIn As Dynaset)
850| Dim nFields  As Long, nFld As Long
851|  dynIn.MoveFirst
852|  nFields = dynIn.Fields.Count
853|
854| If grdOut.Cols < nFields + 1 Then grdOut.Cols = nFields + 1
855| grdOut.Row = 0
856|    For nFld = 1 To nFields
857|       grdOut.Col = nFld
858|        'If dynIn.Fields(nFld - 1).Type < 11 Then
859|       grdOut = dynIn.Fields(nFld - 1).Name
860|        'end if
861|    Next nFld
862| End Sub
863|
864| Sub ListFlds (DBIn As Database, QTName As String, FieldList() As String)
865| Dim snpFieldList As Snapshot
866| Set snpFieldList = DBIn.CreateSnapshot(QTName)
867| 'snpFieldList.MoveFirst
868| Dim nFields As Long, i As Long
869| nFields = snpFieldList.Fields.Count
870| ReDim FieldList(nFields)
871| For i = 1 To nFields
872|    FieldList(i) = snpFieldList(i - 1).Name
873|
874| Next i
875| End Sub
876|
877| Sub ListTblQrysInComboBox (DBIn As Database, ComboOut As ComboBox)
878| ' this subroutine lists all queries and tables in a database
879| '  in a combobox, for later user selection.
880|    ComboOut.Clear
881|    Dim DB_SYSTEMOBJECT As Long
882|    DB_SYSTEMOBJECT = &H80000002
883|
884|    Dim snpQTList As Snapshot 'declared locally so it goes away
885|    Set snpQTList = DBIn.ListTables()
886|
```

```
887|    Dim QTName As String
888|    snpQTList.MoveFirst
889|    Do Until snpQTList.EOF 'For each Table and query
890|       If Not ((snpQTList("Attributes") And DB_SYSTEMOBJECT) <> 0) And
      (snpQTList("Attributes") < 6) Then
891|         'Exclude system objects from list of tables and queries presented to user
892|         QTName = snpQTList("Name")
893|         ComboOut.AddItem QTName
894|       End If
895|       snpQTList.MoveNext
896|    Loop
897|    snpQTList.Close
898|    Set snpQTList = Nothing
899| End Sub
900|
901| Sub ListTblQrysInStringArray (DBIn As Database, QTList() As String,
      flgNoParameterQueries As Integer)
902| Erase QTList
903| On Error GoTo ListTblQrysErr
904| Dim snpQTList As Snapshot, snpTemp As Snapshot
905| Set snpQTList = DBIn.ListTables()
906| snpQTList.MoveFirst
907| snpQTList.MoveLast
908| Dim nNonSys As Long, flgHadParameters As Integer, i As Long
909| nNonSys = 0
910| 'GWPouch 940807 Unique to MapVuQry
911|   Dim nTblQrys As Long 'Dangerous local redifinition. xxx See if this is used elsewhere
912| nTblQrys = snpQTList.RecordCount
913| snpQTList.MoveFirst
914|
915| For i = 0 To nTblQrys - 1
916|
917|  'If Not ((snpQTList("Attributes") And DB_SYSTEMOBJECT) <> 0) Then
918|  If snpQTList("Attributes") = 0 Then 'Apparently, tables and regular queries will
919|                                      ' have an attribute value of 0. External tables????
920|                                      ' This does get rid of action queries and system tables.
921|   'Exclude system objects from list of tables and queries presented to user
922|   flgHadParameters = False
923|   If flgNoParameterQueries And (snpQTList("TableType") = DB_QUERYDEF) Then
924|     Set snpTemp = DBIn.CreateSnapshot(snpQTList("Name"))
925|     ' This attempts to create a snapshot from the querydef.  It will fail
926|     '  if there were no parameters supplied to a parameter query or if
927|     '  there was an action query.  Kind of slow way of finding out though.
928|     If flgHadParameters Then GoTo SkipThisOne
929|   End If
930|   nNonSys = nNonSys + 1
931|   ReDim Preserve QTList(nNonSys)
932|   QTList(nNonSys) = snpQTList("Name")
933| SkipThisOne:
934|  End If
935|  snpQTList.MoveNext
936| Next i
937| nTblQrys = nNonSys
938| 'After the above loop finishes, QTList(1) to (nNonSys) contain
```

```
939| ' the names of all the non-system tables in the database
940| 'nTblQrys now contains the number of non-system tables and queries.
941| snpQTList.Close
942| Set snpQTList = Nothing
943|
944| Exit Sub
945|
946| ListTblQrysErr:
947| 'dDebug.Print Err, nNonSys
948| 'Select Case Err
949| '   Case 3061
950| '      flgHadParameters = True
951| '      Resume Next
952| '   Case Else
953| '      Resume Next
954| 'End Select
955| 'Rather a broad-shot method, but any query/table that can't be opened will
956| ' cause an explosion.
957| 'dDebug.Print "in error handler    ", snpQTList("Name"), snpQTList("attributes"), Err
958| flgHadParameters = True
959| Resume Next
960|
961|
962| End Sub
963|
964| Sub OpenQSFiles (dnsIn As Dynaset)
965|    'Kluge 950609 GWPouch at KGS
966|    ' Added code to allow E-Map to open and run a file every time
967|    ' the user selects a feature with a spatial query.  The file
968|    ' to open must be in a field called OnClickOpenFile, and should
969|    ' be a filename with an extension recognized by file manager.
970| Dim n As Long, nOpenFile As Long, strFileToOpen As String
971| nOpenFile = -1
972|   For n = 0 To dnsIn.Fields.Count - 1
973|      If UCase$(dnsIn.Fields(n).Name) = "ONCLICKOPENFILE" Then
974|        nOpenFile = n
975|        Exit For
976|      End If
977|   Next n
978|
979| If nOpenFile = -1 Then Exit Sub
980|
981| On Error Resume Next
982| dnsIn.MoveFirst
983|
984|
985|     Dim Scr_hWnd As Integer
986|     Scr_hWnd = GetDesktopWindow()
987| Do
988|  strFileToOpen = Trim$(dnsIn(nOpenFile))
989|  n = ShellExecute(Scr_hWnd, "Open", strFileToOpen, "", "C:\", SW_SHOWNORMAL)
990|  dnsIn.MoveNext
991| Loop Until dnsIn.EOF
992|
```

```
 993|
 994| End Sub
 995|
 996| Sub OverwriteGridFromDynaset (grdOut As Grid, dynIn As Dynaset)
 997|   'GWPouch 940807 Unique to MapVuQry frmMap.lblPercentDone = "Filling grid with
      Data from Dynaset"
 998|
 999|   'GWPouch 940807 Unique to MapVuQry frmMap.pnl3dPercentDone.FloodPercent = 0
1000|
1001| Dim nRecs As Long, nRec As Long, nFields  As Long, nFld As Long
1002| On Error GoTo ErrOverwriteGridFromDynaset
1003|   dynIn.MoveFirst
1004|   dynIn.MoveLast
1005|   dynIn.MoveFirst
1006|   nRecs = dynIn.RecordCount
1007|   nFields = dynIn.Fields.Count
1008| If grdOut.Rows < nRecs + 1 Then grdOut.Rows = nRecs + 1
1009| If grdOut.Cols < nFields + 1 Then grdOut.Cols = nFields + 1
1010| For nRec = 1 To nRecs
1011|     grdOut.Row = nRec
1012|     For nFld = 1 To nFields
1013|        grdOut.Col = nFld
1014|          'If dynIn.Fields(nFld - 1).Type < 11 Then
1015|        grdOut = Format(dynIn(nFld - 1))
1016|          'end if
1017|     Next nFld
1018|     dynIn.MoveNext
1019|     'GWPouch 940807 Unique to MapVuQry frmMap.pnl3dPercentDone.FloodPercent =
      nRec / nRecs * 100
1020| Next nRec
1021| 'GWPouch 940807 Unique to MapVuQry frmMap.pnl3dPercentDone.FloodPercent =
      100
1022| 'GWPouch 940807 Unique to MapVuQry frmMap.lblPercentDone = "Done filling grid"
1023| ErrOverwriteGridFromDynaset:
1024| If Err = 3021 Then 'nothing in dynaset
1025|     grdOut.Rows = 1
1026|     grdOut.Rows = 2
1027|     grdOut.FixedRows = 1
1028|     Resume Next
1029| End If
1030|
1031| End Sub
1032|
1033| Function ReplaceString (InString As String, strNowIs As String, strShouldBe As String)
      As String
1034| ' OutString=ReplaceString (InString , strNowIs , strShouldBe )
1035|   Dim where As Long
1036|   Dim Temp As String
1037|   where = InStr(InString, strNowIs)
1038|  If (where <> 0) Then
1039|     Temp = Left$(InString, where - 1) & strShouldBe & Mid$(InString, where +
      Len(strNowIs))
1040|     ReplaceString = ReplaceString(Temp, strNowIs, strShouldBe)
1041|   Else
```

```
1042|     ReplaceString = InString
1043|  End If
1044|  End Function
1045|
1046|  Sub SetGridCellText (TargetGrid As Grid, ColNum As Long, RowNum As Long,
       WhatToWrite As String)
1047|  On Error GoTo ERRSetGridCellText
1048|   TargetGrid.Col = ColNum
1049|   TargetGrid.Row = RowNum
1050|   TargetGrid = WhatToWrite
1051|
1052|
1053|  Exit Sub
1054|  ERRSetGridCellText:
1055|  Err = 0
1056|  Exit Sub
1057|  Resume
1058|  End Sub
1059|
1060|  Sub SetGridFocusColumn (TargetGrid As Grid, Column As Long)
1061|    TargetGrid.Col = Column
1062|    TargetGrid.SelStartCol = Column
1063|    TargetGrid.SelEndCol = Column
1064|  End Sub
1065|
1066|  Sub SetGridFocusRow (TargetGrid As Grid, Row As Long)
1067|    TargetGrid.Row = Row
1068|    TargetGrid.SelStartRow = Row
1069|    TargetGrid.SelEndRow = Row
1070|  End Sub
1071|
1072|  Sub SetstrNULL (NewstrNULL As String)
1073|   strNULL = NewstrNULL
1074|  End Sub
1075|
1076|  Sub SetXDiv0 (NewXDiv0 As Integer)
1077|   XDiv0 = NewXDiv0
1078|  End Sub
1079|
1080|  Sub SetXNull (NewXNull As Integer)
1081|   XNull = NewXNull
1082|  End Sub
1083|
1084|  Function StartDoc (DocName As String) As Integer
1085|      Dim Scr_hWnd As Integer
1086|      Scr_hWnd = GetDesktopWindow()
1087|      StartDoc = ShellExecute(Scr_hWnd, "Open", DocName, "", "C:\",
       SW_SHOWNORMAL)
1088|  End Function
1089|
1090|  Function Strip2Brackets (StringIn As String) As String
1091|   Dim Temp As String
1092|   Temp = Trim$(StringIn)
1093|   If Left$(Temp, 1) = "[" Then Temp = Mid$(Temp, 2)
```

```
1094|  If Right$(Temp, 1) = "]" Then Temp = Left$(Temp, Len(Temp) - 1)
1095|  Strip2Brackets = Temp
1096|  End Function
1097|
1098|  Function UCTRim16 (InString As String) As String
1099|  Dim Temp As String
1100|  If Len(InString) > 16 Then
1101|     Temp = UCase$(Left(InString, 16))
1102|    Else
1103|     Temp = UCase$(InString)
1104|  End If
1105|
1106|  Do While InStr(Temp, " ")
1107|    Mid$(Temp, InStr(Temp, " "), 1) = "_"
1108|  Loop
1109|  UCTRim16 = Temp
1110|  End Function
1111|
1112|  Sub WriteClipboardTextToDynaset (TextIn As String, dnsOut As Dynaset, flgOptions As
       Long)
1113|  Dim n As Long
1114|
1115|   'A probably inactive subroutine
1116|
1117|
1118|  'dnsOut.Edit
1119|  'For n = 1 To nFields
1120|  '
1121|  '    dnsOut.Fields(n - 1).Value = CVar("A") 'grdAttributes)
1122|  'Next n
1123|  '
1124|  'dnsOut.Update
1125|
1126|
1127|  End Sub
1128|
1129|  Sub ZAddDBLabeledColumn (Target As Grid, Title1 As String, Title2 As String)
1130|   Dim ThisCol As Long
1131|
1132|   Target.Cols = Target.Cols + 1
1133|   ThisCol = Target.Cols - 1
1134|   SetGridCellText Target, ThisCol, 0, Title1
1135|   SetGridCellText Target, ThisCol, 1, Title2
1136|
1137|  End Sub
1138|
1139|  Function ZAppendConstantTextToClipText (InClip As String, strAppendMe As String) As
       String
1140|  'This function is to be used to append data to the end of a set of clipboard style text.
1141|  ' A typical use would be to append the mouse coordinates and time of clicking onto the
       end
1142|  '  the results of a spatial query.
1143|  Dim n As Long, nRows As Long
1144|  Dim Temp As String
```

```
1145|
1146| nRows = CountOccurrences(InClip, Chr$(13))
1147| For n = 1 To nRows
1148|    Temp = Temp & ExtractDelimitedString(InClip, Chr$(13), n) & TabChar &
     strAppendMe & Chr$(13)
1149| Next n
1150| ZAppendConstantTextToClipText = Temp
1151| End Function
1152|
1153| Function ZCur (XIn As Variant, XNull As Currency, XDiv0 As Currency) As Currency
1154|  Dim Temp As Currency
1155|  On Error GoTo ErrZCur
1156|  Temp = XIn
1157|  ZCur = Temp
1158| Exit Function
1159| ErrZCur:
1160| Select Case Err
1161|  Case 11 'Division by zero
1162|    Temp = XDiv0
1163|  Case 6 'Overflow
1164|    Temp = XDiv0
1165|  Case 94 'Invalid use of null
1166|    Temp = XNull
1167| End Select
1168| Resume Next
1169| End Function
1170|
1171| Function ZDbl (XIn As Variant, XNull As Double, XDiv0 As Double) As Double
1172|  Dim Temp As Double
1173|  On Error GoTo ErrZDbl
1174|  Temp = XIn
1175|  ZDbl = Temp
1176| Exit Function
1177| ErrZDbl:
1178| Select Case Err
1179|  Case 11 'Division by zero
1180|    Temp = XDiv0
1181|  Case 6 'Overflow
1182|    Temp = XDiv0
1183|  Case 94 'Invalid use of null
1184|    Temp = XNull
1185| End Select
1186| Resume Next
1187| End Function
1188|
1189| Sub ZInitializeConstants ()
1190|    ZInitializeTextConstants
1191|    ZInitializeFldTypeList
1192|   ' ZInitializeSQLText xxxxx
1193|    Zero = 0
1194|    One = 1
1195|    Two = 2
1196| End Sub
1197|
```

```
1198| Sub ZInitializeFldTypeList ()
1199|  FldTypeIDNums(0) = "Undefined: Should be fixed in database"
1200|  FldTypeIDNums(1) = "Logical"       'Global Const DB_BOOLEAN = 1
1201|  FldTypeIDNums(2) = "Byte"          'Global Const DB_BYTE = 2
1202|  FldTypeIDNums(3) = "Integer*2"     'Global Const DB_INTEGER = 3
1203|  FldTypeIDNums(4) = "Integer*4"     'Global Const DB_LONG = 4
1204|  FldTypeIDNums(5) = "Integer*8"     'Global Const DB_CURRENCY = 5
1205|  FldTypeIDNums(6) = "Real*4"        'Global Const DB_SINGLE = 6
1206|  FldTypeIDNums(7) = "Real*8"        'Global Const DB_DOUBLE = 7
1207|  FldTypeIDNums(8) = "Date"          'Global Const DB_DATE = 8
1208|  FldTypeIDNums(9) = "Undefined: Should be fixed in database"
1209|  FldTypeIDNums(10) = "Text"         'Global Const DB_TEXT = 10
1210|  FldTypeIDNums(11) = "Long Binary"  'Global Const DB_LONGBINARY = 11
1211|  FldTypeIDNums(12) = "Memo"         'Global Const DB_MEMO = 12
1212|
1213|  ALLTypesAllowed(1) = DB_BOOLEAN
1214|  ALLTypesAllowed(2) = DB_BYTE
1215|  ALLTypesAllowed(3) = DB_INTEGER
1216|  ALLTypesAllowed(4) = DB_LONG
1217|  ALLTypesAllowed(5) = DB_CURRENCY
1218|  ALLTypesAllowed(6) = DB_SINGLE
1219|  ALLTypesAllowed(7) = DB_DOUBLE
1220|  ALLTypesAllowed(8) = DB_DATE
1221|
1222|
1223|
1224|
1225|
1226|
1227|
1228|
1229|
1230|
1231|
1232| End Sub
1233|
1234| Sub ZInitializeTextConstants ()
1235| ' The hex DA, ascii CR-LF used at ends of lines
1236|    CRLF = Chr$(13) & Chr$(10)
1237|    TabChar = Chr$(9)
1238|    DoubleQuote = Chr$(34)
1239|    SingleQuote = Chr$(39)
1240|    CR = Chr$(13)
1241|    LF = Chr$(10)
1242| End Sub
1243|
1244| Function ZInt (XIn As Variant, XNull As Integer, XDiv0 As Integer) As Integer
1245|  Dim Temp As Integer
1246|  On Error GoTo ErrZInt
1247|  Temp = XIn
1248|  ZInt = Temp
1249| Exit Function
1250| ErrZInt:
1251| Select Case Err
```

```
1252|   Case 11 'Division by zero
1253|     Temp = XDiv0
1254|   Case 6 'Overflow
1255|     Temp = XDiv0
1256|   Case 94 'Invalid use of null
1257|     Temp = XNull
1258| End Select
1259| Resume Next
1260|
1261| End Function
1262|
1263| Function ZLng (XIn As Variant, XNull As Long, XDiv0 As Long) As Long
1264|  Dim Temp As Long
1265|  On Error GoTo ErrZLng
1266|  Temp = XIn
1267|  ZLng = Temp
1268| Exit Function
1269| ErrZLng:
1270| Select Case Err
1271|   Case 11 'Division by zero
1272|     Temp = XDiv0
1273|   Case 6 'Overflow
1274|     Temp = XDiv0
1275|   Case 94 'Invalid use of null
1276|     Temp = XNull
1277| End Select
1278| Resume Next
1279|
1280| End Function
1281|
1282| Function ZSng (XIn As Variant, XNull As Single, XDiv0 As Single) As Single
1283|  Dim Temp As Single
1284|  On Error GoTo ErrZSng
1285|  Temp = XIn
1286|  ZSng = Temp
1287| Exit Function
1288| ErrZSng:
1289| Select Case Err
1290|   Case 11 'Division by zero
1291|     Temp = XDiv0
1292|   Case 6 'Overflow
1293|     Temp = XDiv0
1294|   Case 94 'Invalid use of null
1295|     Temp = XNull
1296| End Select
1297| Resume Next
1298| End Function
1299|
1300| Function ZStr (VarIn As Variant, strNULL As String)
1301| Dim Temp As String
1302| On Error GoTo ErrZStr
1303| Temp = VarIn
1304| ZStr = Temp
1305| ErrZStr:
```

```
1306| 'Select Case Error
1307| '    Case 94 'Invalid use of null
1308| '     Temp = strNULL
1309| '    Case 13 'Type mismatch
1310| '     Temp = strNULL
1311| 'End Select
1312| Temp = strNULL
1313| Resume Next
1314| End Function
1315|
1316| Sub ZTRunParameterQuery (OriginalQuery As QueryDef, DB As Database, snpResults
      As Snapshot, FinalSQLStatement As String)
1317|  Dim OriginalQuerySQL As String
1318|  Dim NewQuerySQL As String
1319|
1320|  Dim Junk2 As Snapshot
1321|  Dim ParamName As String
1322|  Dim UserPrompt As String
1323|  Dim ParamChosen As String
1324|  Dim Temp1 As Long
1325|  Dim temp2 As Long
1326|
1327|  FinalSQLStatement = OriginalQuery.SQL
1328|  Set Junk2 = OriginalQuery.ListParameters()
1329|  Do Until Junk2.EOF
1330|     ParamName = Junk2("Name")
1331|     UserPrompt = ParamName & " @@ of type " & FldTypeIDNums(Val(Junk2("Type")))
1332|     ParamChosen = InputBox(UserPrompt, "Parameter Query " & OriginalQuery.Name)
1333| 'The following two lines attempt to set the
1334| 'parameters of a parameter query, but this can apparently not
1335| ' be done.  According to Microsoft, you can't apparently run
1336| ' a parameter query from code without hard-wiring it.
1337| 'OriginalQuery!ParamName = paramchosen
1338| 'NewQuery=OriginalQuery.createsnapshot()
1339| '
1340| 'The apparent solution is to take the query, find the parameters, then
1341| ' programatically substitute user choices for the parameters in the
1342| ' SQL statement, then run that query.
1343|    If ((Junk2("Type") > 7) Or (Junk2("Type") = 0)) Then 'Add Single quotes for all
      datatypes that might be strings.
1344|        If (ParamChosen = "") Then
1345|          ParamChosen = "'~'"
1346|        Else
1347|          ParamChosen = "'" & ParamChosen & "'"
1348|        End If
1349|      Else ' for numeric datatypes, use a default of -1
1350|        If (ParamChosen = "") Then ParamChosen = "-1"
1351|    End If
1352|    FinalSQLStatement = ReplaceString(FinalSQLStatement, ParamName,
      ParamChosen)
1353|    Junk2.MoveNext
1354|  Loop
1355|  Junk2.Close
1356|  OriginalQuery.Close
```

```
1357| 'Now, zap that damn parameter statement.
1358|  Temp1 = InStr(FinalSQLStatement, "PARAMETER")                    'Find the first
       occurence of PARAMETER
1359|  If Temp1 <> 0 Then 'The SQL statement DOES contain a parameter section
1360|     temp2 = InStr(Mid$(FinalSQLStatement, Temp1, 65535), Chr$(10))          'and the
       first CR-LF after that
1361|     FinalSQLStatement = Left$(FinalSQLStatement, Temp1 - 1) &
       Mid$(FinalSQLStatement, temp2 + Temp1)   'and throw out them and everything in
       between.
1362|  End If
1363|  'Set snpResults = DB.CreateSnapshot(FinalSQLStatement)
1364| 'Figure out how to set the parameter query to something VBasic does not
1365| '  object to.  VBasic does object to any attempt to SET a formal parameter.
1366| '
1367| End Sub
1368|
```

```
 1| Option Explicit
 2|
 3| Sub asngPrint (A() As Single)
 4| Dim n As Long
 5| For n = LBound(A) To UBound(A)
 6|    Debug.Print n, A(n)
 7| Next n
 8| End Sub
 9|
10| Sub DBGPrintAllThemes ()
11| Dim n As Long
12| For n = 1 To nThemes
13|  Debug.Print "Theme #" & n
14|  DBGPrintTheme Themes(n)
15| Next
16| End Sub
17|
18| Sub DBGPrintTheme (ThemeToPrint As typTheme)
19|
20|    Debug.Print "ThemeName=" & ThemeToPrint.ThemeName  ' As String     'A name
    for the layer, like "Appropiations, sized"
21|    Debug.Print "MapSource=" & ThemeToPrint.MapSource 'As String 'SQL statement,
    QueryDef, or Table name.  Used to open the snapshot.
22|    Debug.Print "DBSource=" & ThemeToPrint.DBSource 'As String   '  for the data
    displayed in the grid.
23|
24|    Debug.Print "Order=" & ThemeToPrint.myIndex;  ' As Long     'Which order to plot this
    in.  Low numbers appear on the bottom.  Numbers less than or equal to zero are not
    plotted.
25|    Debug.Print "ThemeType=" & ThemeToPrint.ThemeType; 'As Long 'Indicates line,
    point, polygon.....
26|    Debug.Print "FirstLoc=" & ThemeToPrint.FirstLoc; 'As Long 'Pointers to first and last
    locations in the appropriate array for this theme.
27|    Debug.Print "LastLoc=" & ThemeToPrint.LastLoc; ' As Long
28|   Debug.Print
29|   Debug.Print
30| End Sub
31|
32| Sub PrintAPIPoints (ArrayIn() As APIPOINT)
33|  Dim n As Long
34|  For n = LBound(ArrayIn) To UBound(ArrayIn)
35|    Debug.Print "n=" & n & "  X=" & ArrayIn(n).X & "    Y=" & ArrayIn(n).Y
36|  Next n
37|
38| End Sub
39|
40| Sub PrintRealPoints (ArrayIn() As RealPoint)
41|  Dim n As Long
42|  For n = LBound(ArrayIn) To UBound(ArrayIn)
43|    Debug.Print "n=" & n & "  X=" & ArrayIn(n).X & "    Y=" & ArrayIn(n).Y
44|  Next n
45| End Sub
46|
```

```
Option Explicit
Option Base 1
'This module contains DeFinitions of VaRiables, TyPes, and NuMbers.
'Assembled from typdef, vardecl, and numdef
Global INIFile As String 'Probable location of the wheat INI file
Global myUserName As String 'Identity of current user




'Visual Basic defininitions of the structures used in
' the WHEAT modular GIS/RS package.
'
'Copyright 1993 Greg Pouch
'
Type typTheme
   myIndex As Long    'Order in which this theme was loaded.  Stores its index in Themes()
   ThemeType As Long 'Indicates line, point, polygon.....
   FirstLoc As Long 'Pointers to first and last locations in the appropriate array for this theme.
   LastLoc As Long
   ThemeName As String     'A name for the layer, like "Appropiations, sized"
   MapSource As String 'SQL statement, QueryDef, or Table name.  Used to open the snapshot.
   DBSource As String   '  for the data displayed in the grid.
End Type
Global Themes() As typTheme



'Type RealRect   ' a rectangle with REAL*4 coordinates
'   Left As Single
'   Top As Single
'   Right As Single
'   Bottom As Single
'End Type
'^^The above is the definition found as of 940520.
' this conflicted with the definition in geomlib1, so it was chnged
' the following defintion.



Type RealRect   ' a rectangle with REAL*4 coordinates
   Left As Single
   Top As Single
   Right As Single      'New definition, as of 951202 to bring it into conformity with WhtGeom2 and the Windows APIRect
   Bottom As Single     ' Note that it is now a lot like two points
End Type



```

```
 51|
 52| Type RealPoint              ' a single xy point
 53|  X As Single
 54|  Y As Single
 55| End Type
 56|
 57|
 58| Type TileDeclDeluxe 'DataType Tiles for things like sections, townships, etc.
 59|           ' which would almost be rasters but might be scattered, entered
 60|           ' by hand or in an irregular order, etc.
 61|    X As Single
 62|    Y As Single
 63|    Zval As Single 'the value of Z for the user-selected variable [db field]
 64|                ' at the point.  It will determine
 65|                ' what color to make it or symbol to use
 66|    FeatureName As String * 16  'a name the user can recognize
 67|    SymbolText As String * 1'Use Pouch Trial Font#1.character # for squares
 68|    Color As Long ' color to use to plot the Tiles
 69|    Height As Single 'How high (delta-Y) is this Tiles
 70|    Width As Single  'How Wide (delta-X) is this Tiles
 71| End Type
 72|
 73|
 74| Type TileDeclPlain 'DataType Tiles for things like sections, townships, etc.
 75|           ' which would almost be rasters but might be scattered, entered
 76|           ' by hand or in an irregular order, etc.
 77|    nStart As Long
 78|    Zval As Single 'the value of Z for the user-selected variable [db field]
 79|                ' at the point.  It will determine
 80|                ' what color to make it or symbol to use
 81|    ZColor As Long
 82|    FeatureName As String * 16  'a name the user can recognize
 83| End Type
 84|
 85| Type TileDecl2 'DataType Tiles for things like sections, townships, etc.
 86|           ' which would almost be rasters but might be scattered, entered
 87|           ' by hand or in an irregular order, etc.
 88|    FeatureID As Long
 89|    'FeatureName As String * 14  'a name the user can recognize
 90|    nStart As Long 'TileDecls have two points, defining the lower left and upper right
    corners.
 91|    OutlineStyle As Integer
 92|    OutlineWidth As Integer
 93|    OutlineColor As Long
 94|    InteriorColor As Long
 95|    InteriorFillStyle As Integer
 96|
 97| End Type
 98|
 99| 'Global Tiles() As TileDeclPlain
100| Global Tiles() As TileDecl2
101| Global nTiles As Long
102|
103|
```

```
104|              'DataType JOINTEDLINE for things like roads, rivers, boundaries....
105| Type LineDeclPlain
106|    FeatureID As Long 'FeatureName As String * 32  'a name the user can recognize
107|    nStart As Long
108|    nPoints As Long
109|    'XYBlob As String   'Could replace with start and nPoints, both Long, in a global array.
110|    ZWidth As Integer 'the size value of Z for the user-selected variable [db field]
111|    ZStyle As Integer  'the symbol number
112|    ZColor As Long  'the color
113|    MinX As Single
114|    MaxX As Single
115|    MinY As Single
116|    MaxY As Single
117|
118| End Type
119|
120| Global Lines() As LineDeclPlain
121| Global nLines As Long
122|
123| Type PointDeclPlain 'DataType ONEPOINT for things like wells, benchmarks .....
124|    FeatureID As Long ' FeatureName As String * 16  'a name the user can recognize
125|    nStart As Long
126|    ZSize As Single      'the size value of Z for the user-selected variable [db field]
127|    ZColor As Long       'the color
128|    ZSymbol As Integer   'the symbol number
129|    ZFontCode As Integer
130|
131| End Type
132|
133| Global Points() As PointDeclPlain
134| Global nPoints As Long
135|
136|              'DataType HACHLINE for things like shading in polygons, contours (maybe)
137| Type HatchLinePlain
138|    FeatureID As Long 'FeatureName As String * 32  'a name the user can recognize
139|    nStart As Long
140|    nPoints As Long 'Must be a multiple of two.  This will draw lines from P1 to P2, P3 to
     P4 .....
141|    ZWidth As Integer 'the size value of Z for the user-selected variable [db field]
142|    ZStyle As Integer  'the line style
143|    ZColor As Long  'the color
144|    MinX As Single
145|    MaxX As Single
146|    MinY As Single
147|    MaxY As Single
148|
149| End Type
150| Global HatchLines() As HatchLinePlain
151|
152|
153|  'DataType SOLIDPOLYGON for things like roads, rivers, boundaries....
154| Type typPoly
155|    FeatureID As Long ' FeatureName As String * 28  'a name the user can recognize
156|    nStart As Long
```

```
157|    nPoints As Long
158|    FillColor As Long     'The color of the interior
159|    OutlineColor As Long  'The color of the outline
160|    OutlineWidth As Integer
161|    FillStyle As Integer  'To indicate solid, vertically cross-hatched, etc.
162|    MinX As Single
163|    MaxX As Single
164|    MinY As Single
165|    MaxY As Single
166|
167|  End Type
168|
169|  Global Polys() As typPoly
170|  Global nPolys As Long
171|
172|
173|
174|  Type LabelDecl    'DataType LABELTEXT for things like wells, benchmarks .....
175|    FeatureID As Long 'FeatureName As String * 16  'a name the user can recognize
176|    nStart As Long       'Location of the point in memory
177|    FontSize As Single      'the size for the user-selected variable [db field]
178|    ZColor As Long       'the color
179|    FontCode As Integer
180|    Alignment As Integer
181|    Opaque As Integer
182|
183|    Text As String * 64 'Back to fixed length, and use trim$ to display it.
184|    'Text to be posted.  Note that
185|                  'this is now a variable size string
186|                  'rather than the original 96 characters.
187|                  'This was done so that right alignment and centering
188|                  ' can be done properly, but makes the type
189|                  ' uncopyable using LSET.  Need to write a function
190|                  '  to copy this type if it needs copying.
191|  End Type              'Used for linking to spatial queries, too.
192|
193|  Global Labels() As LabelDecl
194|  Global nLabels As Long
195|
196|
197|
198|  Type typTaggedIDNum
199|    IDNum As Long     'The array index.  DOES VARY across runs.
200|    FeatureID As Long 'The serial number. Doesn't vary across runs
201|    X As Single
202|    Y As Single
203|    GeometricFlags As Long
204|    Dist1 As Single
205|    Dist2 As Single
206|
207|  End Type
208|
209|
210|
```

```
211|
212| Type ToolDescription
213|  CmdName As String
214|  CmdIDNum As Integer
215|  hCursor As Integer
216|  hIcon As Integer
217| End Type
218|
219| Type FieldDescription  'used to store the results of a ListFields snapshot.
220|    Name  As String      'Name of the field. in the recordset
221|    Type  As Long        'Number for the field data type.  See table below.
222|    Size  As Long        'Maximum size of the field.
223|                         'For Long Text and Long Binary fields (Type 11 and 12), Size is 0;
224|                         ' for fields of Type 1 or 2, Size is 1.
225|    Attributes As Long   'Numeric value for one or more attributes of the field.  See table
     below.
226|    SourceTable As String'Name of the underlying table that supplied the field.
227|    SourceField As String'Name of the field in the underlying table.
228|    CollatingOrder As Integer  'Comparison code for the current language.  Use this value
     for the compare part of the StrComp and InStr functions.
229| End Type
230|
231|
232|
233|
234|
235|
236| Type SSSpatialQuery
237|    FeatureType As Integer
238|    ThemeName As String
239|    SQLSpatial As String
240|    SQLTable As String
241| End Type
242|
243| Type APIPoint
244|    X As Integer
245|    Y As Integer
246| End Type
247| Type APIRect
248|    Left   As Integer
249|    Top    As Integer
250|    Right  As Integer
251|    Bottom As Integer
252| End Type
253|
254|
255|
256|
257| 'Data access variables of global importance.
258| Global MapDB As Database 'The name of the database on which the map is based.
259|                          ' only one database allowed for now.
260| Global tblFonts As Table 'A list of the available fonts.
261|
262|
```

```
Global snpTblQryList As Snapshot 'List of Tables and Queries in the map database.
Global nTblQrys As Long
Global strTblQryList() As String
Global snpStatsOnOneVariable As Snapshot


Global snpFieldList As Snapshot
Global nFields As Long
Global strFieldsList() As String 'List of Fields

Global snpMapTileQuery As Snapshot
Global snpMapPointQuery As Snapshot
Global snpMapLineQuery As Snapshot
Global snpMapPolyQuery As Snapshot
Global snpMapLabelQuery As Snapshot

'Variables delaing with the global memory block that holds the point data.
Global hGMemAllPoints As Integer 'This is one big global array that holds ALL geometric
data points (point coordinates, line coordinates, etc)
Global hGMemLastPoint As Long  ' This indicates the current last location,
                    'in terms of XYPoints (single) (ie, it is the number of the last point
added to hGMemeAllPoints)
Global lngHGAllPointsMemSize As Long, lngOne As Long


Global flgStopDrawing  As Integer  'a flag to halt drawing
Global flgStopPrinting As Integer  'a flag to halt printer output.

Global NThemes As Long 'The number of Themes (map layers, data types, datasets.....)

Global strMapDBName As String  'the current input file


Global ToolPick(-1 To 50) As ToolDescription 'A short description, including command
string, icon, and cursor ID Numbers

Global TileSizeX As Single 'dimensions of sections.
Global TileSizeY As Single

'This is being kept global, as (if I do convert this to MDI)
'   all MDI forms would share the same layers, and thus databounds,
'    while ViewBounds and such would vary from map to map
' GWPouch 940516
Global DataBounds As RealRect 'the bounding rectangle of the data
                    ' this changes only when new data is
                    ' added
Global DataAspect As Single ' (width/height) for data
Global DataRange As RealPoint


Global MinColorCode As Integer
Global MaxColorCode As Integer
```

```
Global MaxZ As Single ' The maximum Z-value displayed as a Tiles
Global MinZ As Single ' The minimum Z-Value displayed as a Tiles
Global MeanZ As Single
Global StdDevZ As Single

Global TempColor() As Long
'These were originally local variables on the


Global NFieldsOnMap As Integer
Global NFieldsOnDB As Integer


Global FlagRedrawColorBar As Integer
Global FlagCustomCursor  As Integer



Global FontList() As String
Global nFonts As Long

Global Colors(0 To 255)   As Long
Global nSteps As Integer 'Number of steps used in a
                         ' gradational color scheme.
Global nStdDev As Single
Global ColorScaleFactor As Single




Global SizeLookFor As String, SizeShowAlways As String
Global ColorLookFor As String, ColorShowAlways As String
Global SymbolCodeLookFor As String, SymbolCodeShowAlways As String
Global XLookFor As String, XShowAlways As String, YLookFor As String, YShowAlways
As String
Global FeatureNameLookFor As String, FeatureNameShowAlways As String
Global XYLookFor As String, XYShowAlways As String
Global FontCodeLookFor As String, FontCodeShowAlways As String
Global FeatureTextLookFor As String, FeatureTextShowAlways As String
Global LineWidthLookFor As String, LineWidthShowAlways As String
Global LineStyleLookFor As String, LineStyleShowAlways As String
Global FillStyleLookFor As String, FillStyleShowAlways As String
Global AlignmentLookFor  As String, AlignmentShowAlways As String

Global DefaultHighLightColor As Long, DefaultHighLightRadius As Single



'Begin copied from LUT handling
Type SymbolLUT  ' Look-Up Table yielding a symbol (stored as a string*1 character)
   Min As Single
   Max As Single
   Class As Long
   Output As String * 1 'the symbol
```

```
    SymbolCode As Integer 'Numeric representation of the above character
    FontCode As Integer   'Which font did it come from.
End Type
Global Const LUT_SYMBOL = 1

Type I2LUT
    Min As Single
    Max As Single
    Class As Integer
    Output As Integer
End Type
Global Const LUT_I2 = 2

Type I4LUT
    Min As Single
    Max As Single
    Class As Long
    Output As Long
End Type
Global Const LUT_I4 = 4

Type R4LUT  ' Look-Up Table yielding a Real*4/Single
    Min As Single
    Max As Single
    Class As Long
    Output As Single
End Type
Global Const LUT_R4 = 8

Type StringLUT  ' Look-Up Table yielding a symbol (stored as a string*1 character)
    Min As Single
    Max As Single
    Class As Long
    Output As String
End Type
Global Const LUT_STRING = 16

Type VariantLUT
    Min As Single
    Max As Single
    Class As Integer
    Output As Variant
End Type
Global Const LUT_VARIANT = 32


Type ColorLUT  'Color Look-Up Table
    Min As Single
    Max As Single
    Class As Long
    VBColor As Long
    GDIColor As Long
End Type
Global Const LUT_COLOR = 64
```

```
421|
422| Type typRangeLUT
423|    Min As Single
424|    Max As Single
425|    Class As Long
426| End Type
427| 'Used to set the bounds, then sucked into a specific type of LUT.
428| Global RangeLUT() As typRangeLUT
429|
430| 'End copied from LUT handling
431|
432|
433| 'Type SpatialQuery
434| '   FeatureType As Integer
435| '   ThemeName As String
436| '   SQLSpatial As String
437| '   SQLAttributes As String
438| 'End Type
439|
440|
441|
442|
443| Global DummyReal As Single
444| Global DummyLong As Long
445| Global DummyShort As Integer
446| Global DummyDouble As Double
447| Global DummyString As String
448| Global DummyAnything As Variant
449| Global ThemesToPrint() As typTheme
450| Global PrinterRectangle As RealRect
451| Global WorldViewRect As RealRect
452|
453| Global jj As Long
454| Global ii As Long
455|
456|
457|
458|
459| 'File Open/Save Dialog Flags
460| Global Const OFN_READONLY = &H1&
461| Global Const OFN_OVERWRITEPROMPT = &H2&
462| Global Const OFN_HIDEREADONLY = &H4&
463| Global Const OFN_NOCHANGEDIR = &H8&
464| Global Const OFN_SHOWHELP = &H10&
465| Global Const OFN_NOVALIDATE = &H100&
466| Global Const OFN_ALLOWMULTISELECT = &H200&
467| Global Const OFN_EXTENTIONDIFFERENT = &H400&
468| Global Const OFN_PATHMUSTEXIST = &H800&
469| Global Const OFN_FILEMUSTEXIST = &H1000&
470| Global Const OFN_CREATEPROMPT = &H2000&
471| Global Const OFN_SHAREAWARE = &H4000&
472| Global Const OFN_NOREADONLYRETURN = &H8000&
473|
474|
```

```
475| 'Error Constants
476| Global Const CDERR_DIALOGFAILURE = &HFFFF
477|
478| Global Const CDERR_GENERALCODES = &H0
479| Global Const CDERR_STRUCTSIZE = &H1
480| Global Const CDERR_INITIALIZATION = &H2
481| Global Const CDERR_NOTEMPLATE = &H3
482| Global Const CDERR_NOHINSTANCE = &H4
483| Global Const CDERR_LOADSTRFAILURE = &H5
484| Global Const CDERR_FINDRESFAILURE = &H6
485| Global Const CDERR_LOADRESFAILURE = &H7
486| Global Const CDERR_LOCKRESFAILURE = &H8
487| Global Const CDERR_MEMALLOCFAILURE = &H9
488| Global Const CDERR_MEMLOCKFAILURE = &HA
489| Global Const CDERR_NOHOOK = &HB
490|
491| 'Added for CMDLG.VBX
492| Global Const CDERR_CANCEL = &HC
493| Global Const CDERR_NODLL = &HD
494| Global Const CDERR_ERRPROC = &HE
495| Global Const CDERR_ALLOC = &HF
496| Global Const CDERR_HELP = &H10
497|
498| Global Const PDERR_PRINTERCODES = &H1000
499| Global Const PDERR_SETUPFAILURE = &H1001
500| Global Const PDERR_PARSEFAILURE = &H1002
501| Global Const PDERR_RETDEFFAILURE = &H1003
502| Global Const PDERR_LOADDRVFAILURE = &H1004
503| Global Const PDERR_GETDEVMODEFAIL = &H1005
504| Global Const PDERR_INITFAILURE = &H1006
505| Global Const PDERR_NODEVICES = &H1007
506| Global Const PDERR_NODEFAULTPRN = &H1008
507| Global Const PDERR_DNDMMISMATCH = &H1009
508| Global Const PDERR_CREATEICFAILURE = &H100A
509| Global Const PDERR_PRINTERNOTFOUND = &H100B
510|
511| Global Const CFERR_CHOOSEFONTCODES = &H2000
512| Global Const CFERR_NOFONTS = &H2001
513|
514| Global Const FNERR_FILENAMECODES = &H3000
515| Global Const FNERR_SUBCLASSFAILURE = &H3001
516| Global Const FNERR_INVALIDFILENAME = &H3002
517| Global Const FNERR_BUFFERTOOSMALL = &H3003
518|
519| Global Const FRERR_FINDREPLACECODES = &H4000
520| Global Const CCERR_CHOOSECOLORCODES = &H5000
521|
522|
523|
524|
525| ' COLOR FLAGS AND FUNCTIONS
526|
527|
528| 'Windows API calls, declarations, and such
```

```
529|
530| Type PALLETTEENTRY
531|    a As Long
532| End Type
533|
534| ' Logical Palette
535| Type LOGPALETTE
536|    palVersion As Integer
537|    palNumEntries As Integer
538|    palPalEntry(0 To 235) As Long ' Array length is arbitrary; may be changed
539| End Type
540|
541|
542| Global i, j, k As Integer
543|
544|
545|
546|
547| '                BITMAPINFOHEADER
548| Type BITMAPINFOHEADER '40 bytes
549|    biSize As Long
550|    biWidth As Long
551|    biHeight As Long
552|    biPlanes As Integer
553|    biBitCount As Integer
554|    biCompression As Long
555|    biSizeImage As Long
556|    biXPelsPerMeter As Long
557|    biYPelsPerMeter As Long
558|    biClrUsed As Long
559|    biClrImportant As Long
560| End Type
561|
562| '              BITMAPFILEHEADER
563| Type BITMAPFILEHEADER
564|    bfType As String * 2
565|    bfSize As Long
566|    bfReserved1 As Integer
567|    bfReserved2 As Integer
568|    bfOffBits As Long
569| End Type
570|
571| Declare Function UnrealizeObject Lib "GDI" (ByVal hObject As Integer) As Integer
572| 'DeleteObject
573| Declare Function DeleteObject Lib "GDI" (ByVal hObject As Integer) As Integer
574|
575| 'SetTextAlign
576| Declare Function SetTextAlign Lib "GDI" (ByVal hDC As Integer, ByVal wFlags As
    Integer) As Integer
577| 'Vertical Alignment Constants
578| Global Const TA_BASELINE = 24
579| Global Const TA_BOTTOM = 8
580| Global Const TA_TOP = 0
581| ' Horizontal Alignment Constants
```

```
582| Global Const TA_CENTER = 6
583| Global Const TA_LEFT = 0
584| Global Const TA_RIGHT = 2
585|
586|
587| 'FROM FILE NUMDEF.BAS
588| 'Definitions of variable values for WHEAT programs.
589| ' nType
590|
591| Global Const SkipThis = 0     ' Included to allow the deletion of features
592|                               '  without having to shuffle memory.
593|
594| Global Const OnePoint = 1     'a single point,
595|                               'probably plotted with a symbol
596|
597| Global Const JointedLine = 2   ' for linear features such as rivers, highways...
598|
599| Global Const OutLine = 3       '  for the polygon which
600|                                 ' outlines some feature, such as a lake or the
601|                                 ' outcrop area of some formation or mapping unit.
602|                                 '  Most likely, the outcrop would be drawn as a series of
603|                                 ' blocks, then the boundary would be drawn on top of it
604|                                 ' in order to eliminate pixel jagged edges.
605|
606| Global Const BoundBlock = 4    ' the bounding rectangle of the previous feature.  Use
607|                                 ' Use this to rapidly decide whether to plot
608|                                 ' a feature.  ie, plot feature A only if the BoundsBlock
609|                                 ' of A includes the current mapsheet, otherwise skip.
610|
611| Global Const HatchLine = 5      ' Hachuring/shading lines used to indicate
612|                                 '  the area within a polygon.  Like a polygon fill color
613|                                 ' but more reliable since it only relies on the ability of
614|                                 ' a computer to draw lines.
615|
616| Global Const TileBlock = 6     'An inefficient way of representing pixels.
617|                                 ' A Tiles block will be drawn in a particular color, and
618|                                 ' will be used to fill the space on the screen.  Draw with
619|                                 ' line( )-(),color,bf   Use this only for features where
620|                                 ' not every point in the map will be represented.
621|                                 ' For continuous properties, like surface geology, use
622|                                 ' a raster layer.
623|
624| Global Const NodePoint = 7     ' A node, in case I later decide to add
625|                                 ' full vector capabilities with network topology.
626|
627| Global Const LabelText = 8     ' A feature label, like the name of a highway
628|                                 ' or lake or town .....
629|
630| Global Const SolidPolygon = 9  'The actual outline of a polygon, and the
631|                                 '  interior.  For shaded in polygons.
632| Global Const RASTERGRID = 10
633|
634| Global Const EEElipse = 20
635| Global Const PPPixel = 21
```

```
Global Const AAArc = 22
Global Const RRRectangle = 23
Global Const CCCircle = 24

Global RasterGrids() As typGridDescription

'Query spatiale type (to avoid having SQ for spatial query, which looks
'                    an awful lot like SQL)
Global Const QS_NONE = 0
Global Const QS_CLOSESTPOINT = 1
Global Const QS_NEARCURSOR = 2
Global Const QS_RADIUSSEARCH = 4
Global Const QS_WITHINBOX = 8
Global Const QS_WITHINPOLY = 16
Global Const QS_WITHINTILE = 32
Global Const QS_LINEDIST = 64

'Added 95 03 29 to allow checking of contouring/gridding
Global Const QS_RASTERGRID = 128



'Used as bit flags.
' 1 indicates this type of data can be described with a point
' 2 indicates that this type of data can be described with a line
'     Thus 3, for polygons, indicates both point and line data
' 4 indicates that feature has area
' 8 indicates that the feature is regualarly distributed
' 16 incicates
' 32
' 64
'128
Global Const PPOINTS = 1
Global Const LLINES = 2
Global Const PPOLYGONS = 3
Global Const TTILES = 5
Global Const GGRID = 9

' Colors
Global Const BLACK = &H0&
Global Const RED = &HFF&
Global Const GREEN = &HFF00&
Global Const YELLOW = &HFFFF&
Global Const BLUE = &HFF0000
Global Const MAGENTA = &HFF00FF
Global Const CYAN = &HFFFF00
Global Const WHITE = &HFFFFFF

'Modified by Greg Pouch, 10/18/93
Global Const QBBLACK = 0
Global Const QBGRAY = 8
Global Const QBBLUE = 1
Global Const QBLtBlue = 9
Global Const QBGREEN = 2
```

```
690  Global Const QBLtGreen = 10
691  Global Const QBCyan = 3
692  Global Const QBLtCyan = 11
693  Global Const QBRed = 4
694  Global Const QBLtRed = 12
695  Global Const QBMagenta = 5
696  Global Const QBLtMagenta = 13
697  Global Const QBYellow = 6
698  Global Const QBLtYellow = 14
699  Global Const QBWhite = 7
700  Global Const QBBrtWhite = 15
701
702
703  ' System Colors
704  Global Const SCROLL_BARS = &H80000000          ' Scroll-bars gray area.
705  Global Const DESKTOP = &H80000001              ' Desktop.
706  Global Const ACTIVE_TITLE_BAR = &H80000002     ' Active window caption.
707  Global Const INACTIVE_TITLE_BAR = &H80000003   ' Inactive window caption.
708  Global Const MENU_BAR = &H80000004             ' Menu background.
709  Global Const WINDOW_BACKGROUND = &H80000005    ' Window background.
710  Global Const WINDOW_FRAME = &H80000006         ' Window frame.
711  Global Const MENU_TEXT = &H80000007            ' Text in menus.
712  Global Const WINDOW_TEXT = &H80000008          ' Text in windows.
713  Global Const TITLE_BAR_TEXT = &H80000009       ' Text in caption, size box,
     scroll-bar arrow box..
714  Global Const ACTIVE_BORDER = &H8000000A        ' Active window border.
715  Global Const INACTIVE_BORDER = &H8000000B      ' Inactive window border.
716  Global Const APPLICATION_WORKSPACE = &H8000000C ' Background color of
     multiple document interface (MDI) applications.
717  Global Const HIGHLIGHT = &H8000000D            ' Items selected item in a control.
718  Global Const HIGHLIGHT_TEXT = &H8000000E       ' Text of item selected in a control.
719  Global Const BUTTON_FACE = &H8000000F          ' Face shading on command
     buttons.
720  Global Const BUTTON_SHADOW = &H80000010        ' Edge shading on command
     buttons.
721  Global Const GRAY_TEXT = &H80000011            ' Grayed (disabled) text.  This color is
     set to 0 if the current display driver does not support a solid gray color.
722  Global Const BUTTON_TEXT = &H80000012          ' Text on push buttons.
723
724  ' Key Codes
725  Global Const KEY_LBUTTON = &H1
726  Global Const KEY_RBUTTON = &H2
727  Global Const KEY_CANCEL = &H3
728  Global Const KEY_MBUTTON = &H4   ' NOT contiguous with L & RBUTTON
729  Global Const KEY_BACK = &H8
730  Global Const KEY_TAB = &H9
731  Global Const KEY_CLEAR = &HC
732  Global Const KEY_RETURN = &HD
733  Global Const KEY_SHIFT = &H10
734  Global Const KEY_CONTROL = &H11
735  Global Const KEY_MENU = &H12
736  Global Const KEY_PAUSE = &H13
737  Global Const KEY_CAPITAL = &H14
738  Global Const KEY_ESCAPE = &H1B
```

```
Global Const KEY_SPACE = &H20
Global Const KEY_PRIOR = &H21
Global Const KEY_NEXT = &H22
Global Const KEY_END = &H23
Global Const KEY_HOME = &H24
Global Const KEY_LEFT = &H25
Global Const KEY_UP = &H26
Global Const KEY_RIGHT = &H27
Global Const KEY_DOWN = &H28
Global Const KEY_SELECT = &H29
Global Const KEY_PRINT = &H2A
Global Const KEY_EXECUTE = &H2B
Global Const KEY_SNAPSHOT = &H2C
Global Const KEY_INSERT = &H2D
Global Const KEY_DELETE = &H2E
Global Const KEY_HELP = &H2F

' KEY_A thru KEY_Z are the same as their ASCII equivalents: 'A' thru 'Z'
' KEY_0 thru KEY_9 are the same as their ASCII equivalents: '0' thru '9'

Global Const KEY_NUMPAD0 = &H60
Global Const KEY_NUMPAD1 = &H61
Global Const KEY_NUMPAD2 = &H62
Global Const KEY_NUMPAD3 = &H63
Global Const KEY_NUMPAD4 = &H64
Global Const KEY_NUMPAD5 = &H65
Global Const KEY_NUMPAD6 = &H66
Global Const KEY_NUMPAD7 = &H67
Global Const KEY_NUMPAD8 = &H68
Global Const KEY_NUMPAD9 = &H69
Global Const KEY_MULTIPLY = &H6A
Global Const KEY_ADD = &H6B
Global Const KEY_SEPARATOR = &H6C
Global Const KEY_SUBTRACT = &H6D
Global Const KEY_DECIMAL = &H6E
Global Const KEY_DIVIDE = &H6F
Global Const KEY_F1 = &H70
Global Const KEY_F2 = &H71
Global Const KEY_F3 = &H72
Global Const KEY_F4 = &H73
Global Const KEY_F5 = &H74
Global Const KEY_F6 = &H75
Global Const KEY_F7 = &H76
Global Const KEY_F8 = &H77
Global Const KEY_F9 = &H78
Global Const KEY_F10 = &H79
Global Const KEY_F11 = &H7A
Global Const KEY_F12 = &H7B
Global Const KEY_F13 = &H7C
Global Const KEY_F14 = &H7D
Global Const KEY_F15 = &H7E
Global Const KEY_F16 = &H7F
Global Const KEY_NUMLOCK = &H90
Global Const SHIFT_MASK = 1
```

```
Global Const CTRL_MASK = 2
Global Const ALT_MASK = 4

'Color Dialog Flags
Global Const CC_RGBINIT = &H1&
Global Const CC_FULLOPEN = &H2&
Global Const CC_PREVENTFULLOPEN = &H4&
Global Const CC_SHOWHELP = &H8&

'Profile string functions.  Profile int functions not declared, because you
'   can use the string functions, then val them.
'
Declare Function GetProfileString Lib "Kernel" (ByVal lpAppName As String, lpKeyName
As Any, ByVal lpDefault As String, ByVal lpReturnedString As String, ByVal nSize As
Integer) As Integer
'Declare Function GetPrivateProfileString Lib "Kernel" (ByVal lpApplicationName As
String, lpKeyName As Any, ByVal lpDefault As String, ByVal lpReturnedString As String,
ByVal nSize As Integer, ByVal lpFileName As String) As Integer
Declare Function GetPrivateProfileString Lib "Kernel" (ByVal lpSectionName As String,
ByVal lpKeyName As String, ByVal lpDefault As String, ByVal lpReturnedString As
String, ByVal nSize As Integer, ByVal lpFileName As String) As Integer
Declare Function WriteProfileString Lib "Kernel" (ByVal lpApplicationName As String,
lpKeyName As Any, lpString As Any) As Integer
Declare Function WritePrivateProfileString Lib "Kernel" (ByVal lpApplicationName As
String, lpKeyName As Any, lpString As Any, ByVal lplFileName As String) As Integer

Sub CopyTYPTheme (ThemeIn As typTheme, ThemeOut As typTheme)
    ThemeOut.myIndex = ThemeIn.myIndex
    ThemeOut.ThemeType = ThemeIn.ThemeType
    ThemeOut.FirstLoc = ThemeIn.FirstLoc
    ThemeOut.LastLoc = ThemeIn.LastLoc
    ThemeOut.ThemeName = ThemeIn.ThemeName
    ThemeOut.MapSource = ThemeIn.MapSource
    ThemeOut.DBSource = ThemeIn.DBSource

End Sub

Sub InitializeWHEATEnv (INIFile As String)

Dim Buffer  As String, intSizeBuffer As Integer, nBytesBack As Integer
Buffer = Space$(1024)
'The LOOKFOR and SHOWALWAYS strings for each variable for which one is
' provided are to be kept in the [DEFAULT FIELDS] section of the wheat.ini file.

Dim SectionName As String, KeyString As String, DefaultValue As String
    SectionName = "DefaultFields"

'SizeLookFor = "FONTSIZE HEIGHT WIDTH SIZE FONT "
  KeyString = "SizeLookFor"
  DefaultValue = "FONTSIZE HEIGHT FONT "
  Buffer = Space$(1024)
  nBytesBack = GetPrivateProfileString(SectionName, ByVal KeyString, DefaultValue,
Buffer, 1024, INIFile)
  SizeLookFor = Left$(Buffer, nBytesBack) & " "
```

```
838|
839| 'SizeShowAlways = "10 12 18 24 36 "
840|    Buffer = Space$(1024)
841|    KeyString = "SizeShowAlways"
842|    DefaultValue = "10 12 18 24 36 8 6 "
843|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
     1024, INIFile)
844|    SizeShowAlways = Left$(Buffer, nBytesBack) & " "
845| 'ColorLookFor
846|    Buffer = Space$(1024)
847|    KeyString = "ColorLookFor"
848|    DefaultValue = "COLOR COLOUR TINT HUE "
849|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
     1024, INIFile)
850|    ColorLookFor = Left$(Buffer, nBytesBack) & " "
851|
852| 'ColorShowAlways
853|    Buffer = Space$(1024)
854|    KeyString = "ColorShowAlways"
855|    DefaultValue = "0<=>Black 255<=>BrightRed 65280<=>BrightGreen
     16711680<=>BrightBlue 8421504<=>Gray 16777215<=>White 65535<=>Yellow
     33023<=>Orange "
856|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
     1024, INIFile)
857|    ColorShowAlways = Left$(Buffer, nBytesBack) & " "
858|
859| 'SymbolCodeLookFor
860|    Buffer = Space$(1024)
861|    KeyString = "SymbolCodeLookFor"
862|    DefaultValue = "SYM CODE "
863|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
     1024, INIFile)
864|    SymbolCodeLookFor = Left$(Buffer, nBytesBack) & " "
865|
866| 'SymbolCodeShowAlways
867|    Buffer = Space$(1024)
868|    KeyString = "SymbolCodeShowAlways"
869|    DefaultValue = "45 51 33 43 "
870|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
     1024, INIFile)
871|    SymbolCodeShowAlways = Left$(Buffer, nBytesBack) & " "
872|
873| 'XLookFor
874|    Buffer = Space$(1024)
875|    KeyString = "XLookFor"
876|    DefaultValue = "X_ EAST WEST LONG "
877|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
     1024, INIFile)
878|    XLookFor = Left$(Buffer, nBytesBack) & " "
879|  XShowAlways = "" 'Ignore/overwrite user prefernce for having a constant X in some
     map layer.
880|
881| 'YLookFor
882|    Buffer = Space$(1024)
```

```
    KeyString = "YLookFor"
    DefaultValue = "Y_ NORTH SOUTH LAT "
    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
1024, INIFile)
    YLookFor = Left$(Buffer, nBytesBack) & " "
  YShowAlways = "" 'Ignore/overwrite user prefernce for having a constant X in some
map layer.

'FeatureNameLookFor
    Buffer = Space$(1024)
    KeyString = "FeatureNameLookFor"
    DefaultValue = "SERIAL NUM ID # "
    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
1024, INIFile)
    FeatureNameLookFor = Left$(Buffer, nBytesBack) & " "
  FeatureNameShowAlways = "" 'Ignore/overwrite user prefernce for having a constant
NAME in some map layer.

'XYLookFor
    Buffer = Space$(1024)
    KeyString = "XYLookFor"
    DefaultValue = "XY BLOB CHAIN LINE POLY BORDER BOUNDARY "
    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
1024, INIFile)
    XYLookFor = Left$(Buffer, nBytesBack) & " "
  XYShowAlways = "" 'Ignore/overwrite user prefernce for having a constant NAME in
some map layer.

'FontCodeLookFor
    Buffer = Space$(1024)
    KeyString = "FontCodeLookFor"
    DefaultValue = "FONTCODE TYPE "
    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
1024, INIFile)
    FontCodeLookFor = Left$(Buffer, nBytesBack) & " "

'FontCodeLookFor
    Buffer = Space$(1024)
    KeyString = "FontCodeShowAlways"
    DefaultValue = "0 1 2 3 "
    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
1024, INIFile)
    FontCodeShowAlways = Left$(Buffer, nBytesBack) & " "

'FeatureTextLookFor
    Buffer = Space$(1024)
    KeyString = "FeatureTextLookFor"
    DefaultValue = "TEXT LABEL NAME DESC "
    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
1024, INIFile)
    FeatureTextLookFor = Left$(Buffer, nBytesBack) & " "

'FeatureTextShowAlways
    Buffer = Space$(1024)
```

```
928|    KeyString = "FeatureTextShowAlways"
929|    DefaultValue = "PutYourMessageHere,In DoubleQuotes"
930|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
     1024, INIFile)
931|    FeatureTextShowAlways = Left$(Buffer, nBytesBack) & " "
932|
933| 'LineWidthLookFor
934|    Buffer = Space$(1024)
935|    KeyString = "LineWidthLookFor"
936|    DefaultValue = "LINEWIDTH WID THICK "
937|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
     1024, INIFile)
938|    LineWidthLookFor = Left$(Buffer, nBytesBack) & " "
939|
940| 'LineWidthShowAlways
941|    Buffer = Space$(1024)
942|    KeyString = "LineWidthShowAlways"
943|    DefaultValue = "1 2 3 4 5 6 "
944|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
     1024, INIFile)
945|    LineWidthShowAlways = Left$(Buffer, nBytesBack) & " "
946|
947| 'LineStyleLookFor
948|    Buffer = Space$(1024)
949|    KeyString = "LineStyleLookFor"
950|    DefaultValue = "LINESTYLE STYLE "
951|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
     1024, INIFile)
952|    LineStyleLookFor = Left$(Buffer, nBytesBack) & " "
953|
954| 'LineStyleShowAlways
955|    Buffer = Space$(1024)
956|    KeyString = "LineStyleShowAlways"
957|    DefaultValue = "0<=>Solid 1<=>Dash 2<=>Dot 3<=Dash-Dot 4<=>Dash-Dot-Dot
     5<=>Transparent 6<=>Inside_Solid "
958|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
     1024, INIFile)
959|    LineStyleShowAlways = Left$(Buffer, nBytesBack) & " "
960|
961| 'FillStyleLookFor
962|    Buffer = Space$(1024)
963|    KeyString = "FillStyleLookFor"
964|    DefaultValue = "FILLSTYLE FILL STYLE INTERIOR INSIDE"
965|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
     1024, INIFile)
966|    FillStyleLookFor = Left$(Buffer, nBytesBack) & " "
967|
968| 'FillStyleShowAlways
969|    Buffer = Space$(1024)
970|    KeyString = "FillStyleShowAlways"
971|    DefaultValue = "0<=>Solid 1<=>Transparent 2<=>Horizontal 3<=>Vertical
     4<=>Upward_Diagonal 5<=>DownwardDiagonal 6<=>HVCross(+)
     7<=>DiagonalCross(x) "
972|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
```

```
      1024, INIFile)
 973|    FillStyleShowAlways = Left$(Buffer, nBytesBack) & " "
 974|
 975| 'AlignmentLookFor
 976|    Buffer = Space$(1024)
 977|    KeyString = "AlignmentLookFor"
 978|    DefaultValue = "ALIGN ORIENT "
 979|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
      1024, INIFile)
 980|    AlignmentLookFor = Left$(Buffer, nBytesBack) & " "
 981| 'AlignmentShowAlways
 982|    Buffer = Space$(1024)
 983|    KeyString = "AlignmentShowAlways"
 984|    DefaultValue = "0<=>Left_Top 2<=>Right_Top 6<=>Center_Top 8<=>Left_Bottom
      10<=>Right_Bottom 14<=>Center_Bottom 24<=>Left_Baseline 26<=>Right_Baseline
      30<=>Center_Baseline "
 985|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
      1024, INIFile)
 986|    AlignmentShowAlways = Left$(Buffer, nBytesBack) & " "
 987|
 988| '[SpatialQueries]
 989| SectionName = "SpatialQueries"
 990|    Buffer = Space$(1024)
 991|    KeyString = "HighlightColor"
 992|    DefaultValue = "255"
 993|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
      1024, INIFile)
 994|    DefaultHighLightColor = Val(Buffer)
 995|
 996|    Buffer = Space$(1024)
 997|    KeyString = "HighlightRadius"
 998|    DefaultValue = "400"
 999|    nBytesBack = GetPrivateProfileString(SectionName, KeyString, DefaultValue, Buffer,
      1024, INIFile)
1000|    DefaultHighLightRadius = Val(Buffer)
1001|    If DefaultHighLightRadius <= 0 Then DefaultHighLightRadius = 200
1002|
1003|
1004|
1005|
1006| End Sub
1007|
```

```
 1| Option Explicit
 2| Option Base 1
 3| ' This module is for use in providing an EQUIVALENCE
 4| ' functionality apparently missing in Visual Basic.
 5|
 6| ' There are a variety of datatypes, name Ztn or Zmtn
 7| ' where t is the first letter of the corresponding Fortran
 8| ' data type, n is the number of bytes in the data type
 9| ' and m is the number of elements in an array of the data types.
10| '
11| ' Fortran   Basic
12| ' Real*4    Single
13| ' Real*8    Double
14| ' Integer*2 Integer
15| ' Integer*4 Long
16| ' Character*n String*n
17| ' Logical   Usually, an integer
18| '
19| ' In all cases, the structure contains one element, a,
20| ' which is where the data lives.
21| '
22| 'By using Lset and Rset (which left and right justify
23| ' data into structures), a single can be stored and
24| ' maninpulated as a string*4, or as an integer*4, or whatever.
25| '
26| 'Ultimately, this will let me store a variably-dimensioned
27| ' array, along with descriptor data, as a string, and thus
28| ' in a memo or long binary (aka OLE object) field in Access/VBasic Access.
29| '
30| 'It also lets me pull the high-order and low-order words out
31| ' of a Windows DWORD variable rather easily, and the same trick
32| ' also works for high and low order bytes.
33| '
34| Type ZR4
35|  a As Single
36| End Type
37| Global ZR4a As ZR4
38|
39| Type ZI4
40|  a As Long
41| End Type
42| Global ZI4a As ZI4
43|
44| Type ZI2
45|  a As Integer
46| End Type
47| Global ZI2a As ZI2
48|
49| Type ZI8
50|  a As Currency
51| End Type
52| Global ZI8a As ZI8
53|
54| Type ZR8
```

```
 55|  a As Single
 56|  End Type
 57|  Global ZR8a As ZR8
 58|
 59|  Type ZCh8
 60|  a As String * 8
 61|  End Type
 62|  Global ZCh8a As ZCh8
 63|
 64|  Type ZCh4
 65|  a As String * 4
 66|  End Type
 67|  Global ZCh4a As ZCh4
 68|
 69|  Type ZCh2
 70|  a As String * 2
 71|  End Type
 72|  Global ZCh2a As ZCh2
 73|
 74|  Type Z8Ch1
 75|  a(8) As String * 1
 76|  End Type
 77|  Global Z8Ch1a As Z8Ch1
 78|
 79|  Type Z4Ch1
 80|  a(4) As String * 1
 81|  End Type
 82|  Global Z4Ch1a As Z4Ch1
 83|
 84|  Type Z2I2
 85|  a As Integer
 86|  b As Integer
 87|  End Type
 88|  Global Z2I2a As Z2I2
 89|
 90|  Global s As String
 91|
 92|  Function AppendInteger2 (StringIn As String, InInteger2 As Integer) As String
 93|  Static Ch2 As ZCh2
 94|  Static I2 As ZI2
 95|  I2.a = InInteger2
 96|  LSet Ch2 = I2
 97|  AppendInteger2 = StringIn & Ch2.a
 98|  End Function
 99|
100|  Function AppendInteger4 (StringIn As String, InInteger4 As Long) As String
101|  Static Ch4 As ZCh4
102|  Static I4 As ZI4
103|  I4.a = InInteger4
104|  LSet Ch4 = I4
105|  AppendInteger4 = StringIn & Ch4.a
106|  End Function
107|
108|  Function AppendReal4 (StringIn As String, InReal4 As Single) As String
```

```
109|  Static Ch4 As ZCh4
110|  Static R4 As ZR4
111|  R4.a = InReal4
112|  LSet Ch4 = R4
113|  AppendReal4 = StringIn & Ch4.a
114|  End Function
115|
116|  Function AppendReal8 (StringIn As String, InReal8 As Double) As String
117|  Static Ch8 As ZCh8
118|  Static R8 As ZR8
119|  R8.a = InReal8
120|  LSet Ch8 = R8
121|  AppendReal8 = StringIn & Ch8.a
122|  End Function
123|
124|  Function ExtractGenericData (strDataIn As String, DataType As Long) As Variant
125|  Select Case DataType
126|     Case DB_BOOLEAN, DB_BYTE, DB_INTEGER
127|        ZCh2a.a = strDataIn
128|        LSet ZI2a = ZCh2a
129|        ExtractGenericData = ZI2a.a
130|        Exit Function
131|     Case DB_LONG
132|        ZCh4a.a = strDataIn
133|        LSet ZI4a = ZCh4a
134|        ExtractGenericData = ZI4a.a
135|        Exit Function
136|     Case DB_CURRENCY
137|        ZCh8a.a = strDataIn
138|        LSet ZI8a = ZCh8a
139|        ExtractGenericData = ZI8a.a
140|        Exit Function
141|     Case DB_SINGLE
142|        ZCh4a.a = strDataIn
143|        LSet ZR4a = ZCh4a
144|        ExtractGenericData = ZR4a.a
145|        Exit Function
146|     Case DB_DOUBLE
147|        ZCh8a.a = strDataIn
148|        LSet ZR8a = ZCh8a
149|        ExtractGenericData = ZR8a.a
150|        Exit Function
151|     Case DB_DATE
152|        ZCh8a.a = strDataIn
153|        LSet ZR8a = ZCh8a
154|        ExtractGenericData = CVDate(ZR8a.a)
155|        Exit Function
156|     Case DB_TEXT
157|        ExtractGenericData = strDataIn
158|        Exit Function
159|     Case DB_LONGBINARY, DB_MEMO
160|        Beep
161|        ExtractGenericData = Null
162|     Case Else
```

```
163|        Beep
164|        ExtractGenericData = Null
165| End Select
166| End Function
167|
168| Function ExtractInteger2 (StringIn As String, Offset As Long) As Integer
169| Static Ch2 As ZCh2
170| Static I2 As ZI2
171| Ch2.a = Mid$(StringIn, Offset, 2)
172| LSet I2 = Ch2
173| ExtractInteger2 = I2.a
174| End Function
175|
176| Function ExtractInteger4 (StringIn As String, Offset As Long) As Long
177| Static Ch4 As ZCh4
178| Static I4 As ZI4
179| Ch4.a = Mid$(StringIn, Offset, 4)
180| LSet I4 = Ch4
181| ExtractInteger4 = I4.a
182| End Function
183|
184| Function ExtractReal4 (StringIn As String, Offset As Long) As Single
185| Static Ch4 As ZCh4
186| Static R4 As ZR4
187| Ch4.a = Mid$(StringIn, Offset, 4)
188| LSet R4 = Ch4
189| ExtractReal4 = R4.a
190| End Function
191|
192| Function ExtractReal8 (StringIn As String, Offset As Long) As Double
193| Static Ch8 As ZCh8
194| Static R8 As ZR8
195| Ch8.a = Mid$(StringIn, Offset, 8)
196| LSet R8 = Ch8
197| ExtractReal8 = R8.a
198| End Function
199|
200| Function HiWord (InWord As Long) As Integer
201| Dim DbleWord As ZI4
202| Dim TwoWord As Z2I2
203| DbleWord.a = InWord
204| LSet TwoWord = DbleWord
205| HiWord = TwoWord.b
206| End Function
207|
208| Function LoWord (InWord As Long) As Integer
209| Dim DbleWord As ZI4
210| Dim TwoWord As Z2I2
211| DbleWord.a = InWord
212| LSet TwoWord = DbleWord
213| LoWord = TwoWord.a
214| End Function
215|
216| Private Sub ZZZDebugDemonstration ()
```

```
 217|  'zr4a.a = 12.43
 218|  's = appendreal4("", 12.43)
 219|  's = appendreal4(s, 13000000000#)
 220|  '
 221|  'zr4a.a = extractreal4(s, 1)
 222|  'Print zr4a.a
 223|  '12.43
 224|  'Print Len(s)
 225|  '8
 226|  'Print Len(zr4a)
 227|  '4
 228|  'Print Len(zr4a.a)
 229|  ' 4
 230|  '
 231|  End Sub
 232|
```

```
Option Explicit
Type typFontLookup
   Number As Integer       'an Integer*2 ID numbers
   Name As String          'The name of the font
   ExtraSpace As String * 4  'Extra space, for whatever comes up
End Type
Global strListOfFonts() As String, FontTable()  As typFontLookup
'Global int1 As Integer, int2 As Integer
'Global rl1 As Integer, rl2 As Integer
'Global lng1 As Integer, lng2 As Integer
'Global tblIn As Table, DB As Database
Dim nF As Integer, nFonts As Integer, nFnt As Integer
Global Const FONT_BOLD = -32768
Global Const FONT_ITALIC = 16384
Const FONT_MASKIN = 16383

Sub CreateFontTable (tblFonts As Table, FontTable() As typFontLookup)
Call GetListOfPrinterFonts(strListOfFonts())
ReDim FontTable(1 To nFonts)
For nF = 1 To nFonts
   FontTable(nF).Name = strListOfFonts(nF)
   FontTable(nF).Number = nF
Next nF
Call WriteFontTable(tblFonts, FontTable())

End Sub

Sub DefineFontTable (DBIn As Database)
Dim NewTable As New TableDef
Dim fldName As New field, fldNumber As New field, fldExtra As New field

Dim fldRawFontData As New field

   NewTable.Name = "zWHEATtblFontList" 'Name the table
'Define the fields
   fldName.Name = "FontName"
   fldName.Type = 10  'Text
   fldName.Size = 255

   fldNumber.Name = "FontCode"
   fldNumber.Type = 3 'Integer*2

   fldExtra.Type = 10
   fldExtra.Size = 4
   fldExtra.Name = "ExtraData"

   fldRawFontData.Name = "RawFontData"
   fldRawFontData.Type = 11 'OLE/Binary Large OBject

'Append the fields to the table
   NewTable.Fields.Append fldName
   NewTable.Fields.Append fldNumber
   NewTable.Fields.Append fldExtra
   NewTable.Fields.Append fldRawFontData
```

```
Dim ndxPrimary As New Index, ndxName  As New Index

    ndxPrimary.Fields = "FontCode"
    ndxPrimary.Primary = True
    ndxName.Name = "FontNames"
    ndxName.Fields = "FontName"
    NewTable.Indexes.Append ndxPrimary
    NewTable.Indexes(0).Name = "PrimaryKey"
    NewTable.Indexes.Append ndxName
' Append the table to the table defs collection.
    DBIn.TableDefs.Append NewTable
End Sub

Function FontIDFromName (FontNameIn As String) As Integer
Dim n As Integer
For n = 1 To nFonts
  If UCase$(Trim$(FontTable(n).Name)) = UCase$(Trim$(FontNameIn)) Then
    FontIDFromName = FontTable(n).Number
    Exit Function
  End If
Next n
Beep
FontIDFromName = 0
End Function

Function FontNameFromID (ByVal IDNumber As Integer) As String
  Dim tmp As String
  tmp = FontTable(FONT_MASKIN And IDNumber).Name
  If tmp = "" Then
    FontNameFromID = "MS Sans Serif"
   Else
    FontNameFromID = Trim$(tmp)
  End If
End Function

Sub GetListOfPrinterFonts (strListOfFonts() As String)
    Dim nFS As Long, nFontsScreen As Long, nFP As Long, nFontsPrinter As Long
    nFontsPrinter = Printer.FontCount
    nFontsScreen = Screen.FontCount
    frmFontTable.cmbFontList.Clear
    For nFP = 0 To nFontsPrinter - 1
      For nFS = 0 To nFontsScreen - 1
        If Screen.Fonts(nFS) = Printer.Fonts(nFP) Then
frmFontTable.cmbFontList.AddItem Printer.Fonts(nFP)
      Next nFS
    Next nFP

    nFonts = frmFontTable.cmbFontList.ListCount

    ReDim strListOfFonts(1 To nFonts)
    For nFnt = 1 To nFonts
       strListOfFonts(nFnt) = frmFontTable.cmbFontList.List(nFnt - 1)
    Next nFnt
End Sub
```

```
108|
109| Function IsBold (FontCode As Integer) As Integer
110|  If (FontCode And FONT_BOLD) <> 0 Then
111|     IsBold = True
112|    Else
113|     IsBold = False
114|  End If
115| End Function
116|
117| Function IsItalic (FontCode As Integer) As Integer
118| If (FontCode And FONT_ITALIC) <> 0 Then
119|     IsItalic = True
120|  Else
121|     IsItalic = False
122| End If
123| End Function
124|
125| Sub LoadComboBoxWithFontNames (cmbTarget As ComboBox)
126|  cmbTarget.Clear
127|  cmbTarget.AddItem ""
128|  For nF = 1 To nFonts
129|     cmbTarget.AddItem Trim$(Left$(FontTable(nF).Name, 32))
130|     cmbTarget.ItemData(cmbTarget.NewIndex) = FontTable(nF).Number
131|  Next nF
132|  '    Dim Temp As String * 1
133|  '    Temp = Left$(FontTable(1).Name, 1)
134|  'cmbTarget.SetFocus
135|  'SendKeys Temp
136|  'SendKeys "+{DOWN}"
137|
138| End Sub
139|
140| Sub LoadFontTable (tblFonts As Table, FontTable() As typFontLookup)
141|
142| Dim fldName As field, fldNumber As field
143|
144| GoSub SUB_FontTableFields
145|
146|
147| On Error GoTo ERRLoadFontTable
148|
149|    ReDim FontTable(0 To 0)
150|    FontTable(0).Name = "MS Sans Serif"
151|    FontTable(0).Number = 0
152|
153|    Dim nFont As Long
154|    nFont = 0
155|    tblFonts.MoveFirst
156|
157|
158|
159|    Do
160|       nFont = nFont + 1
161|       ReDim Preserve FontTable(0 To nFont)
```

```
162|        FontTable(nFont).Name = fldName
163|        FontTable(nFont).Number = fldNumber
164|        tblFonts.MoveNext
165|        If tblFonts.EOF Then Exit Do
166|
167|     Loop 'Until tblFonts.EOF
168|
169|     nFonts = nFont
170|
171|
172| Exit Sub
173| ERRLoadFontTable:
174| If Err = 3021 Then 'No current record, because the table is empty or doesn't exist.
175|
176|     Err = 0
177|     Call CreateFontTable(tblFonts, FontTable())
178|         Resume
179| End If
180| MsgBox "Error #" & Err & " occurred in LoadFontTable." & CRLF & Error
181| Err = 0
182| Exit Sub
183| Resume
184|
185|
186| SUB_FontTableFields:
187| On Error Resume Next
188|   Set fldName = tblFonts("FontName")
189|   Set fldNumber = tblFonts("FontCode")
190|   If fldName Is Nothing Then Set fldName = tblFonts("Name")
191|   If fldNumber Is Nothing Then Set fldNumber = tblFonts("Number")
192|   Err = 0
193| Return
194|
195|
196| End Sub
197|
198| Sub SetFontCodeInControl (ByVal FontCode As Integer, TARGET As Control)
199|     On Error Resume Next
200|     TARGET.FontName = FontNameFromID(FontCode)
201|     TARGET.FontBold = IsBold(FontCode)
202|     TARGET.FontItalic = IsItalic(FontCode)
203|
204| End Sub
205|
206| Sub SetFontCodeInPIC (ByVal FontCode As Integer, PIC As PictureBox)
207|     On Error Resume Next
208|     PIC.FontName = FontNameFromID(FontCode)
209|     PIC.FontBold = IsBold(FontCode)
210|     PIC.FontItalic = IsItalic(FontCode)
211| End Sub
212|
213| Sub SetFontCodeInPRINTER (ByVal FontCode As Integer)
214|    On Error Resume Next
215|     Printer.FontName = FontNameFromID(FontCode)
```

```
216|     Printer.FontBold = IsBold(FontCode)
217|     Printer.FontItalic = IsItalic(FontCode)
218| End Sub
219|
220| Sub WriteFontTable (tblFonts As Table, FontTable() As typFontLookup)
221|
222|   For nFnt = 1 To nFonts
223|     tblFonts.AddNew
224|     tblFonts("FontName") = FontTable(nFnt).Name
225|     tblFonts("FontCode") = FontTable(nFnt).Number
226|     tblFonts.Update
227|   Next nFnt
228| End Sub
229|
```

```
Option Explicit
'Type RealPoint
'   X As Single
'   Y As Single
'End Type

'Global hGMemLastPoint As Long, CRLF As String, lngHGAllPointsMemSize As Long


'structure /RealRect/
'       real*4 Left
'       real*4 Right
'       real*4 Bottom
'       real*4 Top
'      end structure

'Type RealRect   ' a rectangle with REAL*4 coordinates
'   Left As Single
'   Right As Single
'   Bottom As Single
'   Top As Single
'End Type
Type BoundaryLine
    m As Single 'the slope
    h As String * 1'flag to indicate horizontal lines
End Type
'Type APIPoint
'   X As Integer
'   Y As Integer
'End Type
'Type APIRect
'   Left   As Integer
'   Top    As Integer
'   Right  As Integer
'   Bottom As Integer
'End Type




'
'Geometry Library by GWPouch 940331
'Geometry Library modified by GWPouch 940428 to allow huge arrays to actually work.
'  For completely incomprehensible reasons, you can't get a pointer from Windows using GlobalLock,
'   manipulate that pointer using integer*4 arithmetic, then pass it to a DLL subroutine
'   without generating General Protection Failures when the offset extends the pointer past a segment boundary.  BAD!
' You can get the pointer, pass it to the subroutine along with an offset, and all is well.
' That's what all the HUGESubName routines do.
' On 940620, I added two routines, one to map RealPoints to APIPoints, for
'   use with Windows API calls like Polygon and Polyline, and the other to map
'   to map APIPoints to RealPoints; most likely to be used to decode
'   coordinates in Windows metafiles.
```

```
53|
54|
55|
56| 'Split a line-segment into subsegments clipped at a polygon's edges
57| Declare Function ClipLineToPolygon Lib "WhtGeom2.DLL" (P1 As RealPoint, P2 As
    RealPoint, nPts As Long, Poly As RealPoint, Crossings As RealPoint) As Long
58| Declare Function HUGEClipLineToPolygon Lib "WhtGeom2.DLL" (P1 As RealPoint, P2
    As RealPoint, nPts As Long, ByVal ptrToOutline As Long, nPtsTotal As Long, nStart As
    Long, ByVal ptrToCrossings As Long, nCrossingPointsTotal As Long, nCrossingStart As
    Long)
59| ' Note that P1 and P2 are really just points, but that Poly and Crossings specifiy the first
    point in an array of length nPts.
60|
61| 'Find the bounding rectangle of a polygon
62| Declare Sub GetPolyBounds Lib "WhtGeom2.DLL" (PolyFirstPoint As RealPoint, nPts As
    Long, Bounds As RealRect)
63| Declare Sub HUGEGetPolyBounds Lib "WhtGeom2.DLL" (ByVal ptrToRealPoints As
    Long, nPtsTotal As Long, nFirstPoint As Long, nPts As Long, Bounds As RealRect)
64|
65| 'Find closest point, using simple Euclidean Distance.
66| Declare Sub FindClosestPoint Lib "WhtGeom2.DLL" (TestPoint As RealPoint, Points As
    RealPoint, nPoints As Long, IndexClosest As Long, Dist2Closest As Single, PointClosest
    As RealPoint)
67| Declare Sub HUGEFindClosestPoint Lib "WhtGeom2.DLL" (TestPoint As RealPoint,
    ByVal ptrToRealPoints As Long, nPtsTotal As Long, nStartAt As Long, nPointsToLookAt
    As Long, IndexClosest As Long, Dist2Closest As Single, PointClosest As RealPoint)
68|
69| Declare Function IndexOfClosestPoint Lib "WhtGeom2.DLL" (TestPoint As RealPoint,
    Points As RealPoint, nPoints As Long) As Long
70| Declare Function HUGEIndexOfClosestPoint Lib "WhtGeom2.DLL" (TestPoint As
    RealPoint, ByVal ptrToRealPoints As Long, nPtsTotal As Long, nStartAt As Long,
    nPoints As Long) As Long
71|
72| Declare Function Dist2ClosestPoint Lib "WhtGeom2.DLL" (TestPoint As RealPoint,
    Points As RealPoint, nPoints As Long) As Single
73| Declare Function HUGEDist2ClosestPoint Lib "WhtGeom2.DLL" (TestPoint As
    RealPoint, ByVal ptrToRealPoints As Long, nPtsTotal As Long, nStartAt As Long,
    nPoints As Long) As Single
74|
75|
76|
77|
78| 'Find the distance to a jointed line
79| Declare Sub Dist2JointedLine Lib "WhtGeom2.DLL" (TestPoint As RealPoint, Segmnts
    As RealPoint, nCountThisLine As Long, nClosest As Long, Distance As Single)
80| Declare Sub HUGEDist2JointedLine Lib "WhtGeom2.DLL" (TestPoint As RealPoint,
    ByVal Segmnts As Long, nPtsTotal As Long, nStartThisLine As Long, nCountThisLine As
    Long, nClosest As Long, Distance As Single)
81|
82| 'Solve Point-In-Polygon problem
83| Declare Function IsPointInPoly Lib "WhtGeom2.DLL" (TestPoint As RealPoint, Poly As
    RealPoint, nPts As Long) As Long
84| Declare Function HUGEIsPointInPoly Lib "WhtGeom2.DLL" (TestPoint As RealPoint,
    ByVal ptrToRealPoints As Long, nPtsTotal As Long, nPolyStartsAt As Long,
```

```
    nPtsDefiningPolygon As Long) As Long
 85|
 86| Declare Sub TestPointsInPoly Lib "WhtGeom2.DLL" (Poly As RealPoint, nPolyPts As
    Long, TestPoints As RealPoint, nTestPoints As Long)
 87| Declare Sub HUGETestPointsInPoly Lib "WhtGeom2.DLL" (ByVal ptrToRealPoints As
    Long, nPtsTotal As Long, nPolyStartsAt As Long, nPolyPts As Long, ByVal
    ptrToRealPointsTest As Long, nTestPtsTotal As Long, nTestStartsAt As Long,
    nTestPoints As Long)
 88|
 89| 'Which points lie inside a rectangle (cardinally-oriented)
 90| Declare Sub PickPointsInRect Lib "WhtGeom2.DLL" (Rect As RealRect, Points As
    RealPoint, nPts As Long, lngFlag As Long)
 91| Declare Sub HUGEPickPointsInRect Lib "WhtGeom2.DLL" (Rect As RealRect, ByVal
    ptrToRealPoints As Long, nPtsTotal As Long, nStartLookingAt As Long, nPtsToConsider
    As Long, lngFlag As Long, lngFlagCount As Long, lngFlagStartAt As Long)
 92|
 93|
 94|
 95|
 96|
 97| Declare Function DistSQRDToSeg Lib "WhtGeom2.DLL" (TestPoint As RealPoint, P1 As
    RealPoint, P2 As RealPoint) As Single
 98| Declare Function FastCross1 Lib "WhtGeom2.DLL" (P1 As RealPoint, P2 As RealPoint,
    P3 As RealPoint, P4 As RealPoint, TCross As RealPoint, PCross As RealPoint) As Long
 99|
100| Declare Function IsPointInRect Lib "WhtGeom2.DLL" (TestPoint As RealPoint, Bounds
    As RealRect) As Long
101|
102| Declare Sub PointCopy Lib "WhtGeom2.DLL" (in As RealPoint, out As RealPoint)
103| Declare Sub PrepPolyForInteriorTesting Lib "WhtGeom2.DLL" (Poly As RealPoint,
    Boundary As BoundaryLine, nPts As Long, Bounds As RealRect)
104|
105| Declare Function DotProduct Lib "WhtGeom2.DLL" (A As RealPoint, B As RealPoint)
106| Declare Sub RectCheck Lib "WhtGeom2.DLL" Alias "RectCheck" (Rect As RealRect)
107| Declare Function DistSqr2Point Lib "WhtGeom2.DLL" (A As RealPoint, B As RealPoint)
    As Single
108| Declare Function Dist2Point Lib "WhtGeom2.DLL" (A As RealPoint, B As RealPoint) As
    Single
109|
110|
111|
112| Declare Function IsPointInCircle Lib "WhtGeom2.DLL" (TestPoint As RealPoint,
    CntrPoint As RealPoint, Radius As Single) As Long
113|
114|  '
115| 'subroutine ScaleI2ToR4( GDIRect,WorldRect, nPoints,GDIPts,RlPts)
116| Declare Sub ScaleI2ToR4 Lib "WhtGeom2.DLL" (GDIRect As APIRECT, WorldRect As
    RealRect, nPoints As Long, ByVal ptrToGDIPts As Long, ByVal ptrToRealPoints As
    Long)
117| Declare Sub ScaleI2ToR4LOCAL Lib "WhtGeom2.DLL" Alias "ScaleI2ToR4" (GDIRect
    As APIRECT, WorldRect As RealRect, nPoints As Long, GDIPts As APIPOINT,
    RealPoints As RealPoint)
118| Declare Sub ScaleR4ToI2 Lib "WhtGeom2.DLL" (WorldRect As RealRect, GDIRect As
    APIRECT, nPoints As Long, ByVal ptrToRealPoints As Long, ByVal ptrToGDIPts As
```

```
     Long)
 119 Declare Sub ScaleR4ToI2LOCAL Lib "WhtGeom2.DLL" Alias "ScaleR4ToI2"
     (WorldRect As RealRect, GDIRect As APIRECT, nPoints As Long, RIPnts As RealPoint,
     GDIPts As APIPOINT)
 120
 121 Declare Function SCScaleR4ToI2 Lib "WhtGeom2.DLL" (WorldRect As RealRect,
     GDIRect As APIRECT, nPoints As Long, RIPnts As RealPoint, n2 As Long, GDIPts As
     APIPOINT) As Long
 122 Declare Function SCR4Clip Lib "WhtGeom2.DLL" (P1In As RealPoint, P2In As
     RealPoint, ClipRect As RealRect, P1Out As RealPoint, P2Out As RealPoint) As Integer
 123 Declare Function SCCode Lib "WhtGeom2.DLL" (Pt As RealPoint, ClipRect As
     RealRect) As Integer
 124
 125 Declare Sub spsort Lib "NtLbSort.DLL" (X As Single, n As Long, IPerm As Long, KFlag
     As Long, ier As Long)
 126 'Produces an output Permutation array IPERM() sorted on input variable X(),
 127 '  such that X(IPERM(I)) <= X(IPERM(I+1)) (or vice versa)
 128    'Note that the Fortran subroutine want the address of the first element in
 129      'the (active part of) array, so you might pass it T(1) for a 1-based array.
 130 Declare Sub i2psort Lib "NtLbSort.DLL" (ix As Integer, n As Long, IPerm As Integer,
     KFlag As Long, ier As Long)
 131 Declare Sub i4psort Lib "NtLbSort.DLL" (ix As Long, n As Long, IPerm As Long, KFlag
     As Long, ier As Long)
 132 Declare Sub dpsort Lib "NtLbSort.DLL" (DX As Double, n As Long, IPerm As Long,
     KFlag As Long, ier As Long)
 133 'declare SUB HPSORT lib "GEOMLIB1" (byHX, N, STRBEG, STREND, IPERM, KFLAG,
     work, IER)
 134 'Probably cannot use HPSort, which deals well with FORTRAN character variables, with
     Basic-style or C-style strings.
 135
 136 Declare Sub dsort Lib "NtLbSort.DLL" (DX As Double, DY As Double, n As Long, KFlag
     As Long)
 137 Declare Sub ssort Lib "NtLbSort.DLL" (X As Single, Y As Single, n As Long, KFlag As
     Long)
 138 Declare Sub i2sort Lib "NtLbSort.DLL" (ix As Integer, IY As Integer, n As Long, KFlag As
     Long)
 139 Declare Sub i4sort Lib "NtLbSort.DLL" (ix As Long, IY As Long, n As Long, KFlag As
     Long)
 140
 141
 142
 143
 144 Dim InData(4) As RealPoint
 145 Dim nInData As Long
 146
 147 Global A1 As RealPoint
 148 Global A2 As RealPoint
 149 Global B1 As RealPoint
 150 Global B2 As RealPoint
 151 Global Time1 As RealPoint
 152 Global PCross As RealPoint
 153
 154
 155 Global RL(1 To 10) As Single
```

```
156| Global R2(0 To 5)  As Single
157| Global r3(-3 To 2) As Single
158| Dim dummyLong As Long
159|
160|
161|
162|
163| Global J1(90) As Long, J2(90) As Long
164| Global h1(90) As Integer, h2(90) As Integer
165|
166| Global s1(90) As Single, s2(90) As Single
167| Global d1(90) As Double, d2(90) As Double
168|
169| Global ha As Long, hb As Long, hc As Long, hd As Long
170|
171| Global hGMemGDIPoints As Single
172|
173| Sub DoNothing ()
174|
175| End Sub
176|
177| Function ForceBLOBToBePolygon (BLOBIn As String) As String
178|  If Left$(BLOBIn, 8) <> Right$(BLOBIn, 8) Then
179|      ForceBLOBToBePolygon = BLOBIn & Left$(BLOBIn, 8)
180|     Else
181|      ForceBLOBToBePolygon = BLOBIn
182|  End If
183| End Function
184|
185| Sub GenerateHatchLines (Alpha As Single, Spacing As Single, Bounds As RealRect,
    CrosslinePts() As RealPoint)
186| 'Alpha is the rotation of the new X-axis relative to the old, in radians.
187| 'Spacing is the distance along the new Y-axis for the lines.
188| 'CrosslinePts contains pairs of lines, going from NexXMin to NewXMax.
189|
190|
191| 'NewX=  OldX * cos(Alpha) + OldY * sin(Alpha)
192| 'NewY= -OldX * sin(Alpha) + OldY * cos(Alpha)
193| 'OldX=  NewX * cos(Alpha) - NewY * sin(Alpha)
194| 'OldY=  NewX * sin(Alpha) + NewY * cos(Alpha)
195| Dim cosAlpha As Single, sinAlpha As Single
196| cosAlpha = Cos(Alpha)
197| sinAlpha = Sin(Alpha)
198|
199| Dim aLR As RealPoint, aLL As RealPoint, aUL As RealPoint, aUR As RealPoint
200| 'Lower Right
201| aLR.X = Bounds.Right * cosAlpha + Bounds.Bottom * sinAlpha
202| aLR.Y = -Bounds.Right * sinAlpha + Bounds.Bottom * cosAlpha
203| 'Lower left
204| aLL.X = Bounds.Left * cosAlpha + Bounds.Bottom * sinAlpha
205| aLL.Y = -Bounds.Left * sinAlpha + Bounds.Bottom * cosAlpha
206| 'Upper Left
207| aUL.X = Bounds.Left * cosAlpha + Bounds.Top * sinAlpha
208| aUL.Y = -Bounds.Left * sinAlpha + Bounds.Top * cosAlpha
```

```
209  'Upper Right
210  aUR.X = Bounds.Right * cosAlpha + Bounds.Top * sinAlpha
211  aUR.Y = -Bounds.Right * sinAlpha + Bounds.Top * cosAlpha
212
213  Dim NewMinX As Single, NewMinY As Single, NewMaxX As Single, NewMaxY As Single
214  NewMinX = 1E+38
215  NewMinY = 1E+38
216  NewMaxX = -1E+38
217  NewMaxY = -1E+38
218  NewMinX = MinR4(NewMinX, aLR.X)
219  NewMinX = MinR4(NewMinX, aLL.X)
220  NewMinX = MinR4(NewMinX, aUR.X)
221  NewMinX = MinR4(NewMinX, aUL.X)
222
223  NewMinY = MinR4(NewMinY, aLR.Y)
224  NewMinY = MinR4(NewMinY, aLL.Y)
225  NewMinY = MinR4(NewMinY, aUR.Y)
226  NewMinY = MinR4(NewMinY, aUL.Y)
227
228  NewMaxX = MaxR4(NewMaxX, aLR.X)
229  NewMaxX = MaxR4(NewMaxX, aLL.X)
230  NewMaxX = MaxR4(NewMaxX, aUR.X)
231  NewMaxX = MaxR4(NewMaxX, aUL.X)
232
233  NewMaxY = MaxR4(NewMaxY, aLR.Y)
234  NewMaxY = MaxR4(NewMaxY, aLL.Y)
235  NewMaxY = MaxR4(NewMaxY, aUR.Y)
236  NewMaxY = MaxR4(NewMaxY, aUL.Y)
237
238  Dim NewDeltX As Single, NewOffset As RealPoint, OldOffset As RealPoint
239  NewDeltX = NewMaxX - NewMinX
240  OldOffset.X = NewDeltX * cosAlpha
241  OldOffset.Y = NewDeltX * sinAlpha
242  Dim OldLRX As RealPoint
243  OldLRX.X = NewMinX * Cos(Alpha)
244  OldLRX.Y = NewMinX * Sin(Alpha)
245
246  Dim nSteps As Long, n As Long, NewY As Single, NewX As Single
247  nSteps = (NewMaxY - NewMinY) / Spacing + 1
248  ReDim CrosslinePts(1 To nSteps * 2)
249
250  For n = 1 To nSteps * 2 - 1 Step 2
251     NewX = NewMinX
252     NewY = Spacing * n / 2 + NewMinY
253     CrosslinePts(n).X = NewX * cosAlpha - NewY * sinAlpha
254     CrosslinePts(n).Y = NewX * sinAlpha + NewY * cosAlpha
255     NewX = NewMaxX
256     CrosslinePts(n + 1).X = NewX * cosAlpha - NewY * sinAlpha
257     CrosslinePts(n + 1).Y = NewX * sinAlpha + NewY * cosAlpha
258
259     'CrosslinePts(n).X = OldLRX.X - NewY * sinAlpha
260     'CrosslinePts(n).Y = OldLRX.Y + NewY * cosAlpha
261     'CrosslinePts(n + 1).X = CrosslinePts(n).X + OldOffset.X
262     'CrosslinePts(n + 1).Y = CrosslinePts(n).Y + OldOffset.Y
```

```
263|  Next n
264|
265|
266|  End Sub
267|
268|  Sub MagnifyRectangle (RectangleIn As RealRect, RectangleOut As RealRect, Multiplier
       As Single)
269|  Dim RangeX As Single, RangeY As Single
270|  'Expands (Mult>0) or contracts (Mult<0) a rectangle.
271|
272|
273|
274|  RangeX = RectangleIn.Right - RectangleIn.Left
275|  RangeY = RectangleIn.Top - RectangleIn.Bottom
276|
277|  RectangleOut.Top = RectangleIn.Top + RangeY * Multiplier
278|  RectangleOut.Bottom = RectangleIn.Bottom - RangeY * Multiplier
279|  RectangleOut.Right = RectangleIn.Right + RangeX * Multiplier
280|  RectangleOut.Left = RectangleIn.Left - RangeX * Multiplier
281|
282|
283|  End Sub
284|
285|  Function MakeAPICopyOfRealPoints (hGMemReal As Integer, WorldRect As RealRect,
       GDIRect As APIRECT) As Integer
286|  'GWPouch 940620
287|  'This function maps the coordinates of points living in the Global RealPoints memory
       block
288|  '   to API compatible integer*2 points living in the global APIPoints memory block
       hGMemGDI
289|  ' The return value of this function is hGMemGDI, a HANDLE to the global
290|  '   memory block containing the API points.
291|  ' You must lock it down before using to generate pointers to pass directly
292|  '   to the Windows API.  You could copy points out of it with MemWrite
293|  ' DISPOSE of it later by calling GlobalFree(hGMemGDI)
294|  ' Failure to properly dispose of this memory block will eventually choke
295|  '   the current Windows session.
296|
297|
298|
299|
300|  Dim hGMemGDI As Integer, ptrToRealPoints As Long, ptrToGDIPoints As Long
301|  Dim n As Integer, nPointsInMemBlock As Long
302|
303|  'Get the global memory to put the API Points into.
304|  '   This will be half as much as that needed for the RealPoints.
305|  nPointsInMemBlock = GlobalSize(hGMemReal)
306|  nPointsInMemBlock = nPointsInMemBlock \ 2
307|  hGMemGDI = GetWinMem(nPointsInMemBlock)
308|
309|
310|  'Lock both memory blocks down,so that pointers remain valid.
311|  ptrToRealPoints = GlobalLock(hGMemReal)
312|  ptrToGDIPoints = GlobalLock(hGMemGDI)
```

```
313|
314| 'Call the subroutine to do the mapping.
315| Call ScaleR4To12(WorldRect, GDIRect, hGMemLastPoint, ptrToRealPoints,
     ptrToGDIPoints)
316| 'Unlocak the memory blocks, so Windows will work
317| n = GlobalUnlock(hGMemReal)
318| n = GlobalUnlock(hGMemGDI)
319| MakeAPICopyOfRealPoints = hGMemGDI
320|
321|
322| End Function
323|
324| Private Function MaxR4 (A As Single, B As Single) As Single
325| If A > B Then
326|    MaxR4 = A
327|   Else
328|    MaxR4 = B
329| End If
330| End Function
331|
332| Private Function MinR4 (A As Single, B As Single) As Single
333| If A < B Then
334|    MinR4 = A
335|   Else
336|    MinR4 = B
337| End If
338| End Function
339|
340| Sub PolygonToGrid (InPoly() As RealPoint, Value As Variant, hGMemGrid As Integer,
     GridDesc As typGridDescription)
341| 'A subroutine to rasterize a polygon into a grid.
342| 'Greg Pouch 940813 at home.
343| 'This subroutine is descended from ShadeInPolygon (see Geomlib1.bas)
344|
345| ' First, it creates a set of line segments passing through the polygon,
346| '  corresponding to horizontal lines through cell centers.
347| ' Then, it clips those lines and loads the subsegments into the array HatchLines.
348| ' To display them, draw lines from the first to second, third to fourth, etc points
349| '  in the array HatchLines()
350| Dim nPoly As Long, lbPoly As Long, nPolyHolder  As Long
351| 'Dim nSteps As Long, nS As Long, nHtchLines As Long, Theta As Single
352| Dim nX As Long, nP As Long, nH As Long, BoundBox As RealRect, nPxl As Long
353| On Error GoTo ERRPolygonToGrid
354|
355| lbPoly = LBound(InPoly)
356| 'If (InPoly(lbPoly).X <> InPoly(UBound(InPoly)).X) Or (InPoly(lbPoly).Y <>
     InPoly(UBound(InPoly)).Y) Then
357| '  'If the polygon doesn't close, force it!
358| '  ReDim Preserve InPoly(lbPoly To (UBound(InPoly) + 1))
359| '  Call PointCopy(InPoly(lbPoly), InPoly(UBound(InPoly)))
360| 'End If
361| nPoly = UBound(InPoly) - lbPoly + 1
362|
363| Call GetPolyBounds(InPoly(lbPoly), nPoly + 1, BoundBox)
```

```
364| Call MagnifyRectangle(BoundBox, BoundBox, .1)
365| 'Create a set of HatchLines filling the bounding box of the polygon
366| ReDim CrossLines(1 To 2) As RealPoint, ClippedLines(1 To nPoly + 2) As RealPoint
367| Dim nL As Long, nLines As Long, Offset As Long, BaseOffset As Long
368|
369| For nL = GridRowNumFromYCoord(BoundBox.Bottom, GridDesc) To
    GridRowNumFromYCoord(BoundBox.Top, GridDesc)
370|   If nL < 0 Or nL > GridDesc.nRowsY Then GoTo BottomLineLoop
371|   'CrossLines(1).X = BoundBox.Left: CrossLines(2).X = BoundBox.Right
372|   CrossLines(1).X = GridDesc.X_Origin: CrossLines(2).X = GridDesc.X_Origin +
    (GridDesc.nColsX + 1) * GridDesc.DeltaX
373|   CrossLines(1).Y = GridDesc.Y_Origin + nL * GridDesc.DeltaY: CrossLines(2).Y =
    CrossLines(1).Y
374|   nPolyHolder = nPoly
375|   nX = ClipLineToPolygon(CrossLines(1), CrossLines(2), nPolyHolder, InPoly(lbPoly),
    ClippedLines(1))
376|   DoEvents
377|   If nX Mod 2 Then
378|   Call ShowProblemPolygon(InPoly(), ClippedLines(), nX)
379|     Dim MinSep As Single
380|  ' dDebug.Print "Encountered an odd number of points " & nX & " processing line #" &
    nL
381|  MinSep = 1E+34
382|  nP = 1
383|  Do While nP < nX
384|       If ClippedLines(nP + 1).X - ClippedLines(nP).X < MinSep Then MinSep =
    ClippedLines(nP + 1).X - ClippedLines(nP).X
385|  '        dDebug.Print ClippedLines(nP).X, ClippedLines(nP + 1).X, ClippedLines(nP).X
    - ClippedLines(nP + 1).X
386|      If ClippedLines(nP + 1).X = ClippedLines(nP).X Then
387|  '        dDebug.Print nP, nP + 1, ClippedLines(nP).X, ClippedLines(nP + 1).X
388|        For nPxl = nP To nX - 1
389|       ClippedLines(nPxl) = ClippedLines(nPxl + 1)
390|        Next nPxl
391|      nX = nX - 1
392|      End If
393|      nP = nP + 1
394|  Loop
395|
396|  If nX Mod 2 Then 'MsgBox "Still have problems gridding polygons." & "Somehow,
    ClipLineToPolygon returned " & nX & " points. Line#" & nL & " In
    GRIDHNDL.BAS:PolygonToGrid"
397|  nP = 1
398|  Do While nP < nX
399|     If ClippedLines(nP + 1).X - ClippedLines(nP).X = MinSep Then
400|  '        dDebug.Print nP, nP + 1, ClippedLines(nP).X, ClippedLines(nP + 1).X
401|        For nPxl = nP To nX - 1
402|       ClippedLines(nPxl) = ClippedLines(nPxl + 1)
403|        Next nPxl
404|      nX = nX - 1
405|      End If
406|      nP = nP + 1
407|  Loop
408|  End If
```

```
409|  '    Call ShowProblemPolygon(InPoly(), ClippedLines(), nX)
410|     End If
411|
412|     If nX > 0 Then
413|      'Raterize the subsegments
414|        For nP = 1 To nX - 1 Step 2
415|       For nPxl = GridColNumFromXCoord(ClippedLines(nP).X, GridDesc) To
      GridColNumFromXCoord(ClippedLines(nP + 1).X, GridDesc)
416|           If (nPxl > -1) And (nPxl < GridDesc.nColsX) Then 'Use -1 and nColsX because it's
      zero-based.
417|           Call SetIJCellGenericInGlobalMem(nPxl, nL, GridDesc, hGMemGrid, Value)
418|           End If
419|       Next nPxl
420|        Next nP
421|    End If
422|  BottomLineLoop:
423|  Next nL
424|
425|  Exit Sub
426|
427|
428|  ERRPolygonToGrid:
429|     MsgBox "Error number " & Err & " occurred while rasterizing polygons into Grid." &
      CRLF & Error
430|     Err = 0
431|     Exit Sub
432|     Resume
433|
434|  End Sub
435|
436|  Sub ShadeInPolygon (InPoly() As RealPoint, Bearing As Single, Spacing As Single,
      HatchLinesOut() As RealPoint)
437|  ' A subroutine to fill in a polygon with hatch lines (actually, it generates the
438|  '     the hatchlines; some other subroutine/function has to draw them)
439|  '
440|  ' First, it creates a set of line segments passing through the polygon, with
441|  '  specified orientation and spacing.
442|  ' Then, it clips those lines and loads the subsegments into the array HatchLines.
443|  ' To display them, draw lines from the first to second, third to fourth, etc points
444|  '   in the array HatchLines()
445|
446|  ' Greg Pouch   940408
447|
448|  'Bearing is compass bearing (0 to 360) measured clockwise from north, in degrees.
449|  '     East is 90, south is 180, etc.
450|  'InPoly() is a dynamically-allocated [redim PolygonSource() as RealPoint] array of
      RealPoints describing a closed polygon.
451|  'Spacing is the distance (perpindicular to Bearing) between hatch lines.
452|  'HatchLinesOut() is a dynamically allocated array of RealPoints which, on output,
453|  '   contains the hatch lines.  Let nSeg=nPts\2.  Then hatching is done by
454|  '    drawing a line from HatchLinesOut(nS) to HatchLinesOut(nS+1), for nS=1 to nSeg
455|
456|  Dim nPoly As Long, lbPoly As Long, nPolyHolder  As Long
457|  Dim nSteps As Long, nS As Long, nHtchLines As Long, Theta As Single
```

```
458| Dim nX As Long, nP As Long, nH As Long, BoundBox As RealRect
459| 'On Error GoTo ERRShadeInPolygon
460|
461| 'Convert Theta from compass-degrees bearing to math-radians
462| '  PI/180= #radians in a degree=  0.01745329251994
463| Theta = .01745329251994 * (90 - Bearing)
464| If (Spacing = 0) Then Spacing = 1#  'Check that Spacing is reasonable.
465|
466| lbPoly = LBound(InPoly)
467| If (InPoly(lbPoly).X <> InPoly(UBound(InPoly)).X) Or (InPoly(lbPoly).Y <>
    InPoly(UBound(InPoly)).Y) Then
468|   'If the polygon doesn't close, force it!
469|   ReDim Preserve InPoly(lbPoly To (UBound(InPoly) + 1))
470|   Call PointCopy(InPoly(lbPoly), InPoly(UBound(InPoly)))
471| End If
472| nPoly = UBound(InPoly) - lbPoly + 1
473|
474| Call GetPolyBounds(InPoly(lbPoly), nPoly + 1, BoundBox)
475| Call MagnifyRectangle(BoundBox, BoundBox, .1)
476| 'Create a set of HatchLines filling the bounding box of the polygon
477| ReDim CrossLines(0 To 0) As RealPoint
478| Call GenerateHatchLines(Theta, Spacing, BoundBox, CrossLines())
479| 'Generate Hatchlines puts the first point in location 1
480|
481| nHtchLines = 0
482| ReDim HatchLinesOut(1 To 1)
483| nSteps = (UBound(CrossLines) - LBound(CrossLines)) / 2
484| ReDim ClippedLines(1 To nPoly + 2) As RealPoint
485| For nS = 0 To nSteps - 1
486|   nPolyHolder = nPoly 'Save nPoly in nPolyHolder
487|   'Clip the line segment to the polygon.
488|   nX = ClipLineToPolygon(CrossLines(nS * 2 + 1), CrossLines(nS * 2 + 2), nPolyHolder,
    InPoly(lbPoly), ClippedLines(1))
489|   If nX > 0 Then
490|     nH = nHtchLines
491|     nHtchLines = nHtchLines + nX
492|     ReDim Preserve HatchLinesOut(1 To nHtchLines)
493|    'Copy the subsegments to the output array HatchLines
494|     For nP = 1 To nX
495|     Call PointCopy(ClippedLines(nP), HatchLinesOut(nH + nP))
496|     Next nP
497|   End If
498| Next nS
499|
500| If nHtchLines = 0 Then ReDim HatchLinesOut(0 To 0) 'This shouldn't happen, but you
    never know.
501| Exit Sub
502| ERRShadeInPolygon:
503| MsgBox "Error number " & Err & " occurred while shading in polygons." & CRLF & Error
504| Err = 0
505| Exit Sub
506| Resume Next
507| End Sub
508|
```

```
Sub SpitOutRLPTArr (RLPTArr() As RealPoint)
  Dim n As Long
  For n = LBound(RLPTArr) To UBound(RLPTArr)
    Debug.Print n, RLPTArr(n).X, RLPTArr(n).Y
  Next n
End Sub

Sub ZClipLinesToPolygon ()
'integer*4 Function
'    & ClipLineToPolygon (P1, P2, nPts, Poly, Crossings)
'    integer*4 nPts
'    real*4  A, B , C
'    structure /RealPoint/
'      real*4  X
'      real*4   Y
"    end structure
'    structure /RealRect/
'     real*4 Left
'     real*4 Right
'     real*4 Bottom
'     real*4 Top
'    end structure
'    structure /BoundaryLine/
'      real*4 m
'      character*1 h
'    end structure
'
'    record /RealPoint/ P1, P2, Poly(nPts),Crossings(nPts+2)
'
End Sub

Sub ZDefine_ztblWHEATLOG (DBIn As Database)
On Error GoTo ERRDefine_ztblWHEATLOG
'GWPouch 940830
'Defines a table to receive minimalist information about contours


Dim NewTable As New Tabledef
NewTable.Name = "ztblWHEAT_LOG"

Dim nfSerial As New field
nfSerial.Type = DB_LONG
nfSerial.Name = "SerialNumber"
NewTable.Fields.Append nfSerial

Dim nfWhen As New field
nfWhen.Type = DB_DATE
nfWhen.Name = "TimeStamp"
NewTable.Fields.Append nfWhen

Dim nfInputs As New field
nfInputs.Type = DB_MEMO
nfInputs.Name = "Inputs"
NewTable.Fields.Append nfInputs
```

```
563|
564| Dim nfOutputs As New field
565| nfOutputs.Type = DB_MEMO
566| nfOutputs.Name = "Outputs"
567| NewTable.Fields.Append nfOutputs
568|
569| Dim nfProgramName As New field
570| nfProgramName.Type = DB_TEXT
571| nfProgramName.Size = 255
572| nfProgramName.Name = "ProgramName"
573| NewTable.Fields.Append nfProgramName
574|
575| Dim nfUserName As New field
576| nfUserName.Type = DB_TEXT
577| nfUserName.Size = 255
578| nfUserName.Name = "UserName"
579| NewTable.Fields.Append nfUserName
580|
581|
582|
583|
584| DBIn.TableDefs.Append NewTable
585|
586| Exit Sub
587|
588|
589| ERRDefine_ztblWHEATLOG:
590| Select Case Err
591|  Case 3010 'Table of that name already exists
592|    Err = 0
593|    Exit Sub
594|    Resume
595|  Case 3125
596|    'TableOutName = InputBox$(TableOutName & " is not a valid name.  Please choose a
     new one.", "Saving Contours to Table", TableOutName)
597|    'NewTable.Name = TableOutName
598|    'Resume
599|  Case Else
600|    MsgBox "Error #" & Err & " occured in Contours.bas DEFINECONTOURTABLE." &
     CRLF & Error
601|    Err = 0
602|    Exit Sub
603|    Resume
604| End Select
605|
606|
607|
608| End Sub
609|
610| Sub ZDist2ClosestPoint ()
611| 'c **** function Dist2ClosestPoint (TestPoint, Points, nPoints)
612| '     real*4 function Dist2ClosestPoint (TestPoint, Points, nPoints)
613| '     integer*4 nPoints
614| '     structure /RealPoint/
```

```
615I '    real*4 X
616I '    real*4 Y
617I '      end structure
618I '      record /RealPoint/ TestPoint, Points(nPoints)
619I '      integer*4 n
620I '
621I End Sub
622I
623I Sub ZDist2JointedLine ()
624I 'c ***** subroutine Dist2JointedLine(TestPoint, Segmnts, nSegs,nClosest, Distance) *****
625I '      subroutine Dist2JointedLine(TestPoint, Segmnts, nPtsTotal,
626I '     &     nStartThisLine, nCountThisLine, nClosest, Distance)
627I '      real*4 Distance
628I '      integer*4 nPtsTotal, nClosest, nStartThisLine, nCountThisLine,m
629I '      structure /RealPoint/
630I '    real*4 X
631I '    real*4 Y
632I '      end structure
633I '      record /RealPoint/ TestPoint, Segmnts(nPtsTotal)
634I '      record /RealPoint/  PtA, V
635I '      real*4 MinDist
636I '      real*4 A,B,C,D,F,VSqr
637I '
638I
639I
640I 'Old version
641I 'c ***** subroutine Dist2JointedLine(TestPoint, Segments, nSegs,nClosest, Distance)
     *****
642I '      subroutine Dist2JointedLine(TestPoint, Segments, nSegs,
643I '     &     nClosest, Distance)
644I '      real*4 Distance
645I '      integer*4 nSegs, nClosest
646I '      structure /RealPoint/
647I '    real*4 X
648I '    real*4 Y
649I '      end structure
650I '      record /RealPoint/ TestPoint, Segments(nSegs)
651I '      real*4 Dist, MinDist
652I '      real*4 DistSQRDToSeg
653I '      integer*4 n
654I '
655I End Sub
656I
657I Sub zDist2Point ()
658I 'c *****  function Dist2Point(A,B) *****
659I '      real*4 function Dist2Point(A,B)
660I
661I
662I End Sub
663I
664I Sub ZDistSqrdToLine ()
665I 'real*4 function DistSQRDToSeg (TestPoint, P1 , P2 )
666I '      structure /RealPoint/
667I '        real*4  X
```

```
668|  '        real*4  Y
669|  '     end structure
670|  '     record /RealPoint/TestPoint, P1, P2, PtA, V
671|  '
672|  End Sub
673|
674|  Sub ZDotProduct ()
675|  'real*4 function DotProduct(A,B)
676|  '     structure /RealPoint/
677|  "    real*4 X
678|  '    real*4 Y
679|  '     end structure
680|  '     record/RealPoint/ A,
681|  '
682|  End Sub
683|
684|  Sub ZFastCross1 ()
685|  'c ****** Function FastCross1 (P1,P2,P3,P4,TCross,PCross) **********
686|  '     Integer*4 Function FastCross1 (P1,P2,P3,P4,TCross,PCross)
687|  '     structure /RealPoint/
688|  '        real*4  X
689|  '        real*4  Y
690|  '     end structure
691|  '     structure /RealRect/
692|  '      real*4 Left
693|  '      real*4 Right
694|  '      real*4 Bottom
695|  '      real*4 Top
696|  '     end structure
697|  '
698|  '     record/RealPoint/ P1 ,P2, P3, P4, TCross, PCross
699|  ''
700|  End Sub
701|
702|  Sub ZFindClosestPoint ()
703|  'c ***Subroutine FindClosestPoint (TestPoint, Points, nPoints, IndexClosest, Dist2Closest,
      PointClosest) ****
704|  '     Subroutine FindClosestPoint (TestPoint, Points, nPoints,
705|  '    &      IndexClosest, Dist2Closest, PointClosest)
706|  '     integer*4 nPoints,IndexClosest
707|  '     structure /RealPoint/
708|  '    real*4 X
709|  '    real*4 Y
710|  '     end structure
711|  '     record /RealPoint/ TestPoint, Points(nPoints),PointClosest
712|  '     integer*4 n, NClosest
713|  '     real*4 MinDist, Dist, Dist2Closest
714|  '
715|  End Sub
716|
717|  Sub ZGetPolyBounds ()
718|  'c ****** GetPolyBounds (Poly() As RealPoint,nPts, Bounds) ******
719|  '     Subroutine GetPolyBounds(Poly, nPts, Bounds)
720|  '     structure /RealPoint/
```

```
721| '        real*4  X
722| '        real*4   Y
723| '     end structure
724| '     structure /RealRect/
725| '      real*4 Left
726| '      real*4 Right
727| '      real*4 Bottom
728| '      real*4 Top
729| ''    end structure
730| ''    structure /BoundaryLine/
731| '        real*4 m
732| '        character*1 h
733| '     end structure
734| '
735| '     integer*4 nPts
736| '     record /RealRect/ Bounds
737| '     record /RealPoint/ Poly(nPts)
738| '
739| End Sub
740|
741| Sub ZIndexOfClosestPoint ()
742| 'c **** function IndexOfClosestPoint(TestPoint,Points,nPoints) ****
743| '     Integer*4 function IndexOfClosestPoint(TestPoint,Points,nPoints)
744| '     integer*4 nPoints
745| '     structure /RealPoint/
746| '    real*4 X
747| '    real*4 Y
748| '     end structure
749| '     record /RealPoint/ TestPoint, Points(nPoints)
750| '     integer*4 n, NClosest
751| '     real*4 MinDist, Dist
752| '
753| End Sub
754|
755| Sub ZIsPointInPoly ()
756| 'c **** function IsPointInPoly (TestPoint, Poly,nPts) ****
757| '     integer*4 Function IsPointInPoly (TestPoint, Poly,nPts)
758| '     structure /RealPoint/
759| '        real*4  X
760| '        real*4   Y
761| '     end structure
762| '     structure /RealRect/
763| '      real*4 Left
764| '      real*4 Right
765| '      real*4 Bottom
766| '      real*4 Top
767| '     end structure
768| '
769| ''    record /RealPoint/ TestPoint, Poly(nPts)
770| '     integer*4 nPts
771| '
772| End Sub
773|
774| Sub ZIsPointInRect ()
```

```
775| 'c **** Function IsPointInRect (TestPoint, Bounds) ****
776| '     integer*4 Function IsPointInRect (TestPoint, Bounds)
777| '     structure /RealPoint/
778| '        real*4  X
779| '        real*4  Y
780| '     end structure
781| '     structure /RealRect/
782| '      real*4 Left
783| '      real*4 Right
784| '      real*4 Bottom
785| '      real*4 Top
786| '     end structure
787| '     record /RealPoint/ TestPoint
788| '     record /RealRect/ Bounds
789| '
790| End Sub
791|
792| Sub ZPickPointsInRect ()
793| 'c ***** Subroutine PickPointsInRect (Rect , Points, nPts, lngFlag) *****
794| '     Subroutine PickPointsInRect(Rect, Points, nPts, lngFlag)
795| '     structure /RealPoint/
796| '        real*4  X
797| '        real*4  Y
798| '     end structure
799| '     structure /RealRect/
800| '      real*4 Left
801| '      real*4 Right
802| '      real*4 Bottom
803| '      real*4 Top
804| '     end structure
805| '    structure /BoundaryLine/
806| '        real*4 m
807| '        character*1 h
808| '     end structure
809| '     record /RealRect/ Rect
810| '     record /RealPoint/ Points(nPts)
811| '     integer*4 lngFlag(nPts)
812| '     integer*4 n
813|
814| End Sub
815|
816| Sub ZPointCopy ()
817| ' **** Subroutine PointCopy (In, Out) ****
818| '     Subroutine PointCopy(In, Out)
819| '     structure /RealPoint/
820| '        real*4  X
821| '        real*4  Y
822| '     end structure
823| '
824| End Sub
825|
826| Sub ZPrepPolyForInteriorTesting ()
827| 'c ***** subroutine PrepPolyForInteriorTesting (Poly,Boundary,nPts,Bounds) *****
828| '     subroutine PrepPolyForInteriorTesting(Poly, Boundary, nPts, Bounds)
```

```
829|  '     structure /RealPoint/
830|  '       real*4  X
831|  '       real*4   Y
832|  '     end structure
833|  '     structure /RealRect/
834|  '      real*4 Left
835|  '      real*4 Right
836|  '      real*4 Bottom
837|  '      real*4 Top
838|  '     end structure
839|  '     structure /BoundaryLine/
840|  '       real*4 m
841|  '       character*1 h
842|  '     end structure
843|  '
844|  '     record/RealPoint/ Poly(nPts)
845|  '     record /BoundaryLine/ Boundary(nPts)
846|  '     record/RealRect/ Bounds
847|  '     integer*4 nPts
848|  '
849|  '     integer*4 nP
850|  '
851|  End Sub
852|
853|  Function zqrMaxR4 (A As Single, B As Single) As Single
854|  If A > B Then
855|     zqrMaxR4 = A
856|   Else
857|     zqrMaxR4 = B
858|  End If
859|  End Function
860|
861|  Function zqrMinR4 (A As Single, B As Single) As Single
862|  If A < B Then
863|     zqrMinR4 = A
864|    Else
865|     zqrMinR4 = B
866|  End If
867|  End Function
868|
869|  Sub ZRectCheck ()
870|  'subroutine RectCkeck(Rect)
871|  '     structure /RealRect/
872|  '      real*4 Left
873|  '      real*4 Right
874|  '      real*4 Bottom
875|  '      real*4 Top
876|  "     end structure
877|  '     record /RealRect/ Rect
878|  '     real*4 Temp
879|  '
880|  End Sub
881|
882|  Sub ZTestPointsInPoly ()
```

```
883| 'c **** Subroutine TestPointsInPoly
884| "c      (Poly,nPolyPts, TestPoints, nTestPts,IngInPoly) ****
885| 'c
886| '     Subroutine TestPointsInPoly (Poly,nPolyPts, TestPoints, nTestPts,
887| '    &                             IngInPoly)
888| '     structure /RealPoint/
889| '        real*4  X
890| '        real*4   Y
891| '     end structure
892| '     structure /RealRect/
893| '      real*4 Left
894| '      real*4 Right
895| '      real*4 Bottom
896| '      real*4 Top
897| '     end structure
898| '     structure /BoundaryLine/
899| '        real*4 m
900| '        character*1 h
901| '     end structure
902| '     integer*4 nTestPts, nPolyPts
903| '     record /RealPoint/ Poly(nPolyPts), TestPoints(nTestPts)
904| '     integer*4 IngInPoly(nTestPts)
905| '     record /RealRect/ Rect
906| '     record /BoundaryLine/ Boundary[Allocatable,Huge](:)
907| 'c  Tests whether points in TestPoints lie in the polygon (closed loop)
908| 'c   defined by Poly.  On on output, IngInPoly contains a flag value indicating
909| 'c   whether the point lies in (TRUE) or out (.false.)  of the polygon
910| 'c
911| 'c  Based on FindPointsInPoly, by Greg Pouch (home) 930321
912| 'c  Modified to be more consistent with the rest of WHEAT 940330 by Greg Pouch (KGS)
913| '     integer*4   nPPt, nTPt, ier
914| '
915| End Sub
916|
917| Sub zzGenerateHatchLines (Alpha As Single, Spacing As Single, Bounds As RealRect,
      PointsOut() As RealPoint)
918| 'A subroutine to generate a set of hatch lines, at equal spacing, and at angle alpha to the
      x-axis.
919| ' This is used in creating a hatch fill of a polygon or other figure.
920| 'Greg Pouch 940403
921|
922| 'Alpha is the rotation of the new X-axis relative to the old, in radians.
923| 'Spacing is the distance along the new Y-axis for the lines.
924| 'PointsOut contains pairs of lines, going from NexXMin to NewXMax.
925|
926|
927| 'NewX=  OldX * cos(Alpha) + OldY * sin(Alpha)
928| 'NewY= -OldX * sin(Alpha) + OldY * cos(Alpha)
929| 'OldX=  NewX * cos(Alpha) - NewY * sin(Alpha)
930| 'OldY=  NewX * sin(Alpha) + NewY * cos(Alpha)
931| Dim cosAlpha As Single, sinAlpha As Single
932| cosAlpha = Cos(Alpha)
933| sinAlpha = Sin(Alpha)
934|
```

```
935  Dim aLR As RealPoint, aLL As RealPoint, aUL As RealPoint, aUR As RealPoint
936  'Lower Right
937  aLR.X = Bounds.Right * cosAlpha + Bounds.Bottom * sinAlpha
938  aLR.Y = -Bounds.Right * sinAlpha + Bounds.Bottom * cosAlpha
939  'Lower left
940  aLL.X = Bounds.Left * cosAlpha + Bounds.Bottom * sinAlpha
941  aLL.Y = -Bounds.Left * sinAlpha + Bounds.Bottom * cosAlpha
942  'Upper Left
943  aUL.X = Bounds.Left * cosAlpha + Bounds.Top * sinAlpha
944  aUL.Y = -Bounds.Left * sinAlpha + Bounds.Top * cosAlpha
945  'Upper Right
946  aUR.X = Bounds.Right * cosAlpha + Bounds.Top * sinAlpha
947  aUR.Y = -Bounds.Right * sinAlpha + Bounds.Top * cosAlpha
948
949  Dim NewMinX As Single, NewMinY As Single, NewMaxX As Single, NewMaxY As Single
950  NewMinX = 1E+38
951  NewMinY = 1E+38
952  NewMaxX = -1E+38
953  NewMaxY = -1E+38
954  NewMinX = MinR4(NewMinX, aLR.X)
955  NewMinX = MinR4(NewMinX, aLL.X)
956  NewMinX = MinR4(NewMinX, aUR.X)
957  NewMinX = MinR4(NewMinX, aUL.X)
958
959  NewMinY = MinR4(NewMinY, aLR.Y)
960  NewMinY = MinR4(NewMinY, aLL.Y)
961  NewMinY = MinR4(NewMinY, aUR.Y)
962  NewMinY = MinR4(NewMinY, aUL.Y)
963
964  NewMaxX = MaxR4(NewMaxX, aLR.X)
965  NewMaxX = MaxR4(NewMaxX, aLL.X)
966  NewMaxX = MaxR4(NewMaxX, aUR.X)
967  NewMaxX = MaxR4(NewMaxX, aUL.X)
968
969  NewMaxY = MaxR4(NewMaxY, aLR.Y)
970  NewMaxY = MaxR4(NewMaxY, aLL.Y)
971  NewMaxY = MaxR4(NewMaxY, aUR.Y)
972  NewMaxY = MaxR4(NewMaxY, aUL.Y)
973
974  Dim NewDeltX As Single, NewOffset As RealPoint, OldOffset As RealPoint
975  NewDeltX = NewMaxX - NewMinX
976  OldOffset.X = NewDeltX * cosAlpha
977  OldOffset.Y = NewDeltX * sinAlpha
978  Dim OldLRX As RealPoint
979  OldLRX.X = NewMinX * Cos(Alpha)
980  OldLRX.Y = NewMinX * Sin(Alpha)
981
982  Dim nSteps As Long, n As Long, NewY As Single, NewX As Single
983  nSteps = (NewMaxY - NewMinY) / Spacing + 1
984  ReDim PointsOut(0 To nSteps * 2)
985  For n = 0 To (nSteps * 2 - 1) Step 2
986     NewX = NewMinX
987     NewY = Spacing * n / 2 + NewMinY
988     PointsOut(n).X = NewX * cosAlpha - NewY * sinAlpha
```

```
 989|     PointsOut(n).Y = NewX * sinAlpha + NewY * cosAlpha
 990|     NewX = NewMaxX
 991|     PointsOut(n + 1).X = NewX * cosAlpha - NewY * sinAlpha
 992|     PointsOut(n + 1).Y = NewX * sinAlpha + NewY * cosAlpha
 993| Next n
 994|
 995|
 996| End Sub
 997|
 998| Sub zzListOfExportfromGEOMLIB1 ()
 999| '       ssort
1000| '       i4sort
1001| '       i2sort
1002| '       dsort
1003| '       spsort
1004| '       hpsort
1005| '       dpsort
1006| '       i2psort
1007| '       i4psort
1008| '       DotProduct
1009| '       DistSqr2Point
1010| '       Dist2Point
1011| '       GetPolyBounds
1012| '       PickPointsInRect
1013| '       PointCopy
1014| '       PrepPolyForInteriorTesting
1015| '       TestPointsInPoly
1016| '       ClipLineToPolygon
1017| '       FindClosestPoint
1018| '       IndexOfClosestPoint
1019| '       Dist2ClosestPoint
1020| '       DistSQRDToSeg
1021| '       Dist2JointedLine
1022| '       FastCross1
1023| '       IsPointInPoly
1024| '       IsPointInRect
1025| '       IsPointInCircle
1026| '       RectCkeck
1027| '
1028| End Sub
1029|
```

```
 1| Option Explicit
 2|
 3| Dim DummyInt As Integer
 4|
 5| Dim sngScaleFudgeFactor As Single
 6|
 7| Sub aJunk ()
 8| 'Declarations
 9| 'Dim flgDrawgraticule As Long, GraticuleSize  As Single, GraticuleColor As Long,
   GraticuleThick As Long
10|
11|
12| 'At startup
13| 'GraticuleColor = RGB(255, 255, 0): GraticuleThick = 1
14|
15|
16| 'In the drawing routine.  Note that the menu caption is reset frequently.
17| '       Me!zMnuView(200).Checked = Not Me!mvGraticule.Checked
18| '       flgDrawgraticule = Me!zMnuView(200).Checked
19| '       If Me!zMnuView(200).Checked Then
20| '          flgDrawgraticule = True
21| '          Call DrawGraticule(Me!picMap, GraticuleSize, graticuleColor,
   GraticuleThickness)
22| '          Me!zMnuView(200).Caption = "Map Graticule " & Format$(GraticuleSize,
   "0,000")
23| '       Else
24| '          flgDrawgraticule = False
25| '          Me!zMnuView(200).Caption = "Show Map Graticule"
26| '       End If
27|
28| End Sub
29|
30| Sub DrawGraticule (PIC As PictureBox, GraticuleSize As Single, Color As Long, Thick
   As Long)
31| " This function does two things.  It finds out how
32| " big grid cells should be and stores it in GraticuleSize, which can be used by the caller
   to display to user
33| " and it draws a map graticule in the color and thickness specified on the picturebox
34| Dim TempRect As RealRect
35|
36| TempRect.Left = PIC.ScaleLeft
37| TempRect.Right = TempRect.Left + PIC.ScaleWidth
38| TempRect.Top = PIC.ScaleTop
39| TempRect.Bottom = TempRect.Top + PIC.ScaleHeight
40|
41| GraticuleSize = FindGraticuleSize(TempRect)
42|
43|
44| Dim Low As Single, High As Single
45| Dim nYLow As Long, nYHigh As Long
46| Dim nXLow As Long, nXHigh As Long
47|
48| PIC.ForeColor = Color: PIC.DrawWidth = Thick
49|
```

```
'First, draw the graticule in X

Dim X As Single, Y As Single
nXLow = Int(TempRect.Left / GraticuleSize)
  If nXLow * GraticuleSize < TempRect.Left Then nXLow = nXLow + 1
nXHigh = Int(TempRect.Right / GraticuleSize) + 1
  If nXHigh * GraticuleSize > TempRect.Right Then nXHigh = nXHigh - 1

nYLow = Int(TempRect.Bottom / GraticuleSize)
  If nYLow * GraticuleSize < TempRect.Bottom Then nYLow = nYLow + 1
nYHigh = (Int(TempRect.Top / GraticuleSize) + 1)
  If nYHigh * GraticuleSize > TempRect.Top Then nYHigh = nYHigh - 1


Dim nX As Long, nY As Long

For nX = nXLow To nXHigh
  X = nX * GraticuleSize
  PIC.Line (X, TempRect.Bottom)-(X, TempRect.Top)
    'Should optionally display coordinates.  More important for printing and
    'copying than for live maps, where this info comes from mousemove
Next nX


For nY = nYLow To nYHigh
  Y = nY * GraticuleSize
  PIC.Line (TempRect.Left, Y)-(TempRect.Right, Y)
    'Should optionally display coordinates.  More important for printing and
    'copying than for live maps, where this info comes from mousemove
Next nY
Exit Sub

'PIC.FontName = "Arial"
'PIC.FontSize = 8
'PIC.FontBold = False: PIC.FontItalic = False
' nX = SetTextAlign(PIC.hDC, TA_TOP Or TA_LEFT)
'
'For nX = nXLow + 0 To nXLow + 2
'  X = nX * GraticuleSize
'  PIC.CurrentX = X
'  PIC.CurrentY = (nYHigh - .5) * GraticuleSize
'  PIC.Print Format$(X, "0,000")
'Next nX
'
'For nY = nYHigh - 0 To nYHigh - 2 Step -1
'  Y = nY * GraticuleSize
'  PIC.CurrentY = Y
'  PIC.CurrentX = (nXLow + .5) * GraticuleSize
'  PIC.Print Format$(Y, "0,000")
'Next nY


End Sub

```

```
Sub DrawScaleBar (PIC As PictureBox, iWhere As Integer)
" Draw a bar scale in picture box

'iWhere can be 0 (no scale)
'          2 lower right
'          3 lower left
'          4 upper left
'          5 upper right

If iWhere = 0 Then Exit Sub
  Dim i1 As Long, i2 As Long
  Dim XWidth As Single, YHeight As Single
  XWidth = PIC.ScaleWidth: YHeight = -PIC.ScaleHeight

  Dim BarSize As Single, StepSize As Single
  BarSize = .25 * XWidth

Dim LogSmall As Single, iLogSmall As Single, nStep As Long, nSteps As Long
  LogSmall = Log(BarSize) / Log(10#)
  iLogSmall = Int(LogSmall)
  If (LogSmall - iLogSmall) < .31 Then iLogSmall = iLogSmall - 1
  StepSize = 10 ^ iLogSmall

   nSteps = BarSize / StepSize - .5
   BarSize = nSteps * StepSize

'Draw the scale bar as a series of rectangles, or maybe just a line
'This should be located a sixteenth up/down from the bottom/top
'  and centered horizontally in whichever quadrant.

'For now, just draw a dimension line

Dim XMin As Single, XMax As Single, YMin As Single, YMax As Single
Dim XCenter As Single, YCenter As Single


'XCenter = PIC.ScaleLeft + .75 * XWidth
' XMin = XCenter - .5 * BarSize
' XMax = XCenter + .5 * BarSize
   XMax = PIC.ScaleLeft + .975 * XWidth
   i1 = XMax / StepSize - .5
   XMax = i1 * StepSize
   XMin = XMax - BarSize


YCenter = PIC.ScaleTop - .975 * YHeight
'This will make the scale bar one sixteenth the height of the picture
YMax = YCenter + .01 * YHeight
YMin = YCenter - .01 * YHeight

   'Always draw a white background box, and a black dimensioning line
   PIC.Line (XMin, YMin)-(XMax, YMax), RGB(255, 255, 255), BF

   PIC.Line (XMin, YMin)-(XMin, YMax), RGB(0, 0, 0)
```

```
158|   PIC.Line (XMax, YMin)-(XMax, YMax), RGB(0, 0, 0)
159|   PIC.Line (XMin, YCenter)-(XMax, YCenter), RGB(0, 0, 0)
160|
161|   For nStep = 0 To nSteps - 1
162|     If nStep Mod 2 Then
163|       PIC.Line (XMin + nStep * StepSize, YMin)-(XMin + (nStep + 1) * StepSize,
    YCenter), RGB(0, 0, 0), BF
164|       Else
165|       PIC.Line (XMin + nStep * StepSize, YMax)-(XMin + (nStep + 1) * StepSize,
    YCenter), RGB(0, 0, 0), BF
166|     End If
167|   Next nStep
168|   PIC.Line (XMin, YMin)-(XMax, YMax), RGB(0, 0, 0), B
169|
170|   PIC.FontName = "Arial": PIC.FontBold = False: PIC.FontItalic = False: PIC.FontSize =
    8.5
171|   PIC.ForeColor = RGB(0, 0, 0)
172|   iLogSmall = SetTextAlign(PIC.hDC, TA_LEFT Or TA_BOTTOM)
173|   PIC.CurrentX = XMin: PIC.CurrentY = YMax
174|
175| Dim strUnits As String
176|   strUnits = CoordinateUnits()
177|   'PIC.FontTransparent = False
178|   PIC.Print Format$(BarSize, "#,###.#") & " " & strUnits
179|   'PIC.FontTransparent = True
180|
181| 'I was thinking about drawing extra scales here
182|   Select Case strUnits
183|     Case "meters"
184|     Case "feet"
185|     Case Else
186|       'For other units, don't even try adding text
187|   End Select
188| End Sub
189|
190| Function FindGraticuleSize (rctSize As RealRect) As Single
191|
192| Dim dX As Single, dY As Single, Smaller As Single
193|   dX = rctSize.Right - rctSize.Left
194|   dY = rctSize.Top - rctSize.Bottom
195|   If dY < 0 Then dY = -dY
196|
197|   If dX < dY Then
198|     Smaller = dX
199|    Else
200|     Smaller = dY
201|   End If
202|
203| Dim LogSmall As Single, iLogSmall As Single
204|   LogSmall = Log(Smaller) / Log(10#)
205|   iLogSmall = Int(LogSmall)
206|   If (LogSmall - iLogSmall) < .31 Then iLogSmall = iLogSmall - 1
207|   FindGraticuleSize = 10 ^ iLogSmall
208| End Function
```

```
209|
210| Sub PrintGraticule (TargetRect As RealRect, Color As Long, Thick As Long)
211| " This function does two things.  It finds out how
212| " big grid cells should be and stores it in GraticuleSize, which can be used by the caller
     to display to user
213| " and it draws a map graticule in the color and thickness specified on the picturebox
214|   On Error GoTo ERR_PrintGraticule
215|
216|   Dim GraticuleSize As Single
217| GraticuleSize = FindGraticuleSize(TargetRect)
218|
219|
220| Dim Low As Single, High As Single
221| Dim nYLow As Long, nYHigh As Long
222| Dim nXLow As Long, nXHigh As Long
223|
224| Printer.ForeColor = Color: Printer.DrawWidth = Thick
225|
226| 'First, draw the graticule in X
227|
228| Dim X As Single, Y As Single
229| nXLow = Int(TargetRect.Left / GraticuleSize)
230|   If nXLow * GraticuleSize < TargetRect.Left Then nXLow = nXLow + 1
231| nXHigh = Int(TargetRect.Right / GraticuleSize) + 1
232|   If nXHigh * GraticuleSize > TargetRect.Right Then nXHigh = nXHigh - 1
233|
234| nYLow = Int(TargetRect.Bottom / GraticuleSize)
235|   If nYLow * GraticuleSize < TargetRect.Bottom Then nYLow = nYLow + 1
236| nYHigh = (Int(TargetRect.Top / GraticuleSize) + 1)
237|   If nYHigh * GraticuleSize > TargetRect.Top Then nYHigh = nYHigh - 1
238|
239|
240| Dim nX As Long, nY As Long
241|
242| For nX = nXLow To nXHigh
243|   X = nX * GraticuleSize
244|   Printer.Line (X, TargetRect.Bottom)-(X, TargetRect.Top)
245|     'Should optionally display coordinates.  More important for printing and
246|     'copying than for live maps, where this info comes from mousemove
247| Next nX
248|
249|
250| For nY = nYLow To nYHigh
251|   Y = nY * GraticuleSize
252|   Printer.Line (TargetRect.Left, Y)-(TargetRect.Right, Y)
253|     'Should optionally display coordinates.  More important for printing and
254|     'copying than for live maps, where this info comes from mousemove
255| Next nY
256|
257| nX = SetTextAlign(Printer.hDC, TA_BOTTOM + TA_CENTER)
258|
259| Printer.FontName = "Arial"
260| Printer.FontSize = 8
261| Printer.FontBold = False: Printer.FontItalic = False
```

```
262|
263|
264|
265|
266| 'Print coordinate labels along top
267| nX = SetTextAlign(Printer.hDC, TA_LEFT Or TA_BOTTOM)
268| For nX = nXLow To nXHigh
269|   X = nX * GraticuleSize
270|   Printer.CurrentX = X
271|   Printer.CurrentY = TargetRect.Top   '(nYHigh - .5) * GraticuleSize
272|   Printer.Print Format$(X, "0,000")
273| Next nX
274|
275| 'Print coordinate labels along right
276| nX = SetTextAlign(Printer.hDC, TA_LEFT Or TA_TOP)
277| For nY = nYLow To nYHigh
278|   Y = nY * GraticuleSize
279|   Printer.CurrentY = Y
280|   Printer.CurrentX = TargetRect.Right   '(nXLow + .5) * GraticuleSize
281|   Printer.Print Format$(Y, "0,000")
282| Next nY
283|
284|
285|
286| Exit Sub
287| ERR_PrintGraticule:
288|   Select Case Err
289|     Case Else
290|       MsgBox "Error #" & Err & CRLF & Error
291|       Err = 0
292|       Exit Sub
293|       Resume
294|   End Select
295|
296| End Sub
297|
298| Sub PrintScaleBar (TargetRect As RealRect, iWhere As Integer)
299| " Draw a bar scale to the printer
300|
301| 'iWhere can be 0 (no scale)
302| '          2 lower right
303| '          3 lower left
304| '          4 upper left
305| '          5 upper right
306|
307| If iWhere = 0 Then Exit Sub
308|   Dim i1 As Long, i2 As Long
309|   Dim XWidth As Single, YHeight As Single
310|   XWidth = TargetRect.Right - TargetRect.Left
311|   YHeight = TargetRect.Top - TargetRect.Bottom
312|
313|
314|   Dim BarSize As Single, StepSize As Single
315|   BarSize = .25 * XWidth
```

```
316|
317| Dim LogSmall As Single, iLogSmall As Single, nStep As Long, nSteps As Long
318|   LogSmall = Log(BarSize) / Log(10#)
319|   iLogSmall = Int(LogSmall)
320|   If (LogSmall - iLogSmall) < .31 Then iLogSmall = iLogSmall - 1
321|   StepSize = 10 ^ iLogSmall
322|
323|    nSteps = BarSize / StepSize - .5
324|    BarSize = nSteps * StepSize
325|
326| 'Draw the scale bar as a series of rectangles, or maybe just a line
327| 'This should be located a sixteenth up/down from the bottom/top
328| '  and centered horizontally in whichever quadrant.
329|
330| 'For now, just draw a dimension line
331|
332| Dim XMin As Single, XMax As Single, YMin As Single, YMax As Single
333| Dim XCenter As Single, YCenter As Single
334|
335|
336| 'XCenter = Printer.ScaleLeft + .75 * XWidth
337| ' XMin = XCenter - .5 * BarSize
338| ' XMax = XCenter + .5 * BarSize
339|    XMax = TargetRect.Left + .975 * XWidth
340|    i1 = XMax / StepSize - .5
341|    XMax = i1 * StepSize
342|    XMin = XMax - BarSize
343|
344|
345| YCenter = TargetRect.Top - .975 * YHeight
346|  'This will make the scale bar one sixteenth the height of the Printerture
347|  YMax = YCenter + .01 * YHeight
348|  YMin = YCenter - .01 * YHeight
349|
350|    'Always draw a white background box, and a black dimensioning line
351|    Printer.Line (XMin, YMin)-(XMax, YMax), RGB(255, 255, 255), BF
352|
353|    Printer.Line (XMin, YMin)-(XMin, YMax), RGB(0, 0, 0)
354|    Printer.Line (XMax, YMin)-(XMax, YMax), RGB(0, 0, 0)
355|    Printer.Line (XMin, YCenter)-(XMax, YCenter), RGB(0, 0, 0)
356|
357|    For nStep = 0 To nSteps - 1
358|       If nStep Mod 2 Then
359|          Printer.Line (XMin + nStep * StepSize, YMin)-(XMin + (nStep + 1) * StepSize,
      YCenter), RGB(0, 0, 0), BF
360|       Else
361|          Printer.Line (XMin + nStep * StepSize, YMax)-(XMin + (nStep + 1) * StepSize,
      YCenter), RGB(0, 0, 0), BF
362|       End If
363|    Next nStep
364|    Printer.Line (XMin, YMin)-(XMax, YMax), RGB(0, 0, 0), B
365|
366|    Printer.FontName = "Arial": Printer.FontBold = False: Printer.FontItalic = False:
      Printer.FontSize = 8.5
```

```
367|    Printer.ForeColor = RGB(0, 0, 0)
368|    iLogSmall = SetTextAlign(Printer.hDC, TA_LEFT Or TA_BOTTOM)
369|    Printer.CurrentX = XMin: Printer.CurrentY = YMax
370|
371|  Dim strUnits As String
372|    strUnits = CoordinateUnits()
373|
374|    Printer.Print Format$(BarSize, "#,###.#") & " " & strUnits
375|
376|
377|  'I was thinking about drawing extra scales here
378|    Select Case strUnits
379|      Case "meters"
380|      Case "feet"
381|      Case Else
382|        'For other units, don't even try adding text
383|    End Select
384|
385|  End Sub
386|
387|  Function ScaleFudgeFactor () As Single
388|  " Retrieves the scale fudge factor (accounts for screen size)
389|  If sngScaleFudgeFactor = 0 Then
390|    'This should execute only the first time the program runs
391|    'Get it from the INI file
392|    Dim Buffer As String, nBytesBack As Long
393|     Buffer = Space$(1024)
394|     nBytesBack = GetPrivateProfileString(ByVal "MapScale", ByVal "ScaleFudgeFactor",
    ByVal "0.0", Buffer, 1024, ByVal INIFile)
395|      sngScaleFudgeFactor = Val(Buffer)
396|      If sngScaleFudgeFactor = 0 Then
397|        'This should only execute the first time the user runs the program.
398|        nBytesBack = WritePrivateProfileString(ByVal "MapScale", ByVal
    "ScaleFudgeFactor", ByVal "1.0", ByVal INIFile)
399|        sngScaleFudgeFactor = 1#
400|      End If
401|  End If
402|  ScaleFudgeFactor = sngScaleFudgeFactor
403|
404|  End Function
405|
406|  Sub WMFGraticule (GraticuleColor As Long, GraticuleThickness As Long)
407|    Dim rctGDI As APIRECT, rctWorld As RealRect, DummyInt As Integer, hDCMeta  As
    Integer
408|    ReDim Pts(1 To 2) As RealPoint, GDIPts(1 To 2) As APIPoint
409|
410|    hDCMeta = GetHDCMeta()
411|    Call GetMetafileGDIBounds(rctGDI)
412|    Call GetMetaFileWorldBounds(rctWorld)
413|    Dim GraticuleSize As Single
414|
415|    GraticuleSize = FindGraticuleSize(rctWorld)
416|
417|    'SetPenInWMF (LineColor As Long, LineWidth As Integer, LineStyle As Integer)
```

```
418|    Call SetPenInWMF(GraticuleColor, CInt(GraticuleThickness), 0)
419|
420|
421|    Dim Low As Single, High As Single
422|    Dim nYLow As Long, nYHigh As Long
423|    Dim nXLow As Long, nXHigh As Long
424|  'First, draw the graticule in X
425|
426|  Dim X As Single, Y As Single, nX As Long, nY As Long
427|
428|    nXLow = Int(rctWorld.Left / GraticuleSize)
429|      If nXLow * GraticuleSize < rctWorld.Left Then nXLow = nXLow + 1
430|    nXHigh = Int(rctWorld.Right / GraticuleSize) + 1
431|      If nXHigh * GraticuleSize > rctWorld.Right Then nXHigh = nXHigh - 1
432|
433|    nYLow = Int(rctWorld.Bottom / GraticuleSize)
434|      If nYLow * GraticuleSize < rctWorld.Bottom Then nYLow = nYLow + 1
435|    nYHigh = (Int(rctWorld.Top / GraticuleSize) + 1)
436|      If nYHigh * GraticuleSize > rctWorld.Top Then nYHigh = nYHigh - 1
437|
438|  ReDim Pts(1 To 2) As RealPoint, GDIPts(1 To 2) As APIPoint
439|
440|  For nX = nXLow To nXHigh
441|    Pts(1).X = nX * GraticuleSize: Pts(2).X = Pts(1).X
442|    Pts(1).Y = rctWorld.Bottom: Pts(2).Y = rctWorld.Top
443|    Call ScaleR4ToI2LOCAL(rctWorld, rctGDI, Two, Pts(1), GDIPts(1))
444|    DummyInt = Polyline(hDCMeta, GDIPts(1), 2)
445|  Next nX
446|
447|
448|  For nY = nYLow To nYHigh
449|    Pts(1).Y = nY * GraticuleSize: Pts(2).Y = Pts(1).Y
450|    Pts(1).X = rctWorld.Left:   Pts(2).X = rctWorld.Right
451|    Call ScaleR4ToI2LOCAL(rctWorld, rctGDI, Two, Pts(1), GDIPts(1))
452|    DummyInt = Polyline(hDCMeta, GDIPts(1), 2)
453|  Next nY
454|
455|
456|  'Now, add labels
457|   Dim strTemp As String
458|    Dim ZSize As Single, ZFontCode As Integer
459|    ZSize = 10: ZFontCode = FontIDFromName("Arial")
460|    Call SetFontInWMF(ZSize, ZFontCode)
461|    Call SetTextColorInWMF(GraticuleColor)
462|
463|
464|  'Print coordinate labels along top
465|  nX = SetTextAlign(hDCMeta, TA_LEFT Or TA_BOTTOM)
466|  For nX = nXLow To nXHigh
467|    GDIPts(1).X = FindX_WMF(nX * GraticuleSize)
468|    GDIPts(1).Y = FindY_WMF(rctWorld.Top)    '(nYHigh - .5) * GraticuleSize
469|    strTemp = Format$(nX * GraticuleSize, "#,###.#")
470|    DummyInt = TextOut(hDCMeta, GDIPts(1).X, GDIPts(1).Y, ByVal strTemp,
     Len(strTemp))
```

```
471|    'dDebug.Print nX, DummyInt, strTemp
472|  Next nX
473|
474|  'Print coordinate labels along right
475|  nX = SetTextAlign(hDCMeta, TA_LEFT Or TA_TOP)
476|  For nY = nYLow To nYHigh
477|    GDIPts(1).Y = FindY_WMF(nY * GraticuleSize)
478|    GDIPts(1).X = FindX_WMF(rctWorld.Right)
479|    strTemp = Format$(nY * GraticuleSize, "#,###.#")
480|    DummyInt = TextOut(hDCMeta, GDIPts(1).X, GDIPts(1).Y, ByVal strTemp,
       Len(strTemp))
481|  Next nY
482|
483|
484|
485|
486|
487|  End Sub
488|
489|  Sub WMFScaleBar (iWhere As Integer)
490|  " Draw a bar scale in a metafile
491|
492|  'iWhere can be 0 (no scale)
493|  '          2 lower right
494|  '          3 lower left
495|  '          4 upper left
496|  '          5 upper right
497|
498|  If iWhere = 0 Then Exit Sub
499|   'This stuff is needed by metafiles
500|    Dim rctGDI As APIRECT, rctWorld As RealRect, hDCMeta   As Integer
501|    ReDim Pts(1 To 2) As RealPoint, GDIPts(1 To 2) As APIPoint
502|
503|    hDCMeta = GetHDCMeta()
504|    Call GetMetafileGDIBounds(rctGDI)
505|    Call GetMetaFileWorldBounds(rctWorld)
506|
507|
508|
509|    Dim i1 As Long, i2 As Long
510|    Dim XWidth As Single, YHeight As Single
511|    XWidth = rctWorld.Right - rctWorld.Left
512|    YHeight = rctWorld.Top - rctWorld.Bottom
513|
514|
515|    Dim BarSize As Single, StepSize As Single
516|    BarSize = .25 * XWidth
517|
518|  Dim LogSmall As Single, iLogSmall As Single, nStep As Long, nSteps As Long
519|    LogSmall = Log(BarSize) / Log(10#)
520|    iLogSmall = Int(LogSmall)
521|    If (LogSmall - iLogSmall) < .31 Then iLogSmall = iLogSmall - 1
522|    StepSize = 10 ^ iLogSmall
523|
```

```
524|    nSteps = BarSize / StepSize - .5
525|    BarSize = nSteps * StepSize
526|
527| 'Draw the scale bar as a series of rectangles, or maybe just a line
528| 'This should be located a sixteenth up/down from the bottom/top
529| '  and centered horizontally in whichever quadrant.
530|
531| 'For now, just draw a dimension line
532|
533| Dim XMin As Single, XMax As Single, YMin As Single, YMax As Single
534| Dim XCenter As Single, YCenter As Single
535|
536|
537|    XMax = rctWorld.Left + .975 * XWidth
538|    i1 = XMax / StepSize - .5
539|    XMax = i1 * StepSize
540|    XMin = XMax - BarSize
541|
542|
543| YCenter = rctWorld.Top - .975 * YHeight
544|  'This will make the scale bar one sixteenth the height of the Printerture
545|  YMax = YCenter + .01 * YHeight
546|  YMin = YCenter - .01 * YHeight
547|
548|
549|
550|
551|
552|
553|
554|
555|
556|    'Always draw a white background box, and a black dimensioning line
557|    'pPrinter.Line (XMin, YMin)-(XMax, YMax), RGB(255, 255, 255), BF
558|      Pts(1).X = XMin: Pts(1).Y = YMin
559|      Pts(2).X = XMax: Pts(2).Y = YMax
560|
561|      Call ScaleR4To12LOCAL(rctWorld, rctGDI, Two, Pts(1), GDIPts(1))
562|      Call SetPenInWMF(RGB(0, 0, 0), 1, 0)
563|      Call SetBrushInWMF(0, RGB(255, 255, 255))
564|      DummyInt = Rectangle(hDCMeta, GDIPts(1).X, GDIPts(1).Y, GDIPts(2).X,
    GDIPts(2).Y)
565|
566|
567|    Call SetBrushInWMF(0, RGB(0, 0, 0)) 'A solid black brush
568|    For nStep = 0 To nSteps - 1
569|      If nStep Mod 2 Then
570|        Pts(1).X = XMin + nStep * StepSize: Pts(1).Y = YMin
571|        Pts(2).X = XMin + (nStep + 1) * StepSize: Pts(2).Y = YCenter
572|        Call ScaleR4To12LOCAL(rctWorld, rctGDI, Two, Pts(1), GDIPts(1))
573|        DummyInt = Rectangle(hDCMeta, GDIPts(1).X, GDIPts(1).Y, GDIPts(2).X,
    GDIPts(2).Y)
574|       Else
575|        Pts(1).X = XMin + nStep * StepSize: Pts(1).Y = YMax
```

```
576|        Pts(2).X = XMin + (nStep + 1) * StepSize: Pts(2).Y = YCenter
577|        Call ScaleR4ToI2LOCAL(rctWorld, rctGDI, Two, Pts(1), GDIPts(1))
578|        DummyInt = Rectangle(hDCMeta, GDIPts(1).X, GDIPts(1).Y, GDIPts(2).X,
     GDIPts(2).Y)
579|      End If
580|    Next nStep
581|
582|    'pPrinter.Line (XMin, YMin)-(XMax, YMax), RGB(0, 0, 0), B
583|    Dim ZSize As Single, ZFontCode As Integer
584|    ZSize = 8.5: ZFontCode = FontIDFromName("Arial")
585|    Call SetFontInWMF(ZSize, ZFontCode)
586|    Call SetTextColorInWMF(RGB(0, 0, 0))
587|    iLogSmall = SetTextAlign(hDCMeta, TA_LEFT Or TA_BOTTOM)
588|
589|     GDIPts(1).X = FindX_WMF(XMin)
590|     GDIPts(1).Y = FindY_WMF(YMax)
591|
592|  Dim strUnits As String
593|    strUnits = Format$(BarSize, "#,###.#") & " " & CoordinateUnits()
594|    DummyInt = TextOut(hDCMeta, GDIPts(1).X, GDIPts(1).Y, ByVal strUnits,
     Len(strUnits))
595|
596|
597|
598|  'I was thinking about drawing extra scales here
599|    Select Case strUnits
600|      Case "meters"
601|      Case "feet"
602|      Case Else
603|        'For other units, don't even try adding text
604|    End Select
605|
606|  End Sub
607|
```

```
 1| Option Explicit
 2| 'Type RealPoint
 3| '   X As Single
 4| '   Y As Single
 5| 'End Type
 6| Type String8
 7|    f As String * 8
 8| End Type
 9| Dim RlPt As RealPoint
10| Dim S8 As String8
11|
12| ' PolyFill() Modes
13| Global Const ALTERNATE = 1
14| Global Const WINDING = 2
15|
16| 'Declare Sub Polygon Lib "GDI" (ByVal hDC As Integer, lpPoints As APIPoint, ByVal
    nCount As Integer)
17| 'Declare Sub HUGEPolygon Lib "GDI" Alias "Polygon" (ByVal hDC As Integer, ByVal
    ptrToAPIPoints As Long, ByVal nCount As Integer)
18| Declare Function Polygon Lib "GDI" (ByVal hDC As Integer, lpPoints As APIPOINT,
    ByVal nCount As Integer) As Integer
19| Declare Function HUGEPolygon Lib "GDI" Alias "Polygon" (ByVal hDC As Integer, ByVal
    ptrToAPIPoints As Long, ByVal nCount As Integer) As Integer
20|
21| Declare Sub SetPolyFillMode Lib "GDI" (ByVal hDC As Integer, ByVal nPolyFillMode As
    Integer)
22| 'Declare Sub Polyline Lib "GDI" (ByVal hDC As Integer, lpPoints As APIPoint, ByVal
    nCount As Integer)
23| 'Declare Sub HUGEPolyline Lib "GDI" Alias "PolyLine" (ByVal hDC As Integer, ByVal
    ptrToAPIPoints As Long, ByVal nCount As Integer)
24| Declare Function Polyline Lib "GDI" (ByVal hDC As Integer, lpPoints As APIPOINT,
    ByVal nCount As Integer) As Integer
25| Declare Function HUGEPolyline Lib "GDI" Alias "PolyLine" (ByVal hDC As Integer, ByVal
    ptrToAPIPoints As Long, ByVal nCount As Integer) As Integer
26|
27| Declare Function Rectangle Lib "GDI" (ByVal hDC As Integer, ByVal X1 As Integer,
    ByVal Y1 As Integer, ByVal X2 As Integer, ByVal Y2 As Integer) As Integer
28|
29| 'File Open/Save Dialog Flags
30| 'Global Const OFN_READONLY = &H1&
31| 'Global Const OFN_OVERWRITEPROMPT = &H2&
32| 'Global Const OFN_HIDEREADONLY = &H4&
33| 'Global Const OFN_NOCHANGEDIR = &H8&
34| 'Global Const OFN_SHOWHELP = &H10&
35| 'Global Const OFN_NOVALIDATE = &H100&
36| 'Global Const OFN_ALLOWMULTISELECT = &H200&
37| 'Global Const OFN_EXTENSIONDIFFERENT = &H400&
38| 'Global Const OFN_PATHMUSTEXIST = &H800&
39| 'Global Const OFN_FILEMUSTEXIST = &H1000&
40| 'Global Const OFN_CREATEPROMPT = &H2000&
41| 'Global Const OFN_SHAREAWARE = &H4000&
42| 'Global Const OFN_NOREADONLYRETURN = &H8000&
43| '
44|
```

```
Dim ReturnVal As Long

Sub BLOB2RealPoints (BLOBin As String, DataPts() As RealPoint)
Dim n As Long, nPoints As Long, NPos As Long
nPoints = Len(BLOBin) / 8 '8 Bytes to a RealPoint
Dim NStart As Long
'We will append datapoints to the end of the current array.
' This way, we can have an unlimited number (kind of) of points
' in a line.  Before starting to fill this array using
' this subroutine, you should redim it to contain zero points
If LBound(DataPts) <> UBound(DataPts) Then
   NStart = UBound(DataPts) + 1
   ReDim Preserve DataPts(1 To UBound(DataPts) + nPoints)
 Else
   NStart = 1
   ReDim Preserve DataPts(1 To nPoints)
End If

'ReDim Preserve DataPts(1 To UBound(DataPts) + NPoints)
NPos = 1  'NPos will keep track of where in the BLOB we are
For n = NStart To UBound(DataPts)
   S8.f = Mid$(BLOBin, NPos, 8)
   LSet RlPt = S8
   DataPts(n) = RlPt
   NPos = NPos + 8
Next n
End Sub

Sub CountGridClipNRowsNCols (ClipText As String, nRows As Long, nCols As Long)
   nCols = 1
   nRows = 1
   Dim TabChar As String * 1, CR  As String * 1
   CR = Chr$(13)
   TabChar = Chr$(9)
   Dim n As Long
' Find number of rows (delimeted by asc(13))
   For n = 1 To Len(ClipText)
      If Mid$(ClipText, n, 1) = CR Then
         nRows = nRows + 1
         nCols = nCols + 1
      End If
      If Mid$(ClipText, n, 1) = TabChar Then
         nCols = nCols + 1
      End If
   Next n
   nCols = nCols \ nRows
End Sub

Sub DrawBLOBLine (XYBLOB As String, Target As PictureBox)
Dim n As Long, nPoints As Long, NPos As Long
n = Len(XYBLOB)
If n < 16 Then Exit Sub' Skip out if the BLOB doesn't contain at least one line segment.
If n Mod 8 <> 0 Then Exit Sub
nPoints = n / 8 '8 Bytes to a RealPoint
```

```
 99| Dim DataPoint As RealPoint, S8 As String8
100|
101| NPos = 1  'NPos will keep track of where in the BLOB we are
102|
103| S8.f = Mid$(XYBLOB, NPos, 8)
104| LSet DataPoint = S8
105|
106| Target.CurrentX = DataPoint.X
107| Target.CurrentY = DataPoint.Y
108| NPos = 9
109|
110| For n = 2 To nPoints
111|    S8.f = Mid$(XYBLOB, NPos, 8)
112|    LSet DataPoint = S8
113|
114|    Target.Line -(DataPoint.X, DataPoint.Y)
115|    NPos = NPos + 8
116| Next n
117|
118| End Sub
119|
120| Sub DrawR4Polygon (PolyIn() As RealPoint, InColor As Long, PIC As PictureBox)
121| Dim OriginalScale As RealRect
122| Dim nPoints As Long, XMult As Single, YMult As Single, XAdd As Single, YAdd As Single
123|
124| Dim lbPoly As Long, ubPoly As Long
125| lbPoly = LBound(PolyIn)
126| ubPoly = UBound(PolyIn)
127| nPoints = ubPoly - lbPoly + 1
128| If nPoints < 3 Then
129|    Beep
130|    Exit Sub
131|   End If
132|
133| ReDim GDIPoly(lbPoly To ubPoly) As APIPOINT
134| SaveScaling PIC, OriginalScale
135|
136| GetScalingWorldToGDI PIC, XAdd, YAdd, XMult, YMult
137| Dim n As Long
138| For n = lbPoly To ubPoly
139|    GDIPoly(n).X = (PolyIn(n).X + XAdd) * XMult
140|    GDIPoly(n).Y = (PolyIn(n).Y + YAdd) * YMult
141| Next n
142|
143|
144| PIC.FillColor = InColor 'This one controls the interior of the polygon.
145|
146| SetPolyFillMode PIC.hDC, ALTERNATE
147| ReturnVal = Polygon(PIC.hDC, GDIPoly(lbPoly), nPoints)
148|
149| RestoreScaling PIC, OriginalScale
150|
151| End Sub
152|
```

```
153| Sub FillGridFromRlPtArr (GridIn As Grid, RlPt() As RealPoint)
154| ' This function fills a Grid with data from an array of RealPoints (XY Coordinates in
     Real*4 format as type RealPoint)
155| ' It assumes the user has turned off any automatic repositioning
156| ' before calling.
157|
158|   Dim n As Long, nPts As Long, m As Long, lngOffset As Long
159|   n = GridIn.Rows
160|   GridIn.Rows = 1
161|   GridIn.Rows = n
162|   GridIn.FixedRows = 1'The last four lines should have cleared the grid
163|   nPts = UBound(RlPt) - LBound(RlPt) + 1
164|   lngOffset = LBound(RlPt) - 1
165|   If GridIn.Rows < nPts + 1 Then GridIn.Rows = nPts + 2
166|   For n = 1 To nPts
167|     m = lngOffset + n
168|     GridIn.Row = n
169|     GridIn.Col = 1
170|     GridIn = Format(RlPt(m).X)
171|     GridIn.Col = 2
172|     GridIn = Format(RlPt(m).Y)
173|   Next n
174| End Sub
175|
176| Sub GetPixelBounds (PIC As PictureBox, PixelRect As APIRECT)
177|   Dim ScalingB4 As RealRect
178|   SaveScaling PIC, ScalingB4
179|   PIC.ScaleMode = 3
180|   PixelRect.Left = PIC.ScaleLeft
181|   PixelRect.Top = PIC.ScaleTop
182|   PixelRect.Bottom = PIC.ScaleTop + PIC.ScaleHeight
183|   PixelRect.Right = PIC.ScaleLeft + PIC.ScaleWidth
184|   RestoreScaling PIC, ScalingB4
185| End Sub
186|
187| Sub GetScalingWorldToGDI (PIC As PictureBox, XAdd As Single, YAdd As Single, XMult
     As Single, YMult As Single)
188|   XAdd = -XAtLeft(PIC)
189|   YAdd = -YAtTop(PIC)
190|   XMult = 1# / XPerPixel(PIC)
191|   YMult = 1# / YPerPixel(PIC)
192| End Sub
193|
194| Function MakeXYBlobFromClipText (GridIn As String) As String
195| Dim nRows As Long, nCols As Long, n As Long
196| Dim NextStart  As Long, NextStop As Long
197|
198| 'For n = nNodesAlready To 1 Step -1
199| '   Unload picNode(n)
200| 'Next n   'Specific to the line/polygon editor
201|
202|
203|
204| NextStart = 1
```

```
205|
206| Dim MaxX As Single, MaxY As Single, MinX As Single, MinY As Single
207| MaxX = -1E+35
208| MaxY = -1E+35
209| MinX = 1E+35
210| MinY = 1E+35
211|
212| Dim Temp As String
213| Call CountGridClipNRowsNCols(GridIn, nRows, nCols)
214| ReDim DataPt(1 To nRows) As RealPoint
215| Temp = GridIn & CR
216| If nRows > 1 Then
217| For n = 1 To nRows
218|    NextStop = InStr(NextStart, Temp, TabChar)
219|    DataPt(n).X = Val(Mid$(Temp, NextStart, NextStop - NextStart))
220|    MinX = MinR4(MinX, DataPt(n).X)
221|    MaxX = MaxR4(MaxX, DataPt(n).X)
222|    NextStart = NextStop + 1
223|    If (nCols > 2) Then
224|       NextStop = InStr(NextStart, Temp, TabChar)
225|     Else
226|       NextStop = InStr(NextStart, Temp, CR)
227|    End If
228|    DataPt(n).Y = Val(Mid$(Temp, NextStart, NextStop - NextStart))
229|    MinY = MinR4(MinY, DataPt(n).Y)
230|    MaxY = MaxR4(MaxY, DataPt(n).Y)
231|    NextStart = InStr(NextStart, Temp, CR) + 1
232| Next n
233|
234|
235| ' The following code displays a line, along with
236| '  pictures at the lines turning points.
237| '  Useful for editing, but should probably be moved to
238| '   a line-editing form and separated from the conversion stuff.
239| ' xxxxx GWPouch 940303
240| 'Set range on box to draw, and draw the line just edited.
241| '   Dim RangeX As Single, RangeY As Single
242| '   RangeX = MaxX - MinX
243| '   RangeY = MaxY - MinY
244| '   'Assume the size of the picture box is fixed, and work from there.
245| '   Picture1.ScaleLeft = MinX - .1 * RangeX
246| '   Picture1.ScaleTop = MaxY + .1 * RangeY
247| '
248| '   If RangeX >= RangeY Then
249| '     'It's a wide line, so set the scale based on width
250| '      Picture1.ScaleWidth = 1.2 * RangeX
251| '      Picture1.ScaleHeight = -1.2 * RangeX 'Set the scaleheight to the same as the
    scalewidth, so it's isotropic, given a square picture box
252| '     Else
253| '      Picture1.ScaleHeight = -1.2 * RangeY
254| '      Picture1.ScaleWidth = 1.2 * RangeY
255| '   End If
256| '   Picture1.BackColor = RGB(255, 255, 0)
257| '   Picture1.DrawStyle = 1
```

```
258|  '  Picture1.DrawWidth = 1
259|  '  Picture1.Line (MinX, MaxY)-(MaxX, MinY), RGB(255, 0, 0), B
260|  '  Picture1.DrawStyle = 0
261|  '  Picture1.ForeColor = RGB(0, 0, 255)
262|  '  'Picture1.DrawWidth = 2
263|  '  Load picNode(1)
264|  '  picNode(1).Left = DataPt(1).X
265|  '  picNode(1).Top = DataPt(1).Y
266|  '  picNode(1).Visible = True
267|  '  For n = 2 To nRows
268|  '     Picture1.Line (DataPt(n - 1).X, DataPt(n - 1).Y)-(DataPt(n).X, DataPt(n).Y)
269|  '        Load picNode(n)
270|  '        picNode(n).Left = DataPt(n).X
271|  '        picNode(n).Top = DataPt(n).Y
272|  '        picNode(n).Visible = True
273|  '  Next n
274|  '  nNodesAlready = nRows
275|  '
276|  MakeXYBlobFromClipText = RealPoints2BLOB(DataPt())
277|  End If
278|  End Function
279|
280|  Function MaxR4 (A1 As Single, A2 As Single) As Single
281|  'These should be moved to a general purpose utility module sometime.
282|
283|   If A1 > A2 Then
284|     MaxR4 = A1
285|    Else
286|     MaxR4 = A2
287|   End If
288|
289|  End Function
290|
291|  Function MinR4 (A1 As Single, A2 As Single) As Single
292|   If A1 < A2 Then
293|     MinR4 = A1
294|    Else
295|     MinR4 = A2
296|   End If
297|  End Function
298|
299|  Function RealPoints2BLOB (DataPts() As RealPoint) As String
300|  Dim n As Long
301|  Dim BigTemp  As String
302|  BigTemp = ""
303|  For n = LBound(DataPts) To UBound(DataPts)
304|     RlPt = DataPts(n)
305|     LSet S8 = RlPt
306|     BigTemp = BigTemp & S8.f
307|  Next n
308|  RealPoints2BLOB = BigTemp
309|  End Function
310|
311|  Sub RestoreScaling (PIC As PictureBox, RestoreFrom As RealRect)
```

```
312| PIC.ScaleMode = 0
313|    PIC.ScaleLeft = RestoreFrom.Left
314|   PIC.ScaleTop = RestoreFrom.Top
315|   PIC.ScaleWidth = RestoreFrom.Right
316|   PIC.ScaleHeight = RestoreFrom.Bottom
317| End Sub
318|
319| Sub SaveScaling (PIC As PictureBox, SaveTo As RealRect)
320| 'GWPouch 940613
321|  SaveTo.Left = PIC.ScaleLeft
322|  SaveTo.Top = PIC.ScaleTop
323|  SaveTo.Right = PIC.ScaleWidth
324|  SaveTo.Bottom = PIC.ScaleHeight
325| End Sub
326|
327| Function XAtLeft (PIC As PictureBox) As Single
328|
329|  XAtLeft = PIC.ScaleLeft
330|
331|   'Dim OriginalScale As RealRect
332|  ' SaveScaling PIC, OriginalScale
333|  ' Dim XBefore As Single, XAfter As Single
334|  ' XBefore = PIC.CurrentX
335|  ' PIC.ScaleMode = 3
336|  ' PIC.CurrentX = 0
337|  ' RestoreScaling PIC, OriginalScale
338|  ' XAtLeft = PIC.CurrentX
339|  ' PIC.CurrentX = XBefore
340| End Function
341|
342| Function XPerPixel (PIC As PictureBox) As Single
343| 'GWPouch 940613
344| 'NOte that this modifes currentX, but you should never trust
345| ' that property anyway.
346| Dim OriginalScale As RealRect
347| SaveScaling PIC, OriginalScale
348| Dim WidthBefore As Single, WidthAfter As Single
349| WidthBefore = PIC.ScaleWidth
350| PIC.ScaleMode = 3
351| WidthAfter = PIC.ScaleWidth
352| RestoreScaling PIC, OriginalScale
353| XPerPixel = WidthBefore / WidthAfter
354|
355|
356|
357| 'XBefore = PIC.CurrentX
358| 'PIC.ScaleMode = 3
359| 'PIC.CurrentX = PIC.CurrentX + 25
360| 'RestoreScaling PIC, OriginalScale
361| 'XAfter = PIC.CurrentX
362| 'If XAfter <> XBefore Then
363| '   XPerPixel = (XAfter - XBefore) / 25#
364| ' Else ' Probably at right edge, so try moving left
365| '   PIC.ScaleMode = 3
```

```
366| '   PIC.CurrentX = PIC.CurrentX - 25
367| '   XAfter = PIC.CurrentX
368| '   XPerPixel = (XBefore - XAfter) / 25#
369| 'End If
370| ' RestoreScaling PIC, OriginalScale
371| End Function
372|
373| Function YAtTop (PIC As PictureBox) As Single
374| YAtTop = PIC.ScaleTop
375| '   Dim OriginalScale As RealRect
376| '   SaveScaling PIC, OriginalScale
377| '   Dim YBefore As Single, YAfter As Single
378| '   YBefore = PIC.CurrentY
379| '   PIC.ScaleMode = 3
380| '   PIC.CurrentY = 0
381| '   RestoreScaling PIC, OriginalScale
382| '   YAtTop = PIC.CurrentY
383| '   PIC.CurrentY = YBefore
384| End Function
385|
386| Function YPerPixel (PIC As PictureBox) As Single
387| 'GWPouch 940613
388| 'Modified to not move the CurrentX,CurrentY 940616
389| 'I hope this improves accuracy too.
390|
391| Dim OriginalScale As RealRect
392| SaveScaling PIC, OriginalScale
393|
394| Dim HeightBefore As Single, HeightAfter As Single
395| HeightBefore = PIC.ScaleHeight
396| PIC.ScaleMode = 3
397| HeightAfter = PIC.ScaleHeight
398| RestoreScaling PIC, OriginalScale
399| YPerPixel = HeightBefore / HeightAfter
400|
401|
402|
403|
404| 'Dim YBefore As Single, YAfter As Single
405| 'YBefore = PIC.CurrentY
406| 'PIC.ScaleMode = 3
407| 'PIC.CurrentY = PIC.CurrentY + 25
408| 'RestoreScaling PIC, OriginalScale
409| 'YAfter = PIC.CurrentY
410| 'If YAfter <> YBefore Then
411| '   YPerPixel = (YAfter - YBefore) / 25#
412| ' Else  ' Probably at right edge, so try moving left
413| '   PIC.ScaleMode = 3
414| '   PIC.CurrentY = PIC.CurrentY - 25
415| '   YAfter = PIC.CurrentY
416| '   YPerPixel = (YBefore - YAfter) / 25#
417| 'End If
418| ' RestoreScaling PIC, OriginalScale
419| End Function
```

```
420|
```

```
 1| Option Explicit
 2| Dim MinX As Single, MinY As Single, MaxX As Single, MaxY As Single
 3|
 4| Global Const LDGEOM_APPEND = 0
 5| Global Const LDGEOM_OVERWRITE = 256
 6| Global Const LDGEOM_CLOSEPOLYGON = 2
 7| Global Const LDGEOM_MAKEHATCHLINES = 4
 8|
 9| Global Const BOLDTYPE = -32768
10| Global Const ITALICTYPE = 16384
11| Global Const MASK14LOBITS = 16383
12|
13| Sub AddDataChangeBounds (x1new As Single, y1new As Single, x2new As Single,
    y2new As Single)
14| 'This subroutine changes DataBounds in response to the addition
15| ' of new data.  Whatever subroutine reads the new data calls
16| ' this subroutine with the DataBounds of the new data.  It then
17| ' finds the minimum rectangle (DataBounds) which encloses both
18| '
19| 'It also resets the DataAspect ratio, which is the ratio
20| ' of the range in x to the range in y
21| 'Finally, it re-adjusts the MaxScaleFactor to fit all the data
22| '  onto the Map when it is zoomed out all the way.
23|
24| If nThemes Then ' execute this bit only if there are already themes
25|                 ' displayed.
26|
27|
28|   If x1new < DataBounds.Left Then
29|     DataBounds.Left = x1new
30|   End If
31|   If x2new < DataBounds.Left Then
32|     DataBounds.Left = x2new
33|   End If
34|
35|   If x1new > DataBounds.Right Then
36|     DataBounds.Right = x1new
37|   End If
38|   If x2new > DataBounds.Right Then
39|     DataBounds.Right = x2new
40|   End If
41|
42|
43|   If y1new < DataBounds.Bottom Then
44|     DataBounds.Bottom = y1new
45|   End If
46|   If y2new < DataBounds.Bottom Then
47|     DataBounds.Bottom = y2new
48|   End If
49|
50|   If y1new > DataBounds.Top Then
51|     DataBounds.Top = y1new
52|   End If
53|   If y2new > DataBounds.Top Then
```

```
    DataBounds.Top = y2new
  End If

Else 'if this is the first theme
  If x1new > x2new Then
    DataBounds.Right = x1new
    DataBounds.Left = x2new
  Else
    DataBounds.Right = x2new
    DataBounds.Left = x1new
  End If

  If y1new > y2new Then
    DataBounds.Top = y1new
    DataBounds.Bottom = y2new
  Else
    DataBounds.Top = y2new
    DataBounds.Bottom = y1new
  End If
End If ' execute the following for any number of themes



'nThemes = nThemes + 1
DataRange.X = (DataBounds.Right - DataBounds.Left)
DataRange.Y = (DataBounds.Top - DataBounds.Bottom)

DataAspect = DataRange.X / DataRange.Y

End Sub

Sub LoadHatchLines (snpLines As Snapshot, HatchLinesOut() As HatchLinePlain, ByVal
iflgReplace As Integer, BoundBox As RealRect)
'  snpLines is a snapshot containing the selected hatch lines.
'  It has fields consisting
'  of following fields in the specified order
'   nStart as long
'   nPoints as long
'   ZWidth As Integer 'the size value of Z for the user-selected variable [db field]
'   ZColor As Long  'the color
'   ZStyle As Integer  'the symbol number
'   Name As String * 16  'a name the user can recognize


' Name, the name of the HatchLines (typically, something like Clinton Lake or IGUCA #7
'   something that means something to the user and  can be used to find data for
DBForm
' HatchLinesOut is an array that will receive these Lines.
' If iflgReplace is set to true, this subroutine will overwrite all existing Lines.
' If set to false, it will add these Lines to the end of the list.
On Error GoTo LoadHatchLinesErr
Dim nStart As Long, nRecs As Long, n  As Long, nJunk As Long
Dim Temp As String
snpLines.MoveFirst
```

```
snpLines.MoveLast
nRecs = snpLines.RecordCount
'GWPouch940824  If AND2(iflgReplace, LDGEOM_OVERWRITE) Then
'GWPouch940824'   'Probably never called this way.
'GWPouch940824'   ReDim HatchLinesOut(1 To nRecs)
'GWPouch940824   nStart = 1 'Note that nStart is an offset.  This lets us avoid a lot of n-1 non-sense.
'GWPouch940824  Else
 'Usually called with iflgReplace=0
   nStart = UBound(HatchLinesOut) + 1
'GWPouch940824   If UBound(HatchLinesOut) = LBound(HatchLinesOut) Then 'We have an unitialized array, so treat it like the overwrite case.
   If UBound(HatchLinesOut) = 0 Then 'We have an unitialized array, so treat it like the overwrite case.
      nStart = 1
      ReDim HatchLinesOut(1 To nRecs)
     Else
      ReDim Preserve HatchLinesOut(1 To nStart + nRecs - 1)
   End If
'GWPouch940824 End If

MinX = 1E+34
MinY = 1E+34
MaxX = -1E+34
MaxY = -1E+34

snpLines.MoveFirst
Dim RlPt As RealPoint, S8 As string8
Dim LineBounds As RealRect
frmMap.pnl3dPercentDone.FloodPercent = 0
frmMap.lblPercentDone = "Loading hatch lines"

For n = nStart To UBound(HatchLinesOut)' For all retrieved records____

nJunk = snpLines("HatchXYBLOB").FieldSize()
If nJunk < 16 Then 'There are no coordinates for this line, so skip to the
                ' the bottom of this loop.
        GoTo BottomLoadHatchLinesLoop
End If
   Temp = snpLines("HatchXYBLOB")
   Dim NPos  As Long, m As Long, One  As Long
   One = 1
   NPos = 1
   HatchLinesOut(n).nStart = hGMemLastPoint
   HatchLinesOut(n).nPoints = nJunk \ 8
   ReDim Pts(1 To HatchLinesOut(n).nPoints) As RealPoint 'This will be properly dimensioned by BLOB2RealPoints
   Call ZLoadBLOB2RealPoints(Temp, Pts())
   Call GetPolyBounds(Pts(1), HatchLinesOut(n).nPoints, LineBounds)
   HatchLinesOut(n).MinX = LineBounds.Left
   HatchLinesOut(n).MaxX = LineBounds.Right
   HatchLinesOut(n).MinY = LineBounds.Bottom
   HatchLinesOut(n).MaxY = LineBounds.Top

```

```
156|    Call SetRlPntsToGlobalMem(hGMemAllPoints, Pts(1), hGMemLastPoint,
       HatchLinesOut(n).nPoints)
157|    If (HatchLinesOut(n).nPoints Mod 2) = 1 Then
158|       HatchLinesOut(n).nPoints = HatchLinesOut(n).nPoints - 1
159|       'GoTo BottomLoadHatchLinesLoop
160|    End If
161|
162|    ' Note that the above subroutine call implicitly took care of setting MinX, MaxX etc.
163| BottomLoadHatchLinesLoop:
164|  HatchLinesOut(n).ZWidth = DSng(snpLines("HatchZWidth"))
165| 'Make sure the line width is a reasonable value
166|    If HatchLinesOut(n).ZWidth < 1 Then HatchLinesOut(n).ZWidth = 1
167|    If HatchLinesOut(n).ZWidth > 100 Then HatchLinesOut(n).ZWidth = 100
168|
169|  HatchLinesOut(n).ZColor = DLng(snpLines("HatchZColor"))
170| 'Color is safe
171|
172|  HatchLinesOut(n).ZStyle = DInt(snpLines("HatchZStyle"))
173|    If HatchLinesOut(n).ZStyle < 0 Then HatchLinesOut(n).ZStyle = 0
174|    If HatchLinesOut(n).ZStyle > 6 Then HatchLinesOut(n).ZStyle = 6
175|
176|  snpLines.MoveNext
177|  frmMap.pnl3dPercentDone.FloodPercent = (n - nStart) / nRecs * 100
178| Next n
179|  frmMap.pnl3dPercentDone.FloodPercent = 100
180|  frmMap.lblPercentDone = "Done loading hatch lines"
181|  Beep: frmMap.pnl3dPercentDone.FloodPercent = 0: frmMap.lblPercentDone = ""
182| 'Could close the snapshot here, but that would be kind of rude.
183| '  Maybe we should close it explicitly in the calling routine.
184| """""""snpLines.Close
185| BoundBox.Left = MinX
186| BoundBox.Right = MaxX
187| BoundBox.Top = MaxY
188| BoundBox.Bottom = MinY
189| Exit Sub
190| LoadHatchLinesErr:
191| Select Case Err
192|  Case 3018 'Could not find field
193|    If InStr(Error$, "zqFeatureID") Then Resume Next
194|    Err = 0: Exit Sub: Resume
195|  Case Else
196|    Err = 0
197|    Exit Sub
198|    Resume Next
199| End Select
200|
201| End Sub
202|
203| Sub LoadLabels (snpLabels As Snapshot, LabelsOut() As LabelDecl, ByVal iflgReplace
       As Integer, BoundBox As RealRect)
204| '  snpLabels is a snapshot containing the selected Labels.
205| '  It has fields consisting
206| '  of following fields in the specified order
207| '   nStart As Long      'Location of the point in memory
```

```
208| '   FontSize As Single     'the size for the user-selected variable [db field]
209| '   ZColor As Long       'the color
210| '   Text As String * 96  'the symbol number
211| '   FontCode As Integer
212| '   Name As String * 16  'a name the user can recognize
213| '
214| 'Modified from LoadLines, 940708 GWPouch
215| '
216| ' Name, the name of the Label (typically, something like 10S32W34 or Pittsburgh, or
217| '   something that means something to the user and  can be used to find data for
    DBForm
218| ' LabelsOut is an array that will receive these Labels.
219| ' If iflgReplace is set to true, this subroutine will overwrite all existing Labels.
220| ' If set to false, it will add these Labels to the end of the list.
221| On Error GoTo LoadLabelsErr:
222| Dim nStart As Long, nRecs As Long, n  As Long, Pt As RealPoint
223| snpLabels.MoveFirst
224| snpLabels.MoveLast
225| nRecs = snpLabels.RecordCount
226| frmMap.lblPercentDone = "Loading labels from " & Themes(nThemes).ThemeName
227| frmMap.pnl3dPercentDone.FloodPercent = 0
228| 'GWPouch940824 If iflgReplace Then
229| 'GWPouch940824     ReDim LabelsOut(1 To nRecs)
230| 'GWPouch940824     nStart = 1 'Note that nStart is an offset.  This lets us avoid a lot of
    n-1 non-sense.
231| 'GWPouch940824   Else
232|     nStart = UBound(LabelsOut) + 1
233|     If UBound(LabelsOut) = 0 Then 'We have an unitialized array, so treat it like the
    overwrite case.
234|     'GWPouch940824  changed from Ubound=lbound to ubound=0.  Could fiddle and
    make this able to display 65000 rather than 32767, but then you get into negative
    subscripts.
235|         nStart = 1
236|         ReDim LabelsOut(1 To nRecs)
237|        Else
238|         ReDim Preserve LabelsOut(1 To nStart + nRecs - 1)
239|     End If
240| 'GWPouch940824 End If
241| MinX = 1E+34
242| MinY = 1E+34
243| MaxX = -1E+34
244| MaxY = -1E+34
245|
246| snpLabels.MoveFirst
247| Dim X As Single, Y As Single
248| For n = nStart To UBound(LabelsOut)' For all retrieved records____
249|
250|  'LabelsOut(n).X = DSng(snpLabels("X"))
251|
252|   X = DSng(snpLabels("zqX"))
253|    MinX = MinR4(X, MinX)
254|    MaxX = MaxR4(X, MaxX)
255|    Pt.X = X 'For some reason, this works but setting Pt.X =DSng doesn't.
256|
```

```
257|  'LabelsOut(n).Y = DSng(snpLabels("Y"))
258|  Y = DSng(snpLabels("zqY"))
259|  'Find the Min and Max of Y
260|     MinY = MinR4(Y, MinY)
261|     MaxY = MaxR4(Y, MaxY)
262|    Pt.Y = Y
263|  """""""""""dDebug.Print n, X, Y, MinX, MaxX, MinY, MaxY
264|  'hGMemLastPoint = hGMemLastPoint + 1
265|  LabelsOut(n).nStart = hGMemLastPoint
266|
267|  Call SetRlPntsToGlobalMem(hGMemAllPoints, Pt, hGMemLastPoint, lngOne)
268|  LabelsOut(n).FontSize = DSng(snpLabels("zqFontSize"))
269|  'Make sure the point-size for the symbol is a reasonable value
270|     If LabelsOut(n).FontSize < 4 Then LabelsOut(n).FontSize = 4
271|     If LabelsOut(n).FontSize > 100 Then LabelsOut(n).FontSize = 100
272|
273|  LabelsOut(n).ZColor = DLng(snpLabels("zqColor"))
274|  'Color is safe
275|  LabelsOut(n).FontCode = DInt(snpLabels("zqFontCode")) 'New 940701
276|  LabelsOut(n).Alignment = DInt(snpLabels("zqAlignment"))'New 940708
277|  LabelsOut(n).Opaque = DInt(snpLabels("zqOpaque"))'New 940708
278|
279|
280|  LabelsOut(n).Text = snpLabels("zqLabelText")
281|  ' Make sure the symbol integer is in a reasonable (printable) range
282|  LabelsOut(n).FeatureID = DLng(snpLabels("zqFeatureID"))
283|
284|
285|
286|  snpLabels.MoveNext
287|  frmMap.pnl3dPercentDone.FloodPercent = (n - nStart) / nRecs * 100
288|  Next n
289|  frmMap.pnl3dPercentDone.FloodPercent = 100
290|  frmMap.lblPercentDone = "Done loading Labels"
291|  Beep: frmMap.pnl3dPercentDone.FloodPercent = 0: frmMap.lblPercentDone = ""
292|  'Could close the snapshot here, but that would be kind of rude.
293|  '  Maybe we should close it explicitly in the calling routine.
294|  """""""""snpLabels.Close
295|  BoundBox.Left = MinX
296|  BoundBox.Right = MaxX
297|  BoundBox.Top = MaxY
298|  BoundBox.Bottom = MinY
299|  Exit Sub
300|  LoadLabelsErr:
301|  'This jumping out for a no returned fields query
302|  ' causes problems down the road.
303|  'Maybe forcing err=0 will work.
304|  Select Case Err
305|   Case 3018 'Could not find field
306|     If InStr(Error$, "zqFeatureID") Then Resume Next
307|     Err = 0: Exit Sub: Resume
308|   Case 94 'Invalid use of null
309|      LabelsOut(n).Text = ""
310|      Resume Next
```

```
311|  Case Else
312|    MsgBox "Error #" & Err & " occurred loading TextLabels." & CRLF & Error
313|    Err = 0
314|    Exit Sub
315|    Resume Next
316|  End Select
317|
318|
319|  End Sub
320|
321|  Sub LoadLines (snpLines As Snapshot, LinesOut() As LineDeclPlain, ByVal iflgReplace
     As Integer, BoundBox As RealRect)
322|  '  snpLines is a snapshot containing the selected lines.
323|  '  It has fields consisting
324|  '  of following fields in the specified order
325|  '   nStart as long
326|  '   nPoints as long
327|  '   ZWidth As Integer 'the size value of Z for the user-selected variable [db field]
328|  '   ZColor As Long  'the color
329|  '   ZStyle As Integer  'the symbol number
330|  '   Name As String * 16  'a name the user can recognize
331|
332|
333|  ' Name, the name of the Lines (typically, something like US-41 or Wabash River or
334|  '    something that means something to the user and  can be used to find data for
     DBForm
335|  '  LinesOut is an array that will receive these Lines.
336|  ' If iflgReplace is set to true, this subroutine will overwrite all existing Lines.
337|  ' If set to false, it will add these Lines to the end of the list.
338|  On Error GoTo LoadLinesErr
339|  Dim nStart As Long, nRecs As Long, n  As Long, nJunk As Long
340|  Dim Temp As String
341|  ' XXX need to include error handling for the case of a null return query (like MinX >
     MaxX) generating no returns.
342|  Dim LineBounds As RealRect
343|  snpLines.MoveFirst
344|  snpLines.MoveLast
345|  nRecs = snpLines.RecordCount
346|  'If AND2(iflgReplace, LDGEOM_OVERWRITE) Then
347|  '    ReDim LinesOut(1 To nRecs)
348|  '    nStart = 1 'Note that nStart is an offset.  This lets us avoid a lot of n-1 non-sense.
349|  '  Else
350|     nStart = UBound(LinesOut) + 1
351|     If UBound(LinesOut) = 0 Then 'We have an unitialized array, so treat it like the
     overwrite case.
352|  'GWPouch940824  changed from Ubound=lbound to ubound=0.  Could fiddle and make
     this able to display 65000 rather than 32767, but then you get into negative subscripts.
353|        nStart = 1
354|        ReDim LinesOut(1 To nRecs)
355|       Else
356|        ReDim Preserve LinesOut(1 To nStart + nRecs - 1)
357|     End If
358|  'End If
359|
```

```
360| MinX = 1E+34
361| MinY = 1E+34
362| MaxX = -1E+34
363| MaxY = -1E+34
364|
365| snpLines.MoveFirst
366| Dim RlPt As RealPoint, S8 As string8
367| frmMap.lblPercentDone = "Loading lines from " & Themes(nThemes).ThemeName
368| frmMap.pnl3dPercentDone.FloodPercent = 0
369| DoEvents
370|
371|
372| For n = nStart To UBound(LinesOut)' For all retrieved records____
373|
374| nJunk = snpLines("zqXYBLOBData").FieldSize()
375| If nJunk < 8 Then 'There are no coordinates for this line, so skip to the
376|                  ' the bottom of this loop.
377|          GoTo BottomLoadLinesLoop
378| End If
379|    Temp = snpLines("zqXYBLOBData")
380|    If AND2(iflgReplace, LDGEOM_CLOSEPOLYGON) Then Temp =
    ForceBLOBToBePolygon(Temp)
381|  nJunk = Len(Temp)
382|    Dim NPos  As Long, m As Long, One  As Long
383|    One = 1
384|    NPos = 1
385|    LinesOut(n).nStart = hGMemLastPoint
386|    LinesOut(n).nPoints = nJunk \ 8
387|    ReDim Pts(1 To LinesOut(n).nPoints) As RealPoint 'This will be properly dimensioned
    by BLOB2RealPoints
388|    Call ZLoadBLOB2RealPoints(Temp, Pts()) 'This implicitly updates MinX,MaxX,MinY,
    MaxY
389|
390|    Call SetRlPntsToGlobalMem(hGMemAllPoints, Pts(1), hGMemLastPoint,
    LinesOut(n).nPoints)
391|    Call GetPolyBounds(Pts(1), LinesOut(n).nPoints, LineBounds)
392|    LinesOut(n).MinX = LineBounds.Left
393|    LinesOut(n).MaxX = LineBounds.Right
394|    LinesOut(n).MinY = LineBounds.Bottom
395|    LinesOut(n).MaxY = LineBounds.Top
396|
397|  LinesOut(n).ZWidth = DSng(snpLines("ZqWidth"))
398| 'Make sure the line width is a reasonable value
399|    If LinesOut(n).ZWidth < 1 Then LinesOut(n).ZWidth = 1
400|    If LinesOut(n).ZWidth > 100 Then LinesOut(n).ZWidth = 100
401|
402|  LinesOut(n).ZColor = DLng(snpLines("ZqColor"))
403| 'Color is safe
404|
405|  LinesOut(n).ZStyle = DInt(snpLines("ZqStyle"))
406|    If LinesOut(n).ZStyle < 0 Then LinesOut(n).ZStyle = 0
407|    If LinesOut(n).ZStyle > 6 Then LinesOut(n).ZStyle = 6
408|
409|
```

```
410  LinesOut(n).FeatureID = DLng(snpLines("zqFeatureID"))
411  BottomLoadLinesLoop:
412   snpLines.MoveNext
413   frmMap.pnl3dPercentDone.FloodPercent = (n - nStart) / nRecs * 100
414  Next n
415   frmMap.pnl3dPercentDone.FloodPercent = 100
416   frmMap.lblPercentDone = "Done loading lines"
417  Beep: frmMap.pnl3dPercentDone.FloodPercent = 0: frmMap.lblPercentDone = ""
418  'Could close the snapshot here, but that would be kind of rude.
419  '  Maybe we should close it explicitly in the calling routine.
420  """""""snpLines.Close
421  BoundBox.Left = MinX
422  BoundBox.Right = MaxX
423  BoundBox.Top = MaxY
424  BoundBox.Bottom = MinY
425  Exit Sub
426  LoadLinesErr:
427  Select Case Err
428   Case 3018 'Could not find field
429    If InStr(Error$, "zqFeatureID") Then Resume Next
430    Err = 0: Exit Sub: Resume
431   Case Else
432    Err = 0
433    Exit Sub
434    Resume Next
435  End Select
436  End Sub
437
438  Sub LoadPoints (snpPoints As Snapshot, PointsOut() As PointDeclPlain, ByVal
     iflgReplace As Integer, BoundBox As RealRect)
439  '  snpPoints is a snapshot containing the selected points.
440  '  It has fields consisting
441  '  of following fields in the specified order
442  ' X      Map coordinates to plot
443  ' Y      Map coordintates
444  ' ZSize      A variable (base table field, or calculated) whose
445  '        value will produce a size
446  ' ZColor Variable for Color
447  ' ZSymbol Variable for Symbol
448  '
449  ' Name, the name of the Points (typically, something like 10S32W34 or Pittsburgh, or
450  '    something that means something to the user and  can be used to find data for
     DBForm
451  '  PointsOut is an array that will receive these points.
452  ' If iflgReplace is set to true, this subroutine will overwrite all existing points.
453  ' If set to false, it will add these points to the end of the list.
454  On Error GoTo LoadPointsErr:
455  Dim nStart As Long, nRecs As Long, n  As Long, Pt As RealPoint
456  snpPoints.MoveFirst
457  snpPoints.MoveLast
458  nRecs = snpPoints.RecordCount
459  frmMap.lblPercentDone = "Loading Points from " & Themes(nThemes).ThemeName
460  frmMap.pnl3dPercentDone.FloodPercent = 0
461  'If iflgReplace Then
```

```
462|  '    ReDim PointsOut(1 To nRecs)
463|  '    nStart = 1 'Note that nStart is an offset.  This lets us avoid a lot of n-1 non-sense.
464|  ' Else
465|     nStart = UBound(PointsOut) + 1
466|     If UBound(PointsOut) = 0 Then 'We have an unitialized array, so treat it like the
       overwrite case.
467|  'GWPouch940824  changed from Ubound=lbound to ubound=0.  Could fiddle and make
       this able to display 65000 rather than 32767, but then you get into negative subscripts.
468|        nStart = 1
469|        ReDim PointsOut(1 To nRecs)
470|       Else
471|        ReDim Preserve PointsOut(1 To nStart + nRecs - 1)
472|     End If
473|  'End If
474|  MinX = 1E+34
475|  MinY = 1E+34
476|  MaxX = -1E+34
477|  MaxY = -1E+34
478|
479|  snpPoints.MoveFirst
480|  Dim X As Single, Y As Single, L As Long
481|  For n = nStart To UBound(PointsOut)' For all retrieved records____
482|
483|   'PointsOut(n).X = DSng(snpPoints("X"))
484|
485|    X = DSng(snpPoints("zqX"))
486|     MinX = MinR4(X, MinX)
487|     MaxX = MaxR4(X, MaxX)
488|     Pt.X = X 'For some reason, this works but setting Pt.X =DSng doesn't.
489|
490|   'PointsOut(n).Y = DSng(snpPoints("Y"))
491|   Y = DSng(snpPoints("zqY"))
492|  'Find the Min and Max of Y
493|     MinY = MinR4(Y, MinY)
494|     MaxY = MaxR4(Y, MaxY)
495|     Pt.Y = Y
496|
497|   'hGMemLastPoint = hGMemLastPoint + 1
498|   PointsOut(n).nStart = hGMemLastPoint
499|   L = PointsOut(n).nStart
500|   Call SetRlPntsToGlobalMem(hGMemAllPoints, Pt, hGMemLastPoint, lngOne)
501|   PointsOut(n).ZSize = DSng(snpPoints("ZqSize"))
502|  'Make sure the point-size for the symbol is a reasonable value
503|     If PointsOut(n).ZSize < 1 Then PointsOut(n).ZSize = 1
504|     If PointsOut(n).ZSize > 100 Then PointsOut(n).ZSize = 100
505|
506|   PointsOut(n).ZColor = DLng(snpPoints("ZqColor"))
507|  'Color is safe
508|
509|   PointsOut(n).ZSymbol = Printable(DInt(snpPoints("ZqSymbol")))
510|  ' Make sure the symbol integer is in a reasonable (printable) range
511|   PointsOut(n).FeatureID = DLng(snpPoints("zqFeatureID"))
512|
513|   PointsOut(n).ZFontCode = DInt(snpPoints("ZqFontCode")) 'New 940701
```

```
514|
515|  snpPoints.MoveNext
516|  frmMap.pnl3dPercentDone.FloodPercent = (n - nStart) / nRecs * 100
517| Next n
518| frmMap.pnl3dPercentDone.FloodPercent = 100
519| frmMap.lblPercentDone = "Done loading points"
520| Beep: frmMap.pnl3dPercentDone.FloodPercent = 0: frmMap.lblPercentDone = ""
521| 'Could close the snapshot here, but that would be kind of rude.
522|  '  Maybe we should close it explicitly in the calling routine.
523| """""""snpPoints.Close
524| BoundBox.Left = MinX
525| BoundBox.Right = MaxX
526| BoundBox.Top = MaxY
527| BoundBox.Bottom = MinY
528| Exit Sub
529| LoadPointsErr:
530| Select Case Err
531|  Case 94: Resume Next 'Skip nulls
532|  Case 3018 'Could not find field
533|    If InStr(Error$, "zqFeatureID") Then Resume Next
534|    Err = 0: Exit Sub: Resume
535|  Case Else
536|    Err = 0
537|    Exit Sub
538|    Resume Next
539| End Select
540|
541|
542|
543| End Sub
544|
545| Sub LoadPolygons (snpPoly As Snapshot, PolysOut() As typPoly, ByVal iflgReplace As
    Integer, BoundBox As RealRect)
546|  '  snpPoly is a snapshot containing the selected polygons
547|  '  It has fields consisting
548|  '  of following fields in the specified order
549|  '    nStart As Long
550|  '    nPoints As Long
551|  '    FillColor As Long     'The color of the interior
552|  '    OutlineColor As Long  'The color of the outline
553|  '    OutlineWidth As Integer
554|  '    FillStyle As Integer  'To indicate solid, vertically cross-hatched, etc.
555|  '    MinX As Single
556|  '    MaxX As Single
557|  '    MinY As Single
558|  '    MaxY As Single
559|  '    Name As String * 28  'a name the user can recognize
560|
561|
562| ' Name, the name of the Lines (typically, something like US-41 or Wabash River or
563|  '    something that means something to the user and  can be used to find data for
    DBForm
564|  ' PolysOut is an array that will receive these Lines.
565| ' If iflgReplace is set to true, this subroutine will overwrite all existing Lines.
```

```
566| ' If set to false, it will add these Lines to the end of the list.
567| On Error GoTo LoadPolygonsErr
568|
569| Dim nStart As Long, nRecs As Long, n  As Long, nJunk As Long
570| Dim Temp As String
571| ' XXX need to include error handling for the case of a null return query (like MinX >
     MaxX) generating no returns.
572| Dim PolyBounds As RealRect
573| snpPoly.MoveFirst
574| snpPoly.MoveLast
575| nRecs = snpPoly.RecordCount
576|
577| MinX = 1E+34
578| MinY = 1E+34
579| MaxX = -1E+34
580| MaxY = -1E+34
581|
582|
583|
584| 'If nRecs = 0 Then GoTo EXITLoadPolygons
585|
586|    nStart = UBound(PolysOut) + 1
587|    If UBound(PolysOut) = 0 Then 'We have an unitialized array, so treat it like the
     overwrite case.
588| 'GWPouch940824  changed from Ubound=lbound to ubound=0.  Could fiddle and make
     this able to display 65000 rather than 32767, but then you get into negative subscripts.
589|       nStart = 1
590|       ReDim PolysOut(1 To nRecs)
591|      Else
592|       ReDim Preserve PolysOut(1 To nStart + nRecs - 1)
593|    End If
594|
595|
596| snpPoly.MoveFirst
597| Dim RlPt As RealPoint, S8 As string8
598| frmMap.lblPercentDone = "Loading Polygons from " & Themes(nThemes).ThemeName
599| frmMap.pnl3dPercentDone.FloodPercent = 0
600|
601| DoEvents
602| For n = nStart To UBound(PolysOut)' For all retrieved records____
603|
604| nJunk = snpPoly("zqPolygonOutline").FieldSize()
605| If nJunk < 12 Then 'There are no coordinates for this polygon, so skip to the
606|                ' the bottom of this loop.
607|          GoTo BottomLoadPolygonLoop
608| End If
609|
610|    Temp = snpPoly("zqPolygonOutline")
611|    'Always force polygons to close.
612|    Temp = ForceBLOBToBePolygon(Temp)
613|   nJunk = Len(Temp)
614|    Dim NPos  As Long, m As Long, One  As Long
615|    One = 1
616|    NPos = 1
```

```
617|     PolysOut(n).nStart = hGMemLastPoint
618|     PolysOut(n).nPoints = nJunk \ 8
619|     ReDim Pts(1 To PolysOut(n).nPoints) As RealPoint 'This will be properly dimensioned
      by BLOB2RealPoints
620|     Call ZLoadBLOB2RealPoints(Temp, Pts()) 'This implicitly updates MinX,MaxX,MinY,
      MaxY
621|     Call SetRlPntsToGlobalMem(hGMemAllPoints, Pts(1), hGMemLastPoint,
      PolysOut(n).nPoints)
622|     Call GetPolyBounds(Pts(1), PolysOut(n).nPoints, PolyBounds)
623|     PolysOut(n).MinX = PolyBounds.Left
624|     PolysOut(n).MaxX = PolyBounds.Right
625|     PolysOut(n).MinY = PolyBounds.Bottom
626|     PolysOut(n).MaxY = PolyBounds.Top
627|
628|   PolysOut(n).FillColor = DLng(snpPoly("zqFillColor"))
629|   PolysOut(n).OutlineColor = DLng(snpPoly("zqOutlineColor"))
630|  'Color is safe
631|
632|   PolysOut(n).FillStyle = DInt(snpPoly("zqFillStyle"))
633|     If PolysOut(n).FillStyle < 0 Then PolysOut(n).FillStyle = 0
634|     If PolysOut(n).FillStyle > 7 Then PolysOut(n).FillStyle = 7
635|
636|   PolysOut(n).OutlineWidth = DInt(snpPoly("zqOutlineWidth"))
637|   'Make sure the line width is a reasonable value
638|     If PolysOut(n).OutlineWidth < 0 Then PolysOut(n).OutlineWidth = 0
639|     If PolysOut(n).OutlineWidth > 100 Then PolysOut(n).OutlineWidth = 100
640|
641|   PolysOut(n).FeatureID = DLng(snpPoly("zqFeatureID"))
642|  BottomLoadPolygonLoop:
643|   snpPoly.MoveNext
644|   frmMap.pnl3dPercentDone.FloodPercent = (n - nStart) / nRecs * 100
645|  Next n
646|   frmMap.pnl3dPercentDone.FloodPercent = 100
647|   frmMap.lblPercentDone = "Done loading Polygons"
648|   Beep: frmMap.pnl3dPercentDone.FloodPercent = 0: frmMap.lblPercentDone = ""
649|  EXITLoadPolygons:
650|  BoundBox.Left = MinX
651|  BoundBox.Right = MaxX
652|  BoundBox.Top = MaxY
653|  BoundBox.Bottom = MinY
654|  Exit Sub
655|  LoadPolygonsErr:
656|  Select Case Err
657|   Case 3018 'Could not find field
658|     If InStr(Error$, "zqFeatureID") Then Resume Next
659|     Err = 0: Exit Sub: Resume
660|   Case Else
661|     Err = 0
662|     Exit Sub
663|     Resume Next
664|  End Select
665|
666|  End Sub
667|
```

```
668| Sub LoadThemeDefinitionFromTable (DBIn As Database, NameOfThemeWanted As
   String)
669|
670| On Error GoTo ERRLoadThemeDefinitionFromTable
671| Dim snpGeomData As Snapshot
672| Dim ThemeToRead As typTheme, BoundBox As RealRect, iflgReplace As Integer
673|
674| Call ReadThemeDefinitionFromTableByTitle(DBIn, NameOfThemeWanted,
   ThemeToRead)
675|
676| If ThemeToRead.MapSource = "" Then Exit Sub'No theme was read.
677|
678|   nThemes = nThemes + 1
679|   ReDim Preserve Themes(1 To nThemes)
680|   Themes(nThemes).myIndex = nThemes
681|   Themes(nThemes).ThemeType = ThemeToRead.ThemeType
682|   Themes(nThemes).ThemeName = ThemeToRead.ThemeName
683|   If Themes(nThemes).ThemeType <> LABELTEXT Then
684|     frmSpatialQuery.cmbLayerList.AddItem Themes(nThemes).ThemeName
685|     frmSpatialQuery.cmbLayerList.ItemData(frmSpatialQuery.cmbLayerList.NewIndex) =
   nThemes
686|   End If
687|   Themes(nThemes).MapSource = ThemeToRead.MapSource
688|   Themes(nThemes).DBSource = ThemeToRead.DBSource
689|
690|
691|
692| ' DBIn need not be the same as MapDB.  This permits the eventual possibility
693| '  of having a central MDB used for MAPDB and each user having a DB full of their own
   theme definitions.
694| ' For now, it will be called with DBIn=MapDB, but this could be easily changed.
695| Set snpGeomData = MapDB.CreateSnapshot(ThemeToRead.MapSource)
696|
697| 'Sub LoadTiles (snpTile As Snapshot, TilesOut() As TileDecl2, ByVal iflgReplace As
   Integer, BoundBox As RealRect)
698| 'Call LoadTiles (snpGeomData, TilesOut() ,iflgReplace, BoundBox )
699| Select Case ThemeToRead.ThemeType
700|  Case JOINTEDLINE
701|      Themes(nThemes).FirstLoc = UBound(Lines) + 1
702|      Call LoadLines(snpGeomData, Lines(), iflgReplace, BoundBox)
703|      Themes(nThemes).LastLoc = UBound(Lines)
704|  Case ONEPOINT
705|     Themes(nThemes).FirstLoc = UBound(Points) + 1
706|     Call LoadPoints(snpGeomData, Points(), iflgReplace, BoundBox)
707|     Themes(nThemes).LastLoc = UBound(Points)
708|  Case TILEBLOCK
709|      Themes(nThemes).FirstLoc = UBound(Tiles) + 1
710|      Call LoadTiles(snpGeomData, Tiles(), iflgReplace, BoundBox)
711|      Themes(nThemes).LastLoc = UBound(Tiles)
712|  Case HATCHLINE
713|      Themes(nThemes).FirstLoc = UBound(HatchLines) + 1
714|      Call LoadHatchLines(snpGeomData, HatchLines(), iflgReplace, BoundBox)
715|      Themes(nThemes).LastLoc = UBound(HatchLines)
716|  Case SOLIDPOLYGON
```

```
717|        Themes(nThemes).FirstLoc = UBound(Polys) + 1
718|        Call LoadPolygons(snpGeomData, Polys(), iflgReplace, BoundBox)
719|        Themes(nThemes).LastLoc = UBound(Polys)
720|    Case LABELTEXT
721|        Themes(nThemes).FirstLoc = UBound(Labels) + 1
722|        Call LoadLabels(snpGeomData, Labels(), iflgReplace, BoundBox)
723|        Themes(nThemes).LastLoc = UBound(Labels)
724| End Select
725|
726| If Themes(nThemes).LastLoc < Themes(nThemes).FirstLoc Then GoSub
    CancelTheme
727| If BoundBox.Left > 1E+33 Or BoundBox.Bottom > 1E+33 Or BoundBox.Top < -1E+33 Or
    BoundBox.Right < -1E+33 Then GoSub CancelTheme
728| If nThemes = 1 Then
729|    LSet DataBounds = BoundBox
730|   Else
731|    Call AddDataChangeBounds(BoundBox.Left, BoundBox.Bottom, BoundBox.Right,
    BoundBox.Top)
732| End If
733|    'xxx need to adjust this so that it adjusts the bounds more often
734|
735|
736| snpGeomData.Close
737| Set snpGeomData = Nothing
738|
739|
740| Exit Sub
741|
742| ERRLoadThemeDefinitionFromTable:
743|  Select Case Err
744|   Case Else
745|    MsgBox "Error #" & Err & " occurred trying to load '" & NameOfThemeWanted & "'
    from disk." & CRLF & Error
746|    Err = 0
747|    GoSub CancelTheme
748|    Exit Sub
749|    Resume
750|  End Select
751| Exit Sub
752|
753| CancelTheme:
754|      Beep
755|      MsgBox "That resulted in no data being retrieved from " & NameOfThemeWanted,
    0, "Generic Theme Loading Error"
756|      nThemes = nThemes - 1
757|      If nThemes = 0 Then
758|        Erase Themes
759|       Else          'ReDim Preserve Themes(0 To nThemes)
760|        ReDim Preserve Themes(1 To nThemes)
761|      End If
762|      Exit Sub
763|      Return
764| End Sub
765|
```

```
Sub LoadTiles (snpTile As Snapshot, TilesOut() As TileDecl2, ByVal iflgReplace As Integer, BoundBox As RealRect)
'GWPouch 940720  Changed the definiton of tiles, so that they are basically
' like polygons (Outlining options as well as interior color, new interior fill style....
'  Also, tiles now have two coordinates defing the NW and SE corners, so that
' Windows GDI Rectangle function can be used to draw them.



'   snpTile is a snapshot containing the selected Tiles
'   It has fields consisting
'   of following fields in the specified order
'    nStart As Long
'    nPoints As Long
'    FillColor As Long     'The color of the interior
'    OutlineColor As Long  'The color of the outline
'    OutlineWidth As Integer
'    FillStyle As Integer  'To indicate solid, vertically cross-hatched, etc.
'    MinX As Single
'    MaxX As Single
'    MinY As Single
'    MaxY As Single
'    Name As String * 28  'a name the user can recognize


' Name, the name of the Lines (typically, something like US-41 or Wabash River or
'    something that means something to the user and  can be used to find data for DBForm
' TilesOut is an array that will receive these Lines.
' If iflgReplace is set to true, this subroutine will overwrite all existing Lines.
' If set to false, it will add these Lines to the end of the list.
On Error GoTo LoadTiles2Err

Dim nStart As Long, nRecs As Long, n  As Long, nJunk As Long

snpTile.MoveFirst
snpTile.MoveLast
nRecs = snpTile.RecordCount
'If AND2(iflgReplace, LDGEOM_OVERWRITE) Then
'    ReDim TilesOut(1 To nRecs)
'    nStart = 1 'Note that nStart is an offset.  This lets us avoid a lot of n-1 non-sense.
' Else
    nStart = UBound(TilesOut) + 1
    If UBound(TilesOut) = 0 Then 'We have an unitialized array, so treat it like the overwrite case.
'GWPouch940824  changed from Ubound=lbound to ubound=0.  Could fiddle and make this able to display 65000 rather than 32767, but then you get into negative subscripts.
        nStart = 1
        ReDim TilesOut(1 To nRecs)
       Else
        ReDim Preserve TilesOut(1 To nStart + nRecs - 1)
    End If
'End If

```

```
MinX = 1E+34
MinY = 1E+34
MaxX = -1E+34
MaxY = -1E+34

snpTile.MoveFirst
ReDim RlPt(1 To 2) As RealPoint

frmMap.pnl3dPercentDone.FloodPercent = 0
frmMap.lblPercentDone = "Loading Tiles from " & Themes(nThemes).ThemeName

For n = nStart To UBound(TilesOut)' For all retrieved records____

   Dim NPos  As Long, m As Long, One  As Long, Two As Long
   One = 1&: Two = 2&
   NPos = 1
   TilesOut(n).nStart = hGMemLastPoint
   RlPt(1).X = DSng(snpTile("zqMinX"))
   RlPt(1).Y = DSng(snpTile("zqMaxY"))
   RlPt(2).X = DSng(snpTile("zqMaxX"))
   RlPt(2).Y = DSng(snpTile("zqMinY"))
   MinX = MinR4(RlPt(1).X, MinR4(RlPt(2).X, MinX))
   MaxX = MaxR4(RlPt(1).X, MaxR4(RlPt(2).X, MaxX))
   MinY = MinR4(RlPt(1).Y, MinR4(RlPt(2).Y, MinY))
   MaxY = MaxR4(RlPt(1).Y, MaxR4(RlPt(2).Y, MaxY))


   Call SetRlPntsToGlobalMem(hGMemAllPoints, RlPt(1), hGMemLastPoint, Two)

  TilesOut(n).InteriorColor = DLng(snpTile("zqFillColor"))
  TilesOut(n).OutlineColor = DLng(snpTile("zqOutlineColor"))
'Color is safe

 TilesOut(n).InteriorFillStyle = DInt(snpTile("zqFillStyle"))
   If TilesOut(n).InteriorFillStyle < 0 Then TilesOut(n).InteriorFillStyle = 0
   If TilesOut(n).InteriorFillStyle > 7 Then TilesOut(n).InteriorFillStyle = 7

 TilesOut(n).OutlineWidth = DInt(snpTile("zqOutlineWidth"))
'Make sure the line width is a reasonable value
   If TilesOut(n).OutlineWidth < 0 Then TilesOut(n).OutlineWidth = 0
   If TilesOut(n).OutlineWidth > 10 Then TilesOut(n).OutlineWidth = 10
 TilesOut(n).OutlineStyle = DInt(snpTile("zqOutlineStyle"))
   If TilesOut(n).OutlineStyle < 0 Then TilesOut(n).OutlineStyle = 5  'Tranparent
   If TilesOut(n).OutlineStyle > 6 Then TilesOut(n).OutlineStyle = 5  'Tranparent


 TilesOut(n).FeatureID = DLng(snpTile("zqFeatureID"))
BottomLoadTiles2Loop:
 snpTile.MoveNext
 frmMap.pnl3dPercentDone.FloodPercent = (n - nStart) / nRecs * 100
Next n
 frmMap.pnl3dPercentDone.FloodPercent = 100
 frmMap.lblPercentDone = "Done loading Tiles"
 Beep: frmMap.pnl3dPercentDone.FloodPercent = 0: frmMap.lblPercentDone = ""
```

```
870  BoundBox.Left = MinX
871  BoundBox.Right = MaxX
872  BoundBox.Top = MaxY
873  BoundBox.Bottom = MinY
874
875
876  Exit Sub
877  LoadTiles2Err:
878  Select Case Err
879   Case 3018 'Could not find field
880     If InStr(Error$, "zqFeatureID") Then Resume Next
881     Err = 0: Exit Sub: Resume
882   Case Else
883     Err = 0
884     Exit Sub
885     Resume Next
886  End Select
887
888
889
890
891
892  End Sub
893
894  Function Printable (InValue As Integer) As Integer
895  Dim Temp As Integer
896  Temp = InValue
897     If InValue < 34 Then Temp = 33'Print a !
898     If InValue > 126 And InValue < 161 Then Temp = 161'An upside down exclamation
     point'
899     If InValue > 255 Then Temp = 255
900    Printable = Temp
901  End Function
902
903  Sub SetFontInPIC (FontCode As Integer, PIC As PictureBox)
904  'This subroutine uses the table tblFonts in the current database to
905  '  keep track of font names.
906  Static FontNameUsed As String, LastFontName As String, nF As Long, tmpCode As
     Integer
907  Static LastFontCode As Integer
908
909  On Error GoTo ERRSetFontInPIC
910  If LastFontCode = FontCode Then
911     Exit Sub   'If it's the same fontcode, leave it alone quickly.
912    Else
913     LastFontCode = FontCode
914  End If
915
916
917
918  'FONTCODE encodes Bold and Italic in the first two bits,
919  '   and a font ID # used in tblFonts to encode the font name
920  '   in the other 14 bits.
921  PIC.FontBold = AND2(FontCode, BOLDTYPE)
```

```
922|  PIC.FontItalic = AND2(FontCode, ITALICTYPE)
923|   'If AND2(FontCode, BOLDTYPE) Then PIC.FontBold = True
924|   'If AND2(FontCode, ITALICTYPE) Then PIC.FontItalic = True
925|  tmpCode = MASK14LOBITS And FontCode
926|
927|   FontNameUsed = "Arial" 'Set default font to Arial
928|   For nF = 1 To UBound(FontTable)
929|   'dDebug.Print nF, tmpCode, FontTable(nF).Number, FontTable(nF).Name
930|      If FontTable(nF).Number = tmpCode Then
931|        FontNameUsed = FontTable(nF).Name
932|        Exit For
933|      End If
934|   Next nF
935|
936|   'tblFonts.Seek "=", (MASK14LOBITS And FontCode)
937|   'If tblFonts.NoMatch Then
938|   '    PIC.FontName = "Arial"
939|   ' Else
940|   '    PIC.FontName = Trim$(tblFonts("Name"))
941|   'End If
942|
943|      PIC.FontName = FontNameUsed
944|  Exit Sub
945|
946|
947|  ERRSetFontInPIC:
948|  Select Case Err
949|    Case 91 'Object variable not set, which pretty much means
950|                ' this is the first time we've executed this
951|                ' subroutine.
952|     Set tblFonts = MapDB.OpenTable("tblFontList")
953|     tblFonts.Index = "PrimaryKey"
954|     Resume
955|    Case 380
956|     FontNameUsed = "Arial"
957|     Resume
958|    Case Else
959|     MsgBox "Error #" & Err & " occurred in SetFontInPIC." & CRLF & Error
960|     Err = 0
961|     Exit Sub
962|     Resume Next
963|  End Select
964|  End Sub
965|
966|  Sub SetFontInPrinter (FontCode As Integer)
967|  'This subroutine uses the table tblFonts in the current database to
968|  ' keep track of font names.
969|  Static FontNameUsed As String, LastFontName As String, nF As Long, tmpCode As
     Integer
970|  Static LastFontCode As Integer
971|  On Error GoTo ERRSetFontInPrinter
972|  If LastFontCode = FontCode Then
973|     Exit Sub   'If it's the same fontcode, leave it alone quickly.
974|   Else
```

```
 975|     LastFontCode = FontCode
 976|  End If
 977|
 978|  'FONTCODE encodes Bold and Italic in the first two bits,
 979|  '   and a font ID # used in tblFonts to encode the font name
 980|  '   in the other 14 bits.
 981|   If AND2(FontCode, BOLDTYPE) Then Printer.FontBold = True
 982|   If AND2(FontCode, ITALICTYPE) Then Printer.FontItalic = True
 983|   tmpCode = MASK14LOBITS And FontCode
 984|
 985|   FontNameUsed = "Arial" 'Set default font to Arial
 986|   For nF = 1 To UBound(FontTable)
 987|      If FontTable(nF).Number = tmpCode Then
 988|        FontNameUsed = FontTable(nF).Name
 989|        Exit For
 990|      End If
 991|   Next nF
 992|   Printer.FontName = FontNameUsed
 993|  '  tblFonts.Seek "=", (MASK14LOBITS And FontCode)
 994|  '  If tblFonts.NoMatch Then
 995|  '     Printer.FontName = "Arial"
 996|  '   Else
 997|  '     Printer.FontName = tblFonts("FontName")
 998|  '  End If
 999|  """""dDebug.Print FontCode, tmpCode, LastFontCode, FontNameUsed
1000|  Exit Sub
1001|
1002|  ERRSetFontInPrinter:
1003|  If Err = 91 Then 'Object variable not set, which pretty much means
1004|                   ' this is the first time we've executed this
1005|                   ' subroutine.
1006|    Set tblFonts = MapDB.OpenTable("tblFontList")
1007|    tblFonts.Index = "PrimaryKey"
1008|    Resume
1009|  End If
1010|
1011|
1012|
1013|  End Sub
1014|
1015|  Sub ZLoadBLOB2RealPoints (BLOBin As String, DataPts() As RealPoint)
1016|  Static S8 As string8, RlPt As RealPoint
1017|  Dim n As Long, nPoints As Long, NPos As Long
1018|  nPoints = Len(BLOBin) / 8 '8 Bytes to a RealPoint
1019|       'This implicitly updates MinX,MaxX,MinY, MaxY
1020|
1021|  NPos = 1  'NPos will keep track of where in the BLOB we are
1022|  For n = 1 To nPoints
1023|     S8.f = Mid$(BLOBin, NPos, 8)
1024|     LSet RlPt = S8
1025|     DataPts(n) = RlPt
1026|     MinX = MinR4(MinX, RlPt.X)
1027|     MinY = MinR4(MinY, RlPt.Y)
1028|     MaxX = MaxR4(MaxX, RlPt.X)
```

```
1029|     MaxY = MaxR4(MaxY, RlPt.Y)
1030|
1031|     NPos = NPos + 8
1032|  Next n
1033|
1034|  End Sub
1035|
1036|  Sub ZZ_OLDLoadTiles (snpTiles As Snapshot, TilesOut() As TileDeclPlain, ByVal
       iflgReplace As Integer, BoundBox As RealRect)
1037|  ' nStart is the first location to put data into.  (i.e, the number of previously
1038|  '    added tiles
1039|  'SQLStatement is the SQL Statement that returns the correct fields to the map,
1040|  ' to wit, it needs to return the following fields in the specified order
1041|  ' X      Map coordinates to plot
1042|  ' Y      Map coordintates
1043|  ' Z      A variable (base table field, or calculated) whose
1044|  '         value will produce a color coded map.
1045|  ' Name, the name of the Tiles (typically, something like 10S32W34 or Pittsburgh, or
1046|  '    something that means something to the user and  can be used to find data for
       DBForm
1047|  On Error GoTo LoadTilesErr
1048|  Dim nStart As Long, nRecs As Long, n  As Long
1049|  snpTiles.MoveFirst
1050|  snpTiles.MoveLast
1051|  nRecs = snpTiles.RecordCount
1052|  If iflgReplace Then
1053|     ReDim TilesOut(1 To nRecs)
1054|     nStart = 1 'Note that nStart is an offset.  This lets us avoid a lot of n-1 non-sense.
1055|   Else
1056|     nStart = UBound(TilesOut) + 1
1057|     If UBound(TilesOut) = LBound(TilesOut) Then 'We have an unitialized array, so treat it
       like the overwrite case.
1058|        nStart = 1
1059|        ReDim TilesOut(1 To nRecs)
1060|       Else
1061|        ReDim Preserve TilesOut(1 To nStart + nRecs - 1)
1062|     End If
1063|  End If
1064|
1065|  frmMap.pnl3dPercentDone.FloodPercent = 0
1066|  frmMap.lblPercentDone = "Loading Tiles:" & Format$(nRecs)
1067|
1068|  MinX = 1E+34
1069|  MinY = 1E+34
1070|  MaxX = -1E+34
1071|  MaxY = -1E+34
1072|  Dim SumZ As Single, SumSqZ As Single', MinZ As Single, MaxZ As Single
1073|  MinZ = 1E+34
1074|  MaxZ = -1E+34
1075|
1076|  Dim Pt As RealPoint
1077|  Dim X As Single, Y As Single, L As Long
1078|  snpTiles.MoveFirst
1079|
```

```
1080  For n = nStart To UBound(TilesOut)  ' For all retrieved records____
1081   X = DSng(snpTiles("X"))
1082     MinX = MinR4(X, MinX)
1083     MaxX = MaxR4(X, MaxX)
1084    Pt.X = X
1085
1086   Y = DSng(snpTiles("Y"))
1087  'Find the Min and Max of Y
1088     MinY = MinR4(Y, MinY)
1089     MaxY = MaxR4(Y, MaxY)
1090     Pt.Y = Y
1091
1092   TilesOut(n).nStart = hGMemLastPoint
1093   L = TilesOut(n).nStart
1094   Call SetRlPntsToGlobalMem(hGMemAllPoints, Pt, hGMemLastPoint, lngOne)
1095
1096
1097
1098   TilesOut(n).Zval = DSng(snpTiles("Z"))
1099  'Find the Min and Max of Y
1100     MinZ = MinR4(TilesOut(n).Zval, MinZ)
1101     MaxZ = MaxR4(TilesOut(n).Zval, MaxZ)
1102     SumZ = SumZ + TilesOut(n).Zval
1103     SumSqZ = SumSqZ + TilesOut(n).Zval * TilesOut(n).Zval
1104
1105   TilesOut(n).FeatureName = DStr(snpTiles("FeatureName"))
1106   snpTiles.MoveNext
1107  frmMap.pnl3dPercentDone.FloodPercent = (n - nStart) / nRecs * 100
1108
1109
1110  Next n
1111  frmMap.pnl3dPercentDone.FloodPercent = 0
1112  frmMap.lblPercentDone = "Done loading tiles"
1113
1114  BoundBox.Left = MinX
1115  BoundBox.Right = MaxX
1116  BoundBox.Top = MaxY
1117  BoundBox.Bottom = MinY
1118
1119
1120  Dim intTemp As Integer
1121
1122
1123  'If (frmTileOptions.optLinMinMax) Then
1124  '    ColorScaleFactor = nSteps / (MaxZ - MinZ)
1125  '    For n = nStart To UBound(TilesOut)  ' For all retrieved records____
1126  '       intTemp = (TilesOut(n).Zval - MinZ) * ColorScaleFactor
1127  '       If intTemp < 0 Then intTemp = 0
1128  '       If intTemp > nSteps Then intTemp = nSteps
1129  '       TilesOut(n).ZColor = Colors(intTemp)
1130  '    Next n
1131  'End If
1132  '
1133  'MeanZ = SumZ / nRecs
```

```
1134| 'StdDevZ = Sqr((nRecs * SumSqZ - SumZ) / (nRecs * nRecs - nRecs))
1135| 'If (frmTileOptions.optMeanSD) Then
1136| '   ColorScaleFactor = nSteps / (nStdDev * StdDevZ)
1137| "Now, loop through and assign colors to all the tiles.
1138| '   For n = nStart To UBound(TilesOut)  ' For all retrieved records____
1139| '       intTemp = (nSteps \ 2 + (TilesOut(n).Zval) - MeanZ) * ColorScaleFactor
1140| '       If intTemp < 0 Then intTemp = 0
1141| '       If intTemp > nSteps Then intTemp = nSteps
1142| '       TilesOut(n).ZColor = Colors(intTemp)
1143| '
1144| '   Next n
1145| 'End If
1146| Exit Sub
1147| LoadTilesErr:
1148| 'This jumping out for a no returned fields query
1149| ' causes problems down the road.
1150| 'Maybe forcing err=0 will work.
1151| Err = 0
1152| Exit Sub
1153| Resume Next
1154|
1155| '    If (frmTileOptions.optDiscrete) Then jj = Z    Possibly read a color directly from the
     query later on.
1156|
1157| End Sub
1158|
```

```
 1| Option Explicit
 2| Dim DB  As Database
 3|
 4|
 5|
 6|
 7| Dim nLevels As Long
 8| 'Type ColorLUT  'Color Look-Up Table
 9| '   Min As Single
10| '   Max As Single
11| '   Class As Long
12| '   VBColor As Long
13| '   GDIColor As Long
14| 'End Type
15|
16|
17| Dim ColorLookup() As ColorLUT
18|
19| Dim mySNPSource As String, strStatField As String
20|
21| Function GetLUTNameFromUser (DBName As String) As String
22| On Error Resume Next
23|  frmLUTIndex!Data1.DatabaseName = DBName
24|  If Err <> 0 Then Exit Function
25|  frmLUTIndex!Data1.Refresh
26|  If Err <> 0 Then
27|    If Err = 3011 Then
28|       Dim DB As Database
29|       Set DB = OpenDatabase(DBName)
30|       Call ZDefineLUTIndexTable(DB)
31|       DB.Close
32|       Set DB = Nothing
33|    End If
34|
35|    Exit Function
36|  End If
37|  frmLUTIndex.Show 1
38|  If frmLUTIndex!cmdOK.FontItalic Then
39|     GetLUTNameFromUser = frmLUTIndex!txtLookUpTableName
40|    Else
41|     GetLUTNameFromUser = ""
42|  End If
43| 'Discard the form
44|  Unload frmLUTIndex
45|  Set frmLUTIndex = Nothing
46| End Function
47|
48| Sub LoadColorLUT (strLoadFrom As String)
49|
50|  On Error GoTo ERRmfOpenSymLUT
51|   Dim LUTToRead As table, LUTName As String
52|   LUTName = strLoadFrom
53|   If LUTName = "" Then Exit Sub
54|   Set LUTToRead = DB.OpenTable(LUTName)
```

```
 55|  If LUTToRead.Fields.Count < 5 Then GoSub ChooseNewSymLUT
 56|
 57|  LUTToRead.MoveFirst : LUTToRead.MoveLast : LUTToRead.MoveFirst
 58|  Dim nRecs As Long, n As Long
 59|
 60|
 61|
 62|  nRecs = LUTToRead.RecordCount
 63|   nLevels = nRecs
 64|  ReDim ColorLookup(1 To nRecs)
 65|  For n = 1 To nRecs
 66|    ColorLookup(n).Class = LUTToRead("Class")
 67|    ColorLookup(n).Min = LUTToRead("Minimum")
 68|    ColorLookup(n).Max = LUTToRead("Maximum")
 69|    ColorLookup(n).VBColor = LUTToRead("VBColor")
 70|    ColorLookup(n).GDIColor = LUTToRead("GDIColor")
 71|    LUTToRead.MoveNext
 72|  Next n
 73|
 74|
 75| Exit Sub
 76| ERRmfOpenSymLUT:
 77|
 78| Select Case Err
 79|    Case 3011 'could not find object
 80|      GoSub ChooseNewSymLUT
 81|      Resume
 82|    Case 3018 'could not find one of the fields
 83|      GoSub ChooseNewSymLUT
 84|      Resume
 85|    Case Else
 86|     MsgBox "Error #" & Err & " occurred trying to load a Symbol Lookup Table." & CRLF
    & Error
 87|     Err = 0
 88|     Exit Sub
 89|     Resume
 90| End Select
 91| ChooseNewSymLUT:
 92|  LUTName = InputBox$("Could not open the lookup table " & LUTName & "." & CRLF &
    "Please choose a new name.")
 93|  If LUTName = "" Then Exit Sub
 94|  Set LUTToRead = DB.OpenTable(LUTName)
 95|
 96| 'Bit here is about the current lut not being valid.
 97|
 98|
 99|
100|
101| Return
102|
103| End Sub
104|
105| Sub PrintColorLegend ()
106| Dim m As Integer, n As Integer, DummyInt As Integer
```

```
107| On Error GoTo ERRPrintLegend
108| Printer.ScaleMode = 6'Set to millimeters
109| Dim PrinterY As Single
110| GoSub PrintHeader
111| PrinterY = 10
112|
113| Printer.FillStyle = 0'Solid fill in fillcolor
114|  For n = 1 To nLevels
115|    PrinterY = PrinterY + 20
116|    If Printer.ScaleHeight - PrinterY < 30 Then
117|
118|    'Maybe print a legend or some such here.
119|       PrinterY = 30
120|       GoSub PrintFooter
121|       Printer.NewPage
122|       GoSub PrintHeader
123|    End If
124|    Printer.CurrentY = PrinterY
125|    Printer.CurrentX = 40
126|    'Printer.FillColor = ColorLookup(n).VBColor
127|
128|    Printer.Line (35, PrinterY - 5)-(45, PrinterY - 5), ColorLookup(n).VBColor, BF
129|    Printer.Line (35, PrinterY - 5)-(45, PrinterY - 5), 0&, B
130|    Printer.CurrentY = PrinterY
131|    Printer.CurrentX = 50
132|    Printer.FontName = "Arial"
133|    Printer.FontSize = 12
134|    DummyInt = SetTextAlign(Printer.hDC, TA_TOP)
135|    Printer.Print ColorLookup(n).Min & " to " & ColorLookup(n).Max
136|
137|  Next n
138| Printer.EndDoc
139| Exit Sub
140|
141| ERRPrintLegend:
142| Select Case Err
143|    Case Else
144|     MsgBox "Error #" & Err & " encountered while printing legend. " & CRLF & Error
145|     Printer.EndDoc
146|     Err = 0
147|     Exit Sub
148|     Resume
149| End Select
150|
151| PrintHeader:
152|  Printer.FontName = "Arial": Printer.FontSize = 12
153|  Printer.CurrentX = 35
154|  Printer.CurrentY = 10
155|  Printer.Print "QUERY: " & mySNPSource
156|  Printer.CurrentX = 35
157|  Printer.Print "FIELD: " & strStatField
158|  Return
159|
160| PrintFooter:
```

```
Printer.FontName = "Arial": Printer.FontSize = 12
Printer.CurrentX = 25
Printer.CurrentY = Printer.ScaleHeight - 25
Printer.Print "Legend printed from WHEAT Symbol Lookup Table Editor on " & Now
Return

End Sub

Sub ShowColorLegend (strCLUTName As String)
 Set DB = MapDB

 Call LoadColorLUT(strCLUTName)

 Dim SourceName As String, VarIn As String, VarOut As String, LUTType As String, Comments As String
 Dim LUTName  As String
 Dim nCount As Long
 Do
  Call z_Read1_LUTIndexEntry(DB, LUTName, SourceName, VarIn, VarOut, LUTType, Comments)
  nCount = nCount + 1
  If nCount > 5000 Then Exit Sub
 Loop Until UCase$(LUTName) = UCase(strCLUTName)
 strStatField = VarIn

 mySNPSource = SourceName

 Call z_Read1_LUTIndexEntry(DB, "CLOSE", SourceName, VarIn, VarOut, LUTType, Comments)
 Set DB = Nothing


Dim m As Integer, n As Integer, DummyInt As Integer
On Error Resume Next
Dim frmLegend As New frmSymbolLegend


frmLegend.Show
frmLegend.picLegend.Cls

frmLegend.Caption = "Colors based on " & strStatField & " in " & mySNPSource
frmLegend.picLegend.Height = nLevels * 15 + 20
frmLegend.VScroll.Max = nLevels * 15

frmLegend.picLegend.FontBold = False
frmLegend.picLegend.FontName = "Arial"
frmLegend.picLegend.FontSize = 10
frmLegend.picLegend.Print "QUERY: " & mySNPSource
frmLegend.picLegend.Print "FIELD: " & strStatField
  'frmLegend.picLegend.FillStyle = 0'Solid fill
For n = 1 To nLevels

  frmLegend.picLegend.CurrentY = n * 15
  frmLegend.picLegend.CurrentX = 6
```

```
212|     frmLegend.picLegend.FillColor = ColorLookup(n).VBColor
213|     frmLegend.picLegend.Line (1, n * 15 - 5)-(11, n * 15 + 5), ColorLookup(n).VBColor,
    BF
214|     frmLegend.picLegend.Line (1, n * 15 - 5)-(11, n * 15 + 5), 0&, B
215|
216|     'frmLegend.picLegend.FontName = FontNameFromIndex(ColorLookup(n).FontCode)
217|     'frmLegend.picLegend.FontSize = 18
218|     'DummyInt = SetTextAlign(frmLegend.picLegend.hDC, TA_BASELINE)
219|     'frmLegend.picLegend.Print Chr$(ColorLookup(n).SymbolCode)
220|     frmLegend.picLegend.CurrentY = n * 15
221|     frmLegend.picLegend.CurrentX = 12
222|     frmLegend.picLegend.FontSize = 12
223|     frmLegend.picLegend.FontName = "Arial"
224|     DummyInt = SetTextAlign(frmLegend.picLegend.hDC, TA_TOP)
225|     frmLegend.picLegend.Print ColorLookup(n).Min & " to " & ColorLookup(n).Max
226|     'frmLegend.picLegend.CurrentY = frmLegend.picLegend.CurrentY + 2 *
    frmLegend.picLegend.TextHeight(ColorLookup(n).Output)
227|   Next n
228|
229|
230| End Sub
231|
232| Private Sub z_Add_LUTIndexEntry (DB As Database, LUTName As String, SourceName
    As String, VarIn As String, VarOut As String, LUTType As String, Comments As String)
233| 'z_Add_LUTIndexEntry (DB, LUTName, SourceName, VarIn, VarOut, LUTType,
    Comments)
234| On Error Resume Next
235|   Dim dns As dynaset
236|   Set dns = DB.CreateDynaset("zWHEATz_LookupTableIndex")
237|   If Err <> 0 Then
238|     Err = 0
239|     Call ZDefineLUTIndexTable(DB)
240|     Set dns = DB.CreateDynaset("zWHEATz_LookupTableIndex")
241|   End If
242|
243|   dns.AddNew
244|     dns("LookUpTableName") = LUTName
245|     dns("TableOrQueryName") = SourceName
246|     dns("InputVariable") = VarIn
247|     dns("OutputVariable") = VarOut
248|     dns("LookUpTableType") = LUTType
249|     dns("Comments") = Comments
250|   dns.Update
251| End Sub
252|
253| Private Sub z_Read1_LUTIndexEntry (DB As Database, LUTName As String,
    SourceName As String, VarIn As String, VarOut As String, LUTType As String,
    Comments As String)
254| 'z_Read1_LUTIndexEntry (DB, LUTName, SourceName, VarIn, VarOut, LUTType,
    Comments)
255| On Error Resume Next
256|   Static dns As dynaset
257|   If LUTName = "CLOSE" Then
258|     dns.Close
```

```
259|      Set dns = Nothing
260|      Exit Sub
261|    End If
262|
263|
264|    If dns Is Nothing Then
265|      Set dns = DB.CreateDynaset("zWHEATz_LookupTableIndex")
266|      dns.MoveFirst
267|      If Err <> 0 Then
268|        Err = 0
269|        Call ZDefineLUTIndexTable(DB)
270|        Beep
271|        Exit Sub
272|      End If
273|    End If
274|
275|      LUTName = dns("LookUpTableName")
276|      SourceName = dns("TableOrQueryName")
277|      VarIn = dns("InputVariable")
278|      VarOut = dns("OutputVariable")
279|      LUTType = dns("LookUpTableType")
280|      Comments = dns("Comments")
281|    dns.MoveNext
282| End Sub
283|
284| Private Sub ZDefineLUTIndexTable (DB As Database)
285| " This will define a table to contain a listing of LUTs that have been saved to disk
286|
287| Dim nfName As New field, nfTable As New field, nfLUTType  As New field
288| Dim nfVarIn As New field, nfVarOut As New field, nfComments  As New field
289|   Const DB_TEXT = 10
290|   Const DB_MEMO = 12
291|    nfName.Name = "LookUpTableName": nfName.Type = DB_TEXT: nfName.Size =
    255
292|    nfTable.Name = "TableOrQueryName": nfTable.Type = DB_TEXT: nfTable.Size = 255
293|    nfVarIn.Name = "InputVariable": nfVarIn.Type = DB_TEXT: nfVarIn.Size = 255
294|    nfVarOut.Name = "OutputVariable": nfVarOut.Type = DB_TEXT: nfVarOut.Size = 255
295|    nfLUTType.Name = "LookUpTableType": nfLUTType.Type = DB_TEXT:
    nfLUTType.Size = 255
296|    nfComments.Name = "Comments": nfComments.Type = DB_MEMO
297|
298|
299| Dim tdf As New tabledef
300|    tdf.Fields.Append nfName
301|    tdf.Fields.Append nfTable
302|    tdf.Fields.Append nfVarIn
303|    tdf.Fields.Append nfVarOut
304|    tdf.Fields.Append nfLUTType
305|    tdf.Fields.Append nfComments
306|
307| Dim ndx As New index
308|    ndx.Fields = "LookUpTableName"
309|    ndx.Primary = True
310|    ndx.Name = "PrimaryKey"
```

```
311|
312|  tdf.Name = "zWHEATz_LookupTableIndex"
313|
314|  DB.TableDefs.Append tdf
315| End Sub
316|
```

```
 1| Option Explicit
 2| 'Variables that are declared so that frmGridDataEdit can
 3| '  be shared between MapViewQuery and GridEdit
 4| Global dynViewInGrid As dynaset, DB As Database
 5| Global OriginalCursorStyle As Integer
 6| Global CommandLine() As String
 7|
 8| 'For help manipulation
 9| Declare Function WinHelp Lib "USER" (ByVal hWnd As Integer, ByVal strHelpFile As
   String, ByVal wCommand As Integer, ByVal dwData As Any) As Integer
10| Global Const gkiHELP_QUIT = 2
11| Global Const gkiHELP_INDEX = 3
12| Global Const gkiHELP_HELPONHELP = 4
13| Global Const gkiHELP_PARTIALKEY = &H105
14|
15|
16|
17|
18| Global Const strProgName = "WHEAT E-MAP ", strVersion = "2.0_ KGS Edition"
19|
20| Declare Function SetParent Lib "User" (ByVal hWndChild As Integer, ByVal
   hWndNewParent As Integer) As Integer
21|
22| Global tblViews As Table ' named views   'NEW 950418
23| 'Should go in the declarations module
24|
25| Dim strCoordBlurb As String, strUnitName As String
26|
27| Function CoordinateBlurb () As String
28|   'The coordinate blurb can be blank
29|   CoordinateBlurb = strCoordBlurb
30| End Function
31|
32| Function CoordinateUnits () As String
33|   If strUnitName = "" Then Call GetCoordinateInfo(MapDB)
34|   CoordinateUnits = strUnitName
35| End Function
36|
37| Sub EMapStartup ()
38| ZIntitializeGlobalArrays
39| Call ZInitializeConstants
40|
41| 'Start the global memory array of realpoints
42|   hGMemAllPoints = GetWinMem(1048567) 'Grab 1 MByte of memory for RealPoints
43|   lngHGAllPointsMemSize = GlobalSize(hGMemAllPoints)
44|   lngOne = 1
45|
46|
47|
48| 'OriginalCursorStyle = GetClassword(Me.picMap.hWnd, GCW_HCURSOR)
49| OriginalCursorStyle = GetClassword(MDI_WHEAT.picToolBar.hWnd,
   GCW_HCURSOR)
50|
51| 'Get user-modifiable default values
```

```
52|    If INIFile = "" Then INIFile = App.Path & "\" & "WHEAT.INI"
53|     Call InitializeWHEATEnv(INIFile)
54|
55| 'Set up fake databounds, so that we can have a correct scaling initially,
56| '  to be overwritten by the first valid theme.
57|     DataBounds.Left = 0
58|     DataBounds.Right = 100
59|     DataBounds.Top = 100
60|     DataBounds.Bottom = 0
61|
62| Load frmHoldCursorBitmaps
63| Unload frmHoldCursorBitmaps
64|
65|
66| End Sub
67|
68| Sub GetCoordinateInfo (DB As Database)
69| Dim rcsUnits As dynaset
70|   On Error Resume Next
71|    Set rcsUnits = DB.CreateDynaset("SELECT DISTINCTROW ztblWHEAT_LOG.Inputs,
    ztblWHEAT_LOG.Outputs, ztblWHEAT_LOG.ProgramName,
    ztblWHEAT_LOG.UserName FROM ztblWHEAT_LOG WHERE
    ((ztblWHEAT_LOG.ProgramName='units') AND (ztblWHEAT_LOG.UserName='units'));
    ")
72|
73|
74|    rcsUnits.MoveFirst
75|
76|
77|    If Err = 3021 Then 'No current record, and thus no units.
78|       Err = 0
79|       strUnitName = InputBox$("Enter the name of the coordinate units (meters, feet, ...)", ,
    "meters")
80|       strCoordBlurb = InputBox$("Enter a brief text description of this coordinate system for
    display on maps", , "UTM Zone 14 NAD27 Coordinates")
81|       rcsUnits.AddNew
82|          rcsUnits("UserName") = "units"
83|          rcsUnits("ProgramName") = "units"
84|          rcsUnits("Inputs") = strUnitName
85|          rcsUnits("Outputs") = strCoordBlurb
86|       rcsUnits.Update
87|      Else
88|
89|          strUnitName = rcsUnits("Inputs")
90|          strCoordBlurb = rcsUnits("Outputs")
91|    End If
92|
93|
94|
95| End Sub
96|
97| Sub LoadMDB ()
98|
99| On Error GoTo ERRLoadMDB
```

```
100|   Call GetCoordinateInfo(MapDB)
101| Set snpTblQryList = MapDB.ListTables()
102| snpTblQryList.MoveFirst
103| snpTblQryList.MoveLast
104| Dim nNonSys As Long
105| nNonSys = 0
106| nTblQrys = snpTblQryList.RecordCount
107| snpTblQryList.MoveFirst
108|
109| For i = 0 To nTblQrys - 1
110|   'dDebug.Print snpTblQryList("Name"), snpTblQryList("Attributes")
111|  If Not ((snpTblQryList("Attributes") And DB_SYSTEMOBJECT) <> 0) Then
112|   'Exclude system objects from list of tables and queries presented to user
113|
114|
115|   nNonSys = nNonSys + 1
116|   ReDim Preserve strTblQryList(nNonSys)
117|   strTblQryList(nNonSys) = snpTblQryList("Name")
118|  End If
119|  snpTblQryList.MoveNext
120| Next i
121|
122| nTblQrys = nNonSys
123| 'After the above loop finishes, strTblQryList(1) to (nNonSys) contain
124| ' the names of all the non-system tables in the database
125| 'nTblQrys now contains the number of non-system tables and queries.
126| snpTblQryList.Close
127|  'GWPouch 951202 Rather than explicitly clearing the lists and reloading them,
128|  '  throw the forms away
129|  Call UnloadThemeDefForms
130|   'Call ZResetTblQryList
131|
132| 'Load the font-listing table for this Wheat MDB, if necessary, make one.
133| Set tblFonts = MapDB.OpenTable("zWHEATtblFontList")
134|
135|
136| Call LoadFontTable(tblFonts, FontTable())
137|  'GWPouch 951202 'Rather than doing this, which loads the forms, leave them
138|  ' alone and fill these comboboxes in the form load event
139|  '   Call LoadComboBoxWithFontNames(frmPointOptionsSimple.cmbFontList)
140|  '   Call LoadComboBoxWithFontNames(frmLabelOptions.cmbFontList)
141|
142| tblFonts.Close
143| Set tblFonts = Nothing
144|
145|
146|
147|  Exit Sub
148|
149| ERRLoadMDB:
150|
151| If Err = 3011 Then ' no tblFontList in the current database, so create one after prompting
    the user.
152|       Call DefineFontTable(MapDB)'  Dim tdfFonts As New tabledef, ndxFonts As New
```

```
    Index, fldName As New field, fldNumber As New field, fldExtra As New field
153|        Resume
154| End If
155|  MsgBox "Error #" & Err & " while opening database.  " & Error
156|  Err = 0
157|  Exit Sub
158|  Resume
159|
160| End Sub
161|
162| Sub Main ()
163| 'WHEAT E-MAP startup module.  By putting the commandline parsing here,
164| '  we can make the frmMap multiply instanced.
165| CRLF = Chr$(13) & Chr$(10)
166|
167|
168| On Error GoTo ERRmfOpen2
169|
170| Dim nWord As Long, flgGotString As Integer
171| ReDim CommandLine(1 To 1)
172| Dim CommandIn As String
173| CommandIn = UCase$(Command$)
174|
175| flgGotString = True
176| nWord = 1
177| Do While flgGotString
178|   ReDim Preserve CommandLine(1 To nWord)
179|   CommandLine(nWord) = ExtractDelimitedString(CommandIn, " ", nWord)
180|   'ddDebug.Print nWord, CommandLine(nWord)
181|
182|   If CommandLine(nWord) = "" Then
183|     flgGotString = False
184|     Else
185|     flgGotString = True
186|   End If
187|   nWord = nWord + 1
188| Loop
189| nWord = nWord - 1
190| Dim nW As Long
191|
192| For nW = 1 To nWord
193|   If InStr(CommandLine(nW), ".MDB") Then strMapDBName = CommandLine(nW)
194|   If InStr(CommandLine(nW), ".INI") Then INIFile = CommandLine(nW)
195|   'Possibly add more command line arguments here.
196| Next
197| If INIFile = "" Then INIFile = App.Path & "\" & "WHEAT.INI"
198|
199|   Call ShowAboutBoxAtStartup
200|
201|   DoEvents
202|
203|   Load frmMap
204|
205|
```

```
206|
207|
208| If strMapDBName <> "" Then
209|   Set MapDB = OpenDatabase(strMapDBName, False, False, "") 'Open the desired
      database
210|   LoadMDB
211|   frmMap.Show
212|   frmMap.SetFocus
213|   SendKeys "%F"
214|   SendKeys "D"
215|   SendKeys "{ESCAPE}"
216|   SendKeys "%F"
217|   SendKeys "O"
218|   SendKeys "{ENTER}"
219|   MDI_WHEAT.Caption = strMapDBName & " " & strProgName & " " & strVersion
220|  Else
221|   frmMap.Show
222|   SendKeys "%Fd"
223|
224| End If
225|
226|
227|
228|
229| Exit Sub
230| ERRmfOpen2:
231| Select Case Err
232| Case 3051 'Could not open file becasue someone else has exclusive lock on it.
233|   MsgBox "Someone else has opened that database in exclusive mode." & CRLF & "You
      can either get the other user or application to close it and re-open it in non-exclusive
      mode, or open in it read-only mode."' & CRLF & "Do you wish to open it in read only
      mode?", 35, "Error opening database")
234|     strMapDBName = ""
235|
236|     Err = 0
237|     Exit Sub
238|     Resume Next
239|
240|  Case Else
241|   MsgBox "Error #" & Err & " occurred while opening the database." & CRLF & Error
242|   Err = 0
243|   Exit Sub
244|  Resume Next
245| End Select
246|
247|
248|
249| End Sub
250|
251| Function MapScaleInWords (ByVal X_World As Single, ByVal X_Screen_mm) As String
252| 'Produces a map scale factor statement
253| Static MapScale As Single
254| MapScale = ScaleFudgeFactor() * (X_World * MapUnitsToMeters_Multiplier() * 1000) /
      (X_Screen_mm)
```

```
255|                    'Map size in world millimeters          'Map size in screen
   millimeters
256| If MapScale > 100 Then
257|    MapScaleInWords = Format$(MapScale, "#,###")
258|   Else
259|    MapScaleInWords = Format$(MapScale, "0.#")
260| End If
261| End Function
262|
263| Function MapUnitsToMeters_Multiplier () As Single
264| Select Case CoordinateUnits()
265|   Case "meters"
266|     MapUnitsToMeters_Multiplier = 1#
267|   Case "feet"
268|     MapUnitsToMeters_Multiplier = .3048
269|   Case "kilometers"
270|     MapUnitsToMeters_Multiplier = 1000#
271|   Case "millimeters"
272|     MapUnitsToMeters_Multiplier = .001
273|   Case "centimeters"
274|     MapUnitsToMeters_Multiplier = .01
275|   Case "miles"
276|     MapUnitsToMeters_Multiplier = 1609.3
277|   Case "inches"
278|     MapUnitsToMeters_Multiplier = .0254
279|   Case "yards"
280|     MapUnitsToMeters_Multiplier = .9144
281|   Case "rods"
282|     MapUnitsToMeters_Multiplier = 5.0292
283|   Case Else
284|     MapUnitsToMeters_Multiplier = 1
285| End Select
286|
287|
288|
289|
290| End Function
291|
292| Sub OpenDataBaseForEMap ()
293| 'open an old file of the form filename.mdb for a MicroSoft DataBase
294| On Error GoTo ERRmfOpen
295| frmOZ.CMDialog1(0).DialogTitle = "WHEAT Database to Display and Query"
296| frmOZ.CMDialog1(0).Filter = "MS Access Databases|*.mdb"
297| frmOZ.CMDialog1(0).Filename = strMapDBName
298| frmOZ.CMDialog1(0).Flags = OFN_FILEMUSTEXIST Or OFN_SHAREAWARE
299|
300| frmOZ.CMDialog1(0).Action = 1
301| Dim Temp As String
302|
303| Temp = UCase$(frmOZ.CMDialog1(0).Filename)
304|
305| If Temp = "" Then Exit Sub   'The user chose cancel
306| If strMapDBName <> "" And Temp = strMapDBName Then Exit Sub 'The user selected
   the same database
```

```
307|     strMapDBName = Temp
308|
309| 'XXXX made the MapDatabase read-write.  Give this some serious thought later on.
     GWP 4 Feb 94
310|     Set MapDB = OpenDatabase(strMapDBName, False, False, "") 'Open the desired
     database
311|     '         DataBase file  , exclusive, read-only, MSAccess/VBasic
312|     LoadMDB
313|
314| 'The calling form enables its own menu items. Enable dimmed menu options.
315| MDI_WHEAT.Caption = strProgName & strVersion & ": " & strMapDBName
316|
317|
318| Exit Sub
319| ERRmfOpen:
320| Select Case Err
321| Case 3051 'Could not open file becasue someone else has exclusive lock on it.
322|   MsgBox "Someone else has opened that database in exclusive mode." & CRLF & "You
     can either get the other user or application to close it and re-open it in non-exclusive
     mode, or open in it read-only mode."' & CRLF & "Do you wish to open it in read only
     mode?", 35, "Error opening database")
323|     strMapDBName = ""
324|
325|     Err = 0
326|     Exit Sub
327|     Resume Next
328|
329|  Case Else
330|   MsgBox "Error #" & Err & " occurred while opening the database." & CRLF & Error
331|   Err = 0
332|   Exit Sub
333|  Resume Next
334| End Select
335|
336| End Sub
337|
338| Sub ReadThemeDefinitionFromTableByTitle (DB As Database, TitleToFind As String,
     ThemeToRead As typTheme)
339|   'Dim tblThemes As Table
340|   Dim tblThemes As snapshot
341|   On Error GoTo ERRReadThemeFromTableByTitle
342|   'Set tblThemes = DB.OpenTable("zWHEATtblThemes")
343|   'tblThemes.Index = "Name"
344|   'tblThemes.Seek "=", TitleToFind
345|   Set tblThemes = DB.CreateSnapshot("zWHEATtblThemes")
346|   tblThemes.FindLast "ThemeName='" & TitleToFind & "'"
347|   If tblThemes.NoMatch Then
348|     MsgBox "Could not find a theme called " & TitleToFind & ".  Sorry."
349|     Exit Sub
350|   End If
351|   ThemeToRead.ThemeName = tblThemes("ThemeName")
352|   ThemeToRead.MapSource = tblThemes("MapSource")
353|   ThemeToRead.DBSource = tblThemes("DBSource")
354|   ThemeToRead.ThemeType = tblThemes("ThemeType")
```

```
355|    tblThemes.Close
356|    Set tblThemes = Nothing
357| Exit Sub
358| ERRReadThemeFromTableByTitle:
359| Select Case Err
360|  Case 3011, 3078 'Table not found
361|   Call ZDefineThemesTable(DB)
362|   MsgBox "There was no previously created table containing theme definitions.  Sorry."
363|   Exit Sub
364|   Resume
365|  Case Else
366|     MsgBox "Error #" & Err & "occurred trying to Read a Theme from Table." & CRLF &
    Error
367|     Err = 0
368|     Exit Sub
369|     Resume
370|  End Select
371| End Sub
372|
373| Sub SaveThemeToTable (DB As Database, ThemeToSave As typTheme)
374|    Dim tblThemes As Table
375|    On Error GoTo ERRSaveThemeToTable
376|    Static LastSerialNumber As Long
377|    If LastSerialNumber = 0 Then GoSub GetHighestSerialNumber
378|    Set tblThemes = DB.OpenTable("zWHEATtblThemes")
379|    tblThemes.AddNew
380|
381|     tblThemes("ThemeName") = ThemeToSave.ThemeName
382|     tblThemes("MapSource") = ThemeToSave.MapSource
383|     tblThemes("DBSource") = ThemeToSave.DBSource
384|     tblThemes("ThemeType") = ThemeToSave.ThemeType
385|     tblThemes("ThemeSerialNumber") = LastSerialNumber + 1
386|     tblThemes("DateModified") = Now
387|     tblThemes("WhoModified") = myUserName
388|
389|    tblThemes.Update
390|    tblThemes.Close
391|    Set tblThemes = Nothing
392|    LastSerialNumber = LastSerialNumber + 1
393| Exit Sub
394| GetHighestSerialNumber:
395| Dim snpHighest As snapshot
396| Set snpHighest = DB.CreateSnapshot("SELECT Max([ThemeSerialNumber]) AS
    MaxNumber FROM zWHEATtblThemes;")
397| If IsNull(snpHighest("MaxNumber")) Then
398|    LastSerialNumber = 1
399|    Else
400|     LastSerialNumber = snpHighest("MaxNumber")
401| End If
402|
403| Return
404| Exit Sub
405| ERRSaveThemeToTable:
406| Select Case Err
```

```
 407|  Case 3011, 3078 'Table not found
 408|   Call ZDefineThemesTable(DB)
 409|   Resume
 410|  Case Else
 411|    MsgBox "Error #" & Err & "occurred trying to Save a Theme to Table." & CRLF & Error
 412|    Err = 0
 413|    Exit Sub
 414|    Resume
 415|  End Select
 416|
 417| End Sub
 418|
 419| Sub ShowAboutBoxAtStartup ()
 420|    Dim sngHowLong As Single
 421|    Dim Buffer As String, nBytesBack  As Long
 422|    Buffer = Space$(1024)
 423|      nBytesBack = GetPrivateProfileString("Disclaimer", "HowLongToShowAboutBox",
      "-10", Buffer, 1024, INIFile)
 424|      sngHowLong = Val(Left$(Buffer, nBytesBack))
 425|      frmAboutBox.Enabled = True
 426|      If sngHowLong < 0 Then
 427|        frmAboutBox.Show 1 'Show it as a modal form
 428|        nBytesBack = GetPrivateProfileString("Disclaimer", "HowLongToShowAboutBox",
      "0", Buffer, 1024, INIFile)
 429|        nBytesBack = WritePrivateProfileString("Disclaimer", ByVal
      "HowLongToShowAboutBox", ByVal "60", INIFile)
 430|       Else
 431|        If sngHowLong < 10 Then sngHowLong = 10
 432|        frmAboutBox.Show 'Show it non-modally form
 433|      End If
 434|
 435| End Sub
 436|
 437| Sub UnloadThemeDefForms ()
 438|   Unload frmPointOptionsSimple
 439|   Set frmPointOptionsSimple = Nothing
 440|   Unload frmLineOptions
 441|    Set frmLineOptions = Nothing
 442|   Unload frmTileOptions
 443|    Set frmTileOptions = Nothing
 444|   Unload frmLabelOptions
 445|    Set frmLabelOptions = Nothing
 446|   Unload frmPolyOptions
 447|    Set frmPolyOptions = Nothing
 448| End Sub
 449|
 450| Sub ZDefineThemesTable (DB As Database)
 451| Dim NewThemeTable As New TableDef
 452| Dim fldMapSource As New Field, fldDBSource As New Field
 453| Dim fldThemeName As New Field, fldSerialNumber  As New Field
 454| Dim fldsngA As New Field, fldsngB As New Field, fldsngC As New Field
 455| Dim fldThemeType As New Field, fldDateModified  As New Field, fldWhoModified  As
      New Field
 456| Beep
```

```
457| Beep
458| NewThemeTable.Name = "zWHEATtblThemes"
459|
460|    fldThemeName.Name = "ThemeName"
461|    fldThemeName.Type = DB_TEXT
462|    fldThemeName.Size = 255
463|
464|    fldMapSource.Name = "MapSource"
465|    fldMapSource.Type = DB_MEMO
466|
467|    fldDBSource.Name = "DBSource"
468|    fldDBSource.Type = DB_MEMO
469|
470|    fldSerialNumber.Name = "SerialNumber"
471|    fldSerialNumber.Type = DB_LONG
472|
473|    fldThemeType.Name = "ThemeType"
474|    fldThemeType.Type = DB_LONG
475|
476|    fldDateModified.Name = "DateModified"
477|    fldDateModified.Type = DB_DATE
478|    fldWhoModified.Name = "WhoModified"
479|    fldWhoModified.Type = DB_TEXT
480|    fldWhoModified.Size = 255
481|
482|    fldsngA.Name = "sngA"
483|    fldsngA.Type = DB_SINGLE
484|    fldsngB.Name = "sngB"
485|    fldsngB.Type = DB_SINGLE
486|    fldsngC.Name = "sngC"
487|    fldsngC.Type = DB_SINGLE
488|
489| 'Append the fields to the table
490|    NewThemeTable.Fields.Append fldThemeName
491|    NewThemeTable.Fields.Append fldMapSource
492|    NewThemeTable.Fields.Append fldDBSource
493|    NewThemeTable.Fields.Append fldSerialNumber
494|    NewThemeTable.Fields.Append fldThemeType
495|    NewThemeTable.Fields.Append fldDateModified
496|    NewThemeTable.Fields.Append fldWhoModified
497|    NewThemeTable.Fields.Append fldsngA
498|    NewThemeTable.Fields.Append fldsngB
499|    NewThemeTable.Fields.Append fldsngC
500| 'Add index on serial number
501| Dim ndxPrimary As New Index, ndxName  As New Index
502|
503|    ndxPrimary.Fields = "SerialNumber"
504|    ndxPrimary = True
505|    NewThemeTable.Indexes.Append ndxPrimary
506|    NewThemeTable.Indexes(0).Name = "PrimaryKey"
507|
508|    ndxName.Fields = "ThemeName"
509|    ndxName = False
510|    NewThemeTable.Indexes.Append ndxName
```

```
511|     NewThemeTable.Indexes(1).Name = "Name"
512|
513|  ' Append the table to the table defs collection.
514|     DB.TableDefs.Append NewThemeTable
515|  End Sub
516|
517|  Sub zDefineViewTable (DB As Database)
518|  'NEW 950418
519|  Dim NewViewTable As New TableDef
520|  Dim nfldName As New Field, nfldSerialNum As New Field
521|  Dim nfldMinX As New Field, nfldMinY  As New Field
522|  Dim nfldMaxX As New Field, nfldMaxY As New Field
523|  Dim nfldWhenAdded As New Field
524|  Beep
525|  Beep
526|  NewViewTable.Name = "zWHEATViews"
527|
528|     nfldName.Name = "ViewName": nfldName.Type = DB_TEXT: nfldName.Size = 255
529|     nfldSerialNum.Name = "ViewSerialNumber": nfldSerialNum.Type = DB_LONG
530|     nfldMinX.Name = "MinX": nfldMinX.Type = DB_SINGLE
531|     nfldMinY.Name = "MinY": nfldMinY.Type = DB_SINGLE
532|     nfldMaxX.Name = "MaxX": nfldMaxX.Type = DB_SINGLE
533|     nfldMaxY.Name = "MaxY": nfldMaxY.Type = DB_SINGLE
534|
535|     nfldWhenAdded.Name = "View Created": nfldWhenAdded.Type = DB_DATE
536|
537|  'Append the fields to the table
538|     NewViewTable.Fields.Append nfldName
539|     NewViewTable.Fields.Append nfldSerialNum
540|     NewViewTable.Fields.Append nfldMinX
541|     NewViewTable.Fields.Append nfldMinY
542|     NewViewTable.Fields.Append nfldMaxX
543|     NewViewTable.Fields.Append nfldMaxY
544|     NewViewTable.Fields.Append nfldWhenAdded
545|  'Add index on serial number
546|  Dim ndxPrimary As New Index
547|
548|     ndxPrimary.Fields = "ViewSerialNumber"
549|     ndxPrimary = True
550|     NewViewTable.Indexes.Append ndxPrimary
551|     NewViewTable.Indexes(0).Name = "PrimaryKey"
552|
553|  ' Append the table to the table defs collection.
554|     DB.TableDefs.Append NewViewTable
555|
556|  End Sub
557|
558|  Sub ZIntitializeGlobalArrays ()
559|  ReDim Points(0 To 0)
560|  ReDim Tiles(0 To 0)
561|  ReDim Lines(0 To 0)
562|  ReDim HatchLines(0 To 0)
563|  ReDim Polys(0 To 0)
564|  ReDim Labels(0 To 0)
```

```
ReDim QSDataOut(0 To 0)

ReDim UserLine(1)
ReDim RasterGrids(0 To 0)

End Sub

Sub ZResetTblQryList ()
On Error Resume Next

frmTileOptions.cmbTblQryList.Clear
frmPointOptionsSimple.cmbTblQryList.Clear
frmLineOptions.cmbTblQryList.Clear
frmPolyOptions.cmbTblQryList.Clear
frmLabelOptions.cmbTblQryList.Clear

For i = 1 To nTblQrys
 frmTileOptions.cmbTblQryList.AddItem strTblQryList(i)
 frmPointOptionsSimple.cmbTblQryList.AddItem strTblQryList(i)
 frmLineOptions.cmbTblQryList.AddItem strTblQryList(i)
 frmPolyOptions.cmbTblQryList.AddItem strTblQryList(i)
 frmLabelOptions.cmbTblQryList.AddItem strTblQryList(i)
Next i

End Sub

```

```
Option Explicit
Dim flgLoadDIB As Long, flgSkipRepainting As Long, ReturnVal As Long
Dim ViewBounds As RealRect, GDIBounds As APIRECT
Dim OriginalScale As RealRect, PrintScale As RealRect
Dim sngWidthMultiplier As Single 'Used to control printed line width
Dim mThemesToPrint() As typTheme

'Fonts Dialog Flags
Global Const CF_SCREENFONTS = &H1&
Global Const CF_PRINTERFONTS = &H2&
Global Const CF_BOTH = &H3&
Global Const CF_SHOWHELP = &H4&
Global Const CF_INITTOLOGFONTSTRUCT = &H40&
Global Const CF_USESTYLE = &H80&
Global Const CF_EFFECTS = &H100&
Global Const CF_APPLY = &H200&
Global Const CF_ANSIONLY = &H400&
Global Const CF_NOVECTORFONTS = &H800&
Global Const CF_NOSIMULATIONS = &H1000&
Global Const CF_LIMITSIZE = &H2000&
Global Const CF_FIXEDPITCHONLY = &H4000&
Global Const CF_WYSIWYG = &H8000&       'must also have CF_SCREENFONTS &
CF_PRINTERFONTS
Global Const CF_FORCEFONTEXIST = &H10000
Global Const CF_SCALABLEONLY = &H20000
Global Const CF_TTONLY = &H40000
Global Const CF_NOFACESEL = &H80000
Global Const CF_NOSTYLESEL = &H100000
Global Const CF_NOSIZESEL = &H200000

Sub GetPixelBoundsPrinter (PixelRect As APIRECT)
  Dim ScalingB4 As RealRect
SaveScalingPrinter ScalingB4
'Move the currentpoint to the upper left corner of the
'  ViewBounds
  Printer.CurrentX = ViewBounds.Left
  Printer.CurrentY = ViewBounds.Top
'Change the scalemode to pixels.
  Printer.ScaleMode = 3
'Get the pixel location of the point corresponding to
' the UpperLeft corner of ViewBounds and store it.
  PixelRect.Left = Printer.CurrentX
  PixelRect.Top = Printer.CurrentY
'Repeat for the lower right corner
  RestoreScalingPrinter ScalingB4
  Printer.CurrentX = ViewBounds.Right
  Printer.CurrentY = ViewBounds.Bottom
  Printer.ScaleMode = 3
  PixelRect.Bottom = Printer.CurrentY
  PixelRect.Right = Printer.CurrentX

  RestoreScalingPrinter ScalingB4

End Sub
```

```
 54|
 55| Private Function OnePixelInWorldCoordinates () As Single
 56| Dim Before As Single, After As Single
 57| Dim LLeft As Single, TTop As Single, WWidth As Single, HHeight As Single
 58| LLeft = Printer.ScaleLeft
 59| TTop = Printer.ScaleTop
 60| WWidth = Printer.ScaleWidth
 61| HHeight = Printer.ScaleHeight
 62|
 63| Before = Printer.CurrentX
 64|
 65| Printer.ScaleMode = 3'pixel
 66|
 67| Printer.CurrentX = Printer.CurrentX + 1
 68|
 69| Printer.ScaleMode = 0
 70|
 71| Printer.ScaleLeft = LLeft
 72| Printer.ScaleTop = TTop
 73| Printer.ScaleWidth = WWidth
 74| Printer.ScaleHeight = HHeight
 75| After = Printer.CurrentX
 76| OnePixelInWorldCoordinates = After - Before
 77|
 78| End Function
 79|
 80| Sub PrintDrawGlobalHatchLine (hGMemAllPoints As Integer, nStart As Long, nPoints As
     Long)
 81|  If nPoints = 0 Then Exit Sub
 82| On Error Resume Next
 83|  ReDim RIPts(1 To nPoints) As RealPoint
 84|  Dim n  As Long
 85|    Call GetRIPntsFromGlobalMem(hGMemAllPoints, RIPts(1), nStart, nPoints)
 86|    For n = 1 To (nPoints - 1) Step 2
 87|       Printer.Line (RIPts(n).X, RIPts(n).Y)-(RIPts(n + 1).X, RIPts(n + 1).Y)
 88|    Next n
 89| End Sub
 90|
 91| Sub Printer_PrintInBox (strText As String, rctFitIn As RealRect)
 92| ' call Printer_PrintInBox (strText , rctFitIn )
 93| "This routine will print the text strText starting in
 94| " the upper left of rctFitIn.  Line wraps will occur at
 95| " rctFitIn.Right (rctFitIn.Bottom is ignored)
 96|
 97| 'It is assumed the caller has already selected the font
 98| ' and such for the Printer
 99|
100| Dim strCopy As String, strThisLine As String, ThisWord As String, strMaybe
101| Dim nChar As Long, nCh As Long, xMax As Single, xSoFar As Single
102|
103|  strCopy = Trim$(strText)
104|  xMax = rctFitIn.Right - rctFitIn.Left
105|
106|  strThisLine = "": strMaybe = ""
```

```
107|  Printer.CurrentX = rctFitIn.Left
108|  Printer.CurrentY = rctFitIn.Top
109|
110|  xMax = xMax
111|  'Maybe set text align
112|  Do
113|    ThisWord = NextWord(strCopy, nChar)
114|    strMaybe = strThisLine & " " & ThisWord
115|    xSoFar = Printer.TextWidth(strMaybe)
116|    If xSoFar > xMax Then
117|      Printer.CurrentX = rctFitIn.Left
118|      Printer.Print strThisLine
119|      strThisLine = ThisWord
120|     Else
121|      strThisLine = strThisLine & " " & ThisWord
122|    End If
123|
124|  Loop Until ThisWord = ""
125|  Printer.CurrentX = rctFitIn.Left
126|  Printer.Print strThisLine
127|
128|
129|
130|  End Sub
131|
132|  Sub PrinterCircleQSResults (QSDataOut() As typTaggedIDNum, lngQSHighlightColor
     As Long, sngQSHighlightRadius As Single, LastQS As RealPoint)
133|   Dim n As Long, nFeatures As Long
134|   Dim lb As Long, ub As Long
135|   On Error Resume Next
136|   lb = LBound(QSDataOut)
137|   ub = UBound(QSDataOut)
138|   For n = lb To ub
139|     Printer.Circle (QSDataOut(n).X, QSDataOut(n).Y), sngQSHighlightRadius,
    lngQSHighlightColor
140|   Next n
141|   If LastQS.X <= ViewBounds.Right And LastQS.X >= ViewBounds.Left And LastQS.Y >=
    ViewBounds.Bottom And LastQS.Y <= ViewBounds.Top Then Call
    ZPrinterBeachBall(LastQS.X, LastQS.Y)
142|  End Sub
143|
144|  '
145|  Sub PrintHatchLines3 (FirstLine As Long, LastLine As Long, nTheme As Long) '
146|  'GWPouch 940531  The ShowTYPE3 series of subroutines checks to see
147|  '  if the desired feature lies inside, or close to the viewport
148|  '  before trying to draw it.  This should avoid many overflow errors
149|  '  and the like.
150|  On Error GoTo ERRShowHatchLines3
151|
152|  Dim n As Long, nLines As Long, flgSkipIt As Integer
153|  If (UBound(HatchLines) = 0) Then Exit Sub
154|  nLines = LastLine - FirstLine + 1
155|  frmMap.pnl3dPercentDone.FloodPercent = 0
156|  frmMap.lblPercentDone = "Printing hatch lines from " & Themes(nTheme).ThemeName
```

```
157|
158| For n = FirstLine To LastLine
159|  DoEvents
160|  If flgStopPrinting Then Exit Sub
161|  flgSkipIt = False
162| ' Check whether the polygon, or whatever, of which this is the hatching
163| '  lies outside the current viewport.
164|  If HatchLines(n).MaxX < ViewBounds.Left Then flgSkipIt = True
165|  If HatchLines(n).MinX > ViewBounds.Right Then flgSkipIt = True
166|  If HatchLines(n).MinY > ViewBounds.Top Then flgSkipIt = True
167|  If HatchLines(n).MaxY < ViewBounds.Bottom Then flgSkipIt = True
168|
169|    Printer.DrawWidth = HatchLines(n).ZWidth
170|    Printer.ForeColor = HatchLines(n).ZColor
171|    Printer.DrawStyle = HatchLines(n).ZStyle
172|
173|    If flgSkipIt = False Then
174|  Call PrintDrawGlobalHatchLine(hGMemAllPoints, HatchLines(n).nStart,
     HatchLines(n).nPoints)
175|    End If
176|
177|    frmMap.pnl3dPercentDone.FloodPercent = (n - FirstLine) / nLines * 100
178| Next n
179|
180| frmMap.pnl3dPercentDone.FloodPercent = 100
181| frmMap.lblPercentDone = "Done PRINTING hatch lines from " &
     Themes(nTheme).ThemeName
182| Exit Sub
183|
184| ERRShowHatchLines3:
185| flgSkipIt = True
186| Resume Next
187|
188|
189|
190| End Sub
191|
192| Sub PrintLabels1 (FirstLabel As Long, LastLabel As Long, nTheme As Long)
193| Dim n As Long, Pt  As RealPoint
194| If (UBound(Labels) = 0) Then Exit Sub
195| Dim flgSkipIt As Integer, ViewBoundsPlus As RealRect
196| 'ViewBoundsPlus will contain a rectangle, ten percent larger on each side,
197| '  than the current ViewPort.  This will allow Labels near the edge to
198| '  be partially shown, even when they don't quite lie in the rectangle.
199| Call MagnifyRectangle(ViewBounds, ViewBoundsPlus, 0#)
200|
201|
202| On Error GoTo ERRPrintLabels1
203|
204|
205| nLabels = LastLabel - FirstLabel + 1
206|
207| frmMap.pnl3dPercentDone.FloodPercent = 0
208| frmMap.lblPercentDone = "PRINTING text labels from " &
```

```
      Themes(nTheme).ThemeName
 209|
 210|  Dim nStart As Long
 211|  For n = FirstLabel To LastLabel
 212|     flgSkipIt = False
 213|    DoEvents
 214|    If flgStopDrawing Then Exit Sub
 215|     nStart = Labels(n).nStart
 216|     Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pt, nStart, lngOne)
 217|  'Check whether the point falls in the viewbounds.
 218|
 219|     If Pt.X < ViewBoundsPlus.Left Then flgSkipIt = True
 220|     If Pt.X > ViewBoundsPlus.Right Then flgSkipIt = True
 221|     If Pt.Y < ViewBoundsPlus.Bottom Then flgSkipIt = True
 222|     If Pt.Y > ViewBoundsPlus.Top Then flgSkipIt = True
 223|     'SetFontCodeInPRINTERxxx Labels(n).FontCode  'New 940712
 224|     Printer.FontSize = Labels(n).FontSize
 225|     If flgSkipIt Then GoTo BotShowLabels
 226|     SetFontInPrinter Labels(n).FontCode
 227|     If flgSkipIt Then GoTo BotShowLabels
 228|
 229|     Printer.ForeColor = Labels(n).ZColor
 230|     DummyShort = SetTextAlign(Printer.hDC, Labels(n).Alignment)
 231|     Printer.FontTransparent = Not Labels(n).Opaque
 232|     If flgSkipIt Then GoTo BotShowLabels
 233|     Printer.CurrentX = Pt.X
 234|     '               If flgSkipIt Then GoTo BotShowPoints2
 235|     Printer.CurrentY = Pt.Y
 236|     '               If flgSkipIt Then GoTo BotShowPoints2
 237|     Printer.Print Trim$(Labels(n).Text)
 238|     '               If flgSkipIt Then GoTo BotShowPoints2
 239|  BotShowLabels:
 240|    frmMap.pnl3dPercentDone.FloodPercent = (n - FirstLabel) / nLabels * 100
 241|  Next n
 242|  frmMap.pnl3dPercentDone.FloodPercent = 100
 243|  frmMap.lblPercentDone = "Done drawing text labels from " &
      Themes(nTheme).ThemeName
 244|  Exit Sub
 245|  ERRPrintLabels1:
 246|  If Err = 6 Then
 247|     flgSkipIt = True
 248|     Resume Next
 249|  End If
 250|  MsgBox "Error #" & Err & " occurred in ShowLabels1." & CRLF & Error
 251|  Err = 0
 252|  Exit Sub
 253|  Resume Next
 254|
 255|
 256|  End Sub
 257|
 258|  '
 259|  Sub PrintLines5 (FirstLine As Long, LastLine As Long, nTheme As Long)
 260|  '940621  This comes full circle.  This is now a descendant of the
```

```
261| ' ShowPolygon5, which is fast.
262| 'GWPouch 940613
263| 'Displays solid color polygons, I hope.
264| 'Derived from ShowLines3.
265|
266| 'This will check to see whether a line overlaps the view rectangle
267| '  before plotting it.
268| Dim n As Long, nLines As Long, nStart As Long, nPoints As Long, nP2 As Long
269| Dim flgSkipToBottom As Integer, iPoints As Integer
270| On Error GoTo ERRShowLines5
271|
272| If (UBound(Lines) = 0) Then Exit Sub
273|
274| nLines = LastLine - FirstLine + 1
275| frmMap.pnl3dPercentDone.FloodPercent = 0
276| frmMap.lblPercentDone = "PRINTING lines from " & Themes(nTheme).ThemeName
277|
278| 'SaveScaling
279|  SaveScalingPrinter OriginalScale
280|  GetPixelBoundsPrinter GDIBounds
281|  Printer.ScaleMode = 3 'Put into pixel mode
282|
283|
284| For n = FirstLine To LastLine
285|  flgSkipToBottom = False
286|  DoEvents
287|  If flgStopPrinting Then GoTo LeaveShowLines5 'Graceful exit that restores scaling
288|  'Test whether the line should be plotted
289|              'XXXX LOGIC PROBLEM.  YOU CAN HAVE A LINE WHOSE BOUNDS
    ARE ABOVE
290|              ' AND BELOW THE CURRENT VIEW WINDOW THAT STILL NEEDS TO
    BE DRAWN.
291|  If Lines(n).MaxX < ViewBounds.Left Then flgSkipToBottom = True
292|  If Lines(n).MinX > ViewBounds.Right Then flgSkipToBottom = True
293|  If Lines(n).MinY > ViewBounds.Top Then flgSkipToBottom = True
294|  If Lines(n).MaxY < ViewBounds.Bottom Then flgSkipToBottom = True
295|  nPoints = Lines(n).nPoints: iPoints = nPoints: nP2 = 3 * nPoints
296|  nStart = Lines(n).nStart
297|  If nPoints < 2 Then flgSkipToBottom = True
298| If flgSkipToBottom = False Then
299|  ReDim LineIn(1 To nPoints) As RealPoint, GDILine(1 To nP2) As APIPoint
300|  Call GetRlPntsFromGlobalMem(hGMemAllPoints, LineIn(1), nStart, nPoints)
301|  'Call ScaleR4To2LOCAL(ViewBounds, GDIBounds, nPoints, LineIn(1), GDILine(1))
302|  'The Printer is mapped to the Printerscale, so that the ViewRect doesn't end up taking
    up the whole sheet.
303|  'Call ScaleR4To2LOCAL(ViewBounds, GDIBounds, nPoints, LineIn(1), GDILine(1))
304|  iPoints = SCScaleR4To2(ViewBounds, GDIBounds, nPoints, LineIn(1), nP2,
    GDILine(1))
305|
306| 'Draw line
307|     If Lines(n).ZWidth = 1 Then
308|        Printer.DrawWidth = Lines(n).ZWidth
309|        Else
310|        Printer.DrawWidth = sngWidthMultiplier * Lines(n).ZWidth
```

```
311|     End If
312|       Printer.ForeColor = Lines(n).ZColor
313|       Printer.DrawStyle = Lines(n).ZStyle
314|       If flgSkipToBottom = False Then ReturnVal = Polyline(Printer.hDC, GDILine(1),
      iPoints)
315|
316| End If
317|
318|   frmMap.pnl3dPercentDone.FloodPercent = (n - FirstLine) / nLines * 100
319| Next n
320|
321| frmMap.pnl3dPercentDone.FloodPercent = 100
322| frmMap.lblPercentDone = "Done PRINTING lines from " &
     Themes(nTheme).ThemeName
323|
324| LeaveShowLines5:
325|  RestoreScalingPrinter OriginalScale
326|  Printer.FillStyle = 1 'Turn off shape filling
327|  Printer.DrawStyle = 0  'Turn on outlining
328| Exit Sub
329|
330|
331| ERRShowLines5:
332| flgSkipToBottom = True
333| Resume Next
334|
335|
336|
337|
338|
339| End Sub
340|
341| Sub PrintOldTiles3 (FirstTile As Long, LastTile As Long, nTheme As Long)
342|
343| ReDim OldTiles(20) As TileDeclPlain
344|
345| Dim OldTilesizeX As Single, OldTilesizeY As Single
346|
347| Dim LLeft As Single, RRight As Single, TTop As Single, BBottom As Single
348| Dim flgSkipPlotting As Integer
349|
350| Dim n As Long, Pt As RealPoint, L As Long, One As Long
351| frmMap.pnl3dPercentDone.FloodPercent = 0
352| frmMap.lblPercentDone = "PRINTING OldTiles from " &
     Themes(nTheme).ThemeName
353| nTiles = LastTile - FirstTile + 1
354|
355| For n = FirstTile To LastTile
356|  DoEvents
357|  If flgStopPrinting Then Exit Sub
358|   L = OldTiles(n).nStart
359|   Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pt, L, lngOne)
360|
361| 'Before trying to plot the tile, check to see whether it falls in
```

```
362|  '  the current viewing rectangle.
363|    LLeft = Pt.X - OldTilesizeX
364|    If LLeft > ViewBounds.Right Then GoTo BottomOldTilesLoop3
365|    RRight = Pt.X + OldTilesizeX
366|    If RRight < ViewBounds.Left Then GoTo BottomOldTilesLoop3
367|    TTop = Pt.Y + OldTilesizeY
368|    If TTop < ViewBounds.Bottom Then GoTo BottomOldTilesLoop3
369|    BBottom = Pt.Y - OldTilesizeY
370|    If BBottom > ViewBounds.Top Then GoTo BottomOldTilesLoop3
371|    'Note that goto s are used rather than a flag variable because this
372|    '  will often let us skip some computations.
373|    Printer.Line (Pt.X - OldTilesizeX, Pt.Y - OldTilesizeY)-(Pt.X + OldTilesizeX, Pt.Y +
     OldTilesizeY), OldTiles(n).ZColor, BF
374|  BottomOldTilesLoop3:
375|    frmMap.pnl3dPercentDone.FloodPercent = (n - FirstTile) / nTiles * 100
376|  Next n
377|
378|  frmMap.pnl3dPercentDone.FloodPercent = 100
379|  frmMap.lblPercentDone = "Done PRINTING OldTiles from " &
     Themes(nTheme).ThemeName
380|
381|
382|  End Sub
383|
384|  '
385|  Sub PrintPoints3 (FirstPoint As Long, LastPoint As Long, nTheme As Long)
386|  Dim n As Long, Pt  As RealPoint
387|  If (UBound(Points) = 0) Then Exit Sub
388|  Dim flgSkipIt As Integer, ViewBoundsPlus As RealRect
389|  'ViewBoundsPlus will contain a rectangle, ten percent larger on each side,
390|  '  than the current ViewPort.  This will allow points near the edge to
391|  '  be partially shown, even when they don't quite lie in the rectangle.
392|  'Call MagnifyRectangle(ViewBounds, ViewBoundsPlus, .1)
393|  Call MagnifyRectangle(ViewBounds, ViewBoundsPlus, 0)
394|
395|  On Error GoTo ERRShowPoints3
396|
397|  DummyShort = SetTextAlign(Printer.hDC, TA_BASELINE)
398|
399|  nPoints = LastPoint - FirstPoint + 1
400|
401|  frmMap.pnl3dPercentDone.FloodPercent = 0
402|  frmMap.lblPercentDone = "PRINTING symbols from " &
     Themes(nTheme).ThemeName
403|  SetFontInPrinter (0)
404|  Dim nStart As Long
405|  For n = FirstPoint To LastPoint
406|     flgSkipIt = False
407|    DoEvents
408|    If flgStopPrinting Then Exit Sub
409|     nStart = Points(n).nStart
410|     Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pt, nStart, lngOne)
411|  'Check whether the point falls in the viewbounds.
412|
```

```
413|   If Pt.X < ViewBoundsPlus.Left Then flgSkipIt = True
414|   If Pt.X > ViewBoundsPlus.Right Then flgSkipIt = True
415|   If Pt.Y < ViewBoundsPlus.Bottom Then flgSkipIt = True
416|   If Pt.Y > ViewBoundsPlus.Top Then flgSkipIt = True
417|   If flgSkipIt Then GoTo BotShowPoints3
418|   Printer.ForeColor = Points(n).ZColor
419|   'frmMap.Print "#" "This line slows program down, but does
420|
421|   '         allow use of color on the Zenith 486 in 401.
422|   Printer.FontSize = Points(n).ZSize
423|   If flgSkipIt Then GoTo BotShowPoints3
424|   SetFontInPrinter (Points(n).ZFontCode)
425|   If flgSkipIt Then GoTo BotShowPoints3
426|   Printer.CurrentX = Pt.X
427|   '               If flgSkipIt Then GoTo BotShowPoints2
428|   Printer.CurrentY = Pt.Y
429|   '               If flgSkipIt Then GoTo BotShowPoints2
430|   Printer.Print Chr$(Points(n).ZSymbol)
431|   '               If flgSkipIt Then GoTo BotShowPoints2
432| BotShowPoints3:
433|   frmMap.pnl3dPercentDone.FloodPercent = (n - FirstPoint) / nPoints * 100
434| Next n
435| frmMap.pnl3dPercentDone.FloodPercent = 100
436| frmMap.lblPercentDone = "Done PRINTING symbols from " &
     Themes(nTheme).ThemeName
437| Exit Sub
438| ERRShowPoints3:
439| Select Case Err
440|
441|  Case 6
442|    flgSkipIt = True
443|    Resume Next
444|  Case Else
445|    Beep
446|    MsgBox "Error #" & Err & " occurred in PrintPoints3." & CRLF & Error
447|    Err = 0
448|    Exit Sub
449|    Resume
450| End Select
451|
452|
453| End Sub
454|
455| Sub PrintPolygons5 (FirstPoly As Long, LastPoly As Long, nTheme As Long)
456| 'GWPouch 940613
457| 'Displays solid color polygons, I hope.
458| 'Derived from ShowLines3.
459|
460| 'This will check to see whether a polygon overlaps the view rectangle
461| '  before plotting it.
462| Dim n As Long, nPolys As Long, nStart As Long, nPoints As Long, nP2 As Long
463| Dim flgSkipToBottom As Integer, iPoints As Integer
464| On Error GoTo ERRShowPolygons5
465|
```

```
466| If (UBound(Polys) = 0) Then Exit Sub
467|
468| nPolys = LastPoly - FirstPoly + 1
469| frmMap.pnl3dPercentDone.FloodPercent = 0
470| frmMap.lblPercentDone = "PRINTING polygons from " &
   Themes(nTheme).ThemeName
471|
472| 'SaveScaling
473|  SaveScalingPrinter OriginalScale
474|  GetPixelBoundsPrinter GDIBounds
475|
476|  Printer.ScaleMode = 3 'Put into pixel mode
477|  SetPolyFillMode Printer.hDC, ALTERNATE 'Set fill mode to allow islands
478|
479| For n = FirstPoly To LastPoly
480|  flgSkipToBottom = False
481|  DoEvents
482|  If flgStopPrinting Then GoTo LeaveShowPolygons5 'Graceful exit that restores scaling
483|  'Test whether the polygon should be plotted
484|                'XXXX LOGIC PROBLEM.  YOU CAN HAVE A LINE WHOSE BOUNDS
   ARE ABOVE
485|                ' AND BELOW THE CURRENT VIEW WINDOW THAT STILL NEEDS TO
   BE DRAWN.
486|  If Polys(n).MaxX < ViewBounds.Left Then flgSkipToBottom = True
487|  If Polys(n).MinX > ViewBounds.Right Then flgSkipToBottom = True
488|  If Polys(n).MinY > ViewBounds.Top Then flgSkipToBottom = True
489|  If Polys(n).MaxY < ViewBounds.Bottom Then flgSkipToBottom = True
490|  nPoints = Polys(n).nPoints: iPoints = nPoints: nP2 = nPoints * 3
491|  nStart = Polys(n).nStart
492|  If nPoints < 3 Then flgSkipToBottom = True
493| If flgSkipToBottom = False Then
494|  ReDim PolyIn(1 To nPoints) As RealPoint, GDIPoly(1 To nP2) As APIPoint
495|  Call GetRIPntsFromGlobalMem(hGMemAllPoints, PolyIn(1), nStart, nPoints)
496|  'Call ScaleR4To2LOCAL(PrintScale, GDIBounds, nPoints, PolyIn(1), GDIPoly(1))
497|  'Call ScaleR4To2LOCAL(ViewBounds, GDIBounds, nPoints, PolyIn(1), GDIPoly(1))
498|  iPoints = SCScaleR4To2(ViewBounds, GDIBounds, nPoints, PolyIn(1), nP2,
   GDIPoly(1))
499|
500| 'Draw optional polygon interior
501|   If Polys(n).FillStyle <> 1 Then
502|       Printer.FillColor = Polys(n).FillColor
503|       Printer.FillStyle = Polys(n).FillStyle
504|       Printer.DrawStyle = 5 'Turn off outlining
505|       If flgSkipToBottom = False Then ReturnVal = Polygon(Printer.hDC, GDIPoly(1),
   iPoints)
506|   End If
507|
508| 'Draw optional outline
509|   If Polys(n).OutlineWidth > 0 Then
510|       If Polys(n).OutlineWidth = 1 Then
511|         Printer.DrawWidth = Polys(n).OutlineWidth
512|        Else
513|         Printer.DrawWidth = sngWidthMultiplier * Polys(n).OutlineWidth
514|       End If
```

```
        Printer.ForeColor = Polys(n).OutlineColor
        Printer.DrawStyle = 6' Inside solid outlining
        If flgSkipToBottom = False Then ReturnVal = Polyline(Printer.hDC, GDIPoly(1),
    iPoints)
      End If

    End If

     frmMap.pnl3dPercentDone.FloodPercent = (n - FirstPoly) / nPolys * 100
    Next n

    frmMap.pnl3dPercentDone.FloodPercent = 100
    frmMap.lblPercentDone = "Done PRINTING polygons from " &
    Themes(nTheme).ThemeName

    LeaveShowPolygons5:
     RestoreScalingPrinter OriginalScale
     Printer.FillStyle = 1 'Turn off shape filling
     Printer.DrawStyle = 0  'Turn on outlining
    Exit Sub


    ERRShowPolygons5:
    flgSkipToBottom = True
    Resume Next

    End Sub

    Sub PrintRasterGrid (myIndex As Long, Junk As Long, nTheme As Long)
    On Error GoTo ERRShowRasterGrid
    Dim nR As Long, nC As Long, PtX As Single, PtY As Single, ColorOut  As Long
    If RasterGrids(myIndex).DataType <> DB_LONG Then Exit Sub
    Printer.DrawWidth = RasterGrids(myIndex).DeltaX / OnePixelInWorldCoordinates() + 1

    For nR = GridRowNumFromYCoord(ViewBounds.Bottom, RasterGrids(myIndex)) To
    GridRowNumFromYCoord(ViewBounds.Top, RasterGrids(myIndex))
      PtY = RasterGrids(myIndex).Y_Origin + nR * RasterGrids(myIndex).DeltaY
      For nC = GridColNumFromXCoord(ViewBounds.Left, RasterGrids(myIndex)) To
    GridColNumFromXCoord(ViewBounds.Right, RasterGrids(myIndex))
        DoEvents
        If flgStopDrawing Then GoTo STOPShowRasterGrid
        PtX = RasterGrids(myIndex).X_Origin + nC * RasterGrids(myIndex).DeltaX
        ColorOut = GetIJCellINT4FromGlobalMem(nC, nR, RasterGrids(myIndex),
    RasterGrids(myIndex).hGMem)
        ' ColorOut = GetXYCellINT4FromGlobalMem(PtX, PtY, RasterGrids(myIndex),
    RasterGrids(myIndex).hGMem)
        If ColorOut <> 0 Then Printer.PSet (PtX, PtY), ColorOut
      Next nC
    Next nR
    STOPShowRasterGrid:
    Printer.DrawWidth = 1
    Exit Sub
    ERRShowRasterGrid:
    Select Case Err
```

```
563|  Case Else
564|  MsgBox "Erro #" & Err & " in ShowRasterGrid." & CRLF
565|  Exit Sub
566|  Resume
567| End Select
568|
569| End Sub
570|
571| Sub PrintThemes (ThemesToPrint() As typTheme, ViewRect As RealRect, PrintRect As
     RealRect, QSDataOut() As typTaggedIDNum, lngQSHighlightColor As Long,
     sngQSHighlightRadius As Single, LastQS As RealPoint)
572|
573| Dim nTh As Long, mm As Long
574| Dim nThemesShownHere As Long
575|   Screen.MousePointer = 11
576| 'ThemesToPrint is an array full of ThemesToPrint to be printed.
577| 'ViewRect is the area, in world coordinates, to be printed
578| 'PrintRect is the location, on the PRINTER, where the map of the area
579| '         ViewRect should show up.
580| 'The units of PrintRect are in MILLIMETERS.
581|
582| Dim PrinterHeight As Single, PrinterWidth As Single
583| Printer.ScaleMode = 6'for millimeters
584| PrinterHeight = Printer.ScaleHeight
585| PrinterWidth = Printer.ScaleWidth
586|
587| 'These ranges leave some space on most printers, to accomodate the
588| ' the rollers and such.  However, these will be used to define the
589| ' the edges.
590|
591| 'Figure out whether the width or height will be the limiting
592| '  factor.
593| Dim XScale As Single, YScale As Single, UsedScale As Single
594| XScale = Abs((PrintRect.Right - PrintRect.Left) / (ViewRect.Right - ViewRect.Left))
595| YScale = Abs((PrintRect.Top - PrintRect.Bottom) / (ViewRect.Top - ViewRect.Bottom))
596| If XScale < YScale Then
597|    UsedScale = XScale
598|  Else
599|    UsedScale = YScale
600| End If
601|
602|
603|
604| 'Set the scale of the printer up so that the corner of the ViewRect map into the corners of
605| ' the PrintRect.
606| PrintScale.Left = ViewRect.Left - PrintRect.Left / UsedScale
607| PrintScale.Top = ViewRect.Top + PrintRect.Top / UsedScale
608| PrintScale.Bottom = PrintScale.Top - Printer.ScaleHeight / UsedScale
609| PrintScale.Right = PrintScale.Left + Printer.ScaleWidth / UsedScale
610|
611| Printer.Scale (PrintScale.Left, PrintScale.Top)-(PrintScale.Right, PrintScale.Bottom)
612|
613| nThemesShownHere = UBound(ThemesToPrint) - LBound(ThemesToPrint) + 1
614| ReDim mThemesToPrint(1 To nThemesShownHere)
```

```
615  'Copy the themes to a module level array, so other subroutines
616  ' can get access to the names properly.
617  For mm = 1 To nThemesShownHere
618     CopyTYPTheme ThemesToPrint(mm), mThemesToPrint(mm)
619  Next mm
620
621
622  LSet ViewBounds = ViewRect
623
624  If flgStopPrinting Then Exit Sub
625
626  For mm = 1 To nThemesShownHere
627     nTh = mm
628
629     DoEvents
630     If flgStopPrinting Then GoTo ExitShowThemes
631  Select Case ThemesToPrint(nTh).ThemeType
632   Case JOINTEDLINE:  Call PrintLines5(ThemesToPrint(nTh).FirstLoc,
     ThemesToPrint(nTh).LastLoc, nTh)
633   Case ONEPOINT:    Call PrintPoints3(ThemesToPrint(nTh).FirstLoc,
     ThemesToPrint(nTh).LastLoc, nTh)
634   Case TILEBLOCK:   Call PrintTiles4(ThemesToPrint(nTh).FirstLoc,
     ThemesToPrint(nTh).LastLoc, nTh)
635   Case HATCHLINE:   Call PrintHatchLines3(ThemesToPrint(nTh).FirstLoc,
     ThemesToPrint(nTh).LastLoc, nTh)
636   Case SOLIDPOLYGON: Call PrintPolygons5(ThemesToPrint(nTh).FirstLoc,
     ThemesToPrint(nTh).LastLoc, nTh)
637   Case LABELTEXT:   Call PrintLabels1(Themes(nTh).FirstLoc, Themes(nTh).LastLoc,
     nTh)
638  End Select
639
640  Next mm
641
642  ExitShowThemes:
643
644  'Display the current spatial selection
645   PrinterCircleQSResults QSDataOut(), lngQSHighlightColor, sngQSHighlightRadius,
     LastQS
646
647
648
649  Printer.Line (ViewRect.Left, ViewRect.Top)-(ViewRect.Right, ViewRect.Bottom), RGB(0,
     0, 0), B
650  Printer.ForeColor = RGB(0, 0, 0)
651
652  Dim Dummy As Integer
653  'Print coordinates, whether the user wanted them or not.
654   Printer.FontSize = 8
655   Printer.FontName = "Arial"
656  'Print bounds of top
657   Dummy = SetTextAlign(Printer.hDC, TA_BOTTOM + TA_CENTER)
658   Printer.CurrentX = (ViewRect.Left + ViewRect.Right) / 2
659   Printer.CurrentY = ViewRect.Top
660   Printer.Print ViewRect.Top
```

```
661| 'Print Bounds of bottom
662|  Dummy = SetTextAlign(Printer.hDC, TA_TOP + TA_CENTER)
663|  Printer.CurrentY = ViewRect.Bottom
664|  Printer.CurrentX = (ViewRect.Left + ViewRect.Right) / 2
665|  Printer.Print ViewRect.Bottom
666| 'Print Bounds of left side
667|  Dummy = SetTextAlign(Printer.hDC, TA_BASELINE + TA_RIGHT)
668|  Printer.CurrentY = (ViewRect.Bottom + ViewRect.Top) / 2
669|  Printer.CurrentX = (ViewRect.Left)
670|  Printer.Print ViewRect.Left
671| 'Print Bounds of left side
672|  Dummy = SetTextAlign(Printer.hDC, TA_BASELINE + TA_LEFT)
673|  Printer.CurrentY = (ViewRect.Bottom + ViewRect.Top) / 2
674|  Printer.CurrentX = (ViewRect.Right)
675|  Printer.Print ViewRect.Right
676|
677|
678|
679|
680|
681| 'If strTitle <> "" Then
682| '  Printer.ForeColor = RGB(0, 0, 0)
683| '  Printer.ScaleMode = 6'Put the printer back into millimeters
684| '  Printer.CurrentX = posTitle.X
685| '  Printer.CurrentY = posTitle.Y
686| '  Printer.FontName = "Times New Roman"
687| '  Printer.FontSize = 18
688| '  Printer.FontBold = True
689| '  Printer.Print strTitle
690| 'End If
691|
692| Printer.EndDoc
693| Screen.MousePointer = 0
694| Beep
695| frmMap.lblPercentDone = "DONE PRINTING"
696| Exit Sub
697|
698|
699|
700| End Sub
701|
702| Sub PrintTiles4 (FirstTile As Long, LastTile As Long, nTheme As Long)
703| 'GWPouch 940720 Converted to use GDI Rectangle, and full outline tiles.
704|
705| Dim flgSkipPlotting As Integer
706| ReDim Pts(1 To 2) As RealPoint, GDIPts(1 To 2) As APIPoint
707| Dim n As Long, L  As Long, One As Long, Two  As Long, dummyInt As Integer
708| Dim GDIBounds As APIRECT
709| One = 1&: Two = 2&
710|
711| frmMap.pnl3dPercentDone.FloodPercent = 0
712| frmMap.lblPercentDone = "PRINTING tiles from " & Themes(nTheme).ThemeName
713| nTiles = LastTile - FirstTile + 1
714|
```

```
 715|   SaveScalingPrinter OriginalScale
 716|   GetPixelBoundsPrinter GDIBounds
 717|   Printer.ScaleMode = 3 'Put into pixel mode
 718|
 719|  For n = FirstTile To LastTile
 720|   flgSkipPlotting = False
 721|   DoEvents
 722|   If flgStopPrinting Then Exit Sub
 723|    L = Tiles(n).nStart
 724|    Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pts(1), L, Two)
 725|
 726|  'Before trying to plot the tile, check to see whether it falls in
 727|  ' the current viewing rectangle.
 728|
 729|    If Pts(1).X > ViewBounds.Right Then flgSkipPlotting = True
 730|    If Pts(2).X < ViewBounds.Left Then flgSkipPlotting = True
 731|    If Pts(1).Y < ViewBounds.Bottom Then flgSkipPlotting = True
 732|    If Pts(2).Y > ViewBounds.Top Then flgSkipPlotting = True
 733|    Printer.FillColor = Tiles(n).InteriorColor
 734|    Printer.FillStyle = Tiles(n).InteriorFillStyle
 735|    Printer.DrawStyle = Tiles(n).OutlineStyle
 736|    Printer.ForeColor = Tiles(n).OutlineColor
 737|    If Tiles(n).OutlineWidth > 0 Then
 738|       If Tiles(n).OutlineWidth = 1 Then
 739|         Printer.DrawWidth = Tiles(n).OutlineWidth
 740|        Else
 741|         Printer.DrawWidth = sngWidthMultiplier * Tiles(n).OutlineWidth
 742|       End If
 743|      Else
 744|        Printer.DrawStyle = 5
 745|    End If
 746|    'Note that goto s are used rather than a flag variable because this
 747|    ' will often let us skip some computations.
 748|    'Printer.Line (Pt.X - TileSizeX, Pt.Y - TileSizeY)-(Pt.X + TileSizeX, Pt.Y + TileSizeY),
      Tiles(n).ZColor, BF
 749|    If Not flgSkipPlotting Then
 750|        Call ScaleR4ToI2LOCAL(ViewBounds, GDIBounds, Two, Pts(1), GDIPts(1))
 751|        dummyInt = Rectangle(Printer.hDC, GDIPts(1).X, GDIPts(1).Y, GDIPts(2).X,
      GDIPts(2).Y)
 752|    End If
 753|
 754|    frmMap.pnl3dPercentDone.FloodPercent = (n - FirstTile) / nTiles * 100
 755|  Next n
 756|   RestoreScalingPrinter OriginalScale
 757|   Printer.FillStyle = 1 'Turn off shape filling
 758|   Printer.DrawStyle = 0  'Turn on outlining
 759|
 760|  frmMap.pnl3dPercentDone.FloodPercent = 100
 761|  frmMap.lblPercentDone = "Done PRINTING tiles from " &
      Themes(nTheme).ThemeName
 762|  Exit Sub
 763|
 764|  ERRPrintTiles4:
 765|  flgSkipPlotting = True
```

```
766|  Resume Next
767|
768|
769|  End Sub
770|
771|  Sub RestorePrinterSettings ()
772|
773|
774|  'Restore the original drawing tools
775|    Printer.DrawStyle = 0  'Solid lines
776|    Printer.DrawMode = 13  'CopyPen
777|    Printer.DrawWidth = 1  'Thin line
778|    Printer.FillStyle = 1  'Transparent fill
779|    Printer.ForeColor = 0&
780|
781|    ReturnVal = SetTextAlign(Printer.hDC, 0) 'TA_TOP + TA_LEFT)
782|  End Sub
783|
784|  Sub RestoreScalingPrinter (RestoreFrom As RealRect)
785|  Printer.ScaleMode = 0
786|    Printer.ScaleLeft = RestoreFrom.Left
787|    Printer.ScaleTop = RestoreFrom.Top
788|    Printer.ScaleWidth = RestoreFrom.Right
789|    Printer.ScaleHeight = RestoreFrom.Bottom
790|
791|  End Sub
792|
793|  Sub SaveScalingPrinter (SaveTo As RealRect)
794|  'GWPouch 940613
795|   SaveTo.Left = Printer.ScaleLeft
796|   SaveTo.Top = Printer.ScaleTop
797|   SaveTo.Right = Printer.ScaleWidth
798|   SaveTo.Bottom = Printer.ScaleHeight
799|
800|  End Sub
801|
802|  Sub SetPrinterViewRect (ViewRect As RealRect)
803|   ViewBounds = ViewRect
804|  End Sub
805|
806|  Sub SetPrintWidthMultiplier (ByVal a As Single)
807|  If a > 1 Then
808|    sngWidthMultiplier = a
809|   Else
810|    sngWidthMultiplier = 5
811|  End If
812|
813|
814|  End Sub
815|
816|  Sub ZPrinterBeachBall (X As Single, Y As Single)
817|
818|        On Error Resume Next
819|        Dim DDrawMode As Integer, DDrawStyle As Integer, PhilColor As Long, PhilStyle
```

```
      As Integer
 820|    'Save current drawing setup
 821|       PhilColor = Printer.FillColor
 822|       PhilStyle = Printer.FillStyle
 823|       DDrawMode = Printer.DrawMode
 824|       DDrawStyle = Printer.DrawStyle
 825|    'Set for  solid color drawing
 826|       Printer.DrawStyle = 0
 827|       'Printer.DrawMode = 7
 828|       Printer.FillStyle = 0
 829|       'Printer.FillColor = RGB(0, 0, 0) 'For printer, draw this in black
 830|    'Make a beachball
 831|       Printer.Circle (X, Y), (ViewBounds.Right - ViewBounds.Left) * .025, 0&,
     -.00000000001, -1.570796326795
 832|       Printer.Circle (X, Y), (ViewBounds.Right - ViewBounds.Left) * .025, 0&,
     -3.14159265359, -4.712388980385
 833|    'restore previous drawing setup
 834|       Printer.FillColor = PhilColor
 835|       Printer.FillStyle = PhilStyle
 836|       Printer.DrawMode = DDrawMode
 837|       Printer.DrawStyle = DDrawStyle
 838|
 839| End Sub
 840|
```

```
 1| Option Explicit
 2| Dim flgLoadDIB As Long, flgSkipRepainting As Long, hMF As Integer, hDCMeta As
    Integer, ReturnVal As Long
 3| Dim ViewBounds As RealRect, GDIBounds As APIRECT
 4| Dim OriginalScale As RealRect, PrintScale As RealRect
 5| Dim ScaleFactor As Single, DummyLong As Long
 6|
 7|
 8|
 9| Dim hFont As Integer, hOldFont As Integer   ', Ascii0 As String * 1   'Ascii0 = Chr$(0)
10| Dim Ascii0 As String * 1
11| 'LOGFONT
12| Global Const LF_FACESIZE = 32
13| Type LOGFONT
14|    lfHeight As Integer
15|    lfWidth As Integer
16|    lfEscapement As Integer
17|    lfOrientation As Integer
18|    lfWeight As Integer
19|    lfItalic As String * 1
20|    lfUnderline As String * 1
21|    lfStrikeOut As String * 1
22|    lfCharSet As String * 1
23|    lfOutPrecision As String * 1
24|    lfClipPrecision As String * 1
25|    lfQuality As String * 1
26|    lfPitchAndFamily As String * 1
27|    lfFaceName As String * LF_FACESIZE
28| End Type
29| Dim MFFont As LOGFONT
30| '  CreateFontIndirect
31| Declare Function CreateFontIndirect Lib "GDI" (lpLogFont As LOGFONT) As Integer
32| Global Const FW_BOLD = 700
33| Global Const FW_NORMAL = 400
34| Global Const OUT_TT_PRECIS = 4
35| Global Const CLIP_TT_ALWAYS = &H20
36| Global Const DRAFT_QUALITY = 1
37| Declare Function DeleteObject Lib "GDI" (ByVal hObject As Integer) As Integer
38| Declare Function SelectObject Lib "GDI" (ByVal hDC As Integer, ByVal hObject As
    Integer) As Integer
39| Declare Function CreateMetaFile Lib "GDI" (lpString As Any) As Integer
40| Declare Function CreatePen Lib "GDI" (ByVal nPenStyle As Integer, ByVal nWidth As
    Integer, ByVal crColor As Long) As Integer
41| Declare Function CloseMetaFile Lib "GDI" (ByVal hMF As Integer) As Integer
42|
43| Type LOGBRUSH
44|    lbStyle As Integer
45|    lbColor As Long
46|    lbHatch As Integer
47| End Type
48| Dim LogicalBrush As LOGBRUSH
49| Declare Function CreateBrushIndirect Lib "GDI" (lpLogBrush As LOGBRUSH) As Integer
50|
51| Global Const BS_HATCHED = 2
```

```
52| Global Const BS_SOLID = 0
53| Declare Function CreateHatchBrush Lib "GDI" (ByVal nIndex As Integer, ByVal crColor
    As Long) As Integer
54| Declare Function SetWindowExt Lib "GDI" (ByVal hDC As Integer, ByVal X As Integer,
    ByVal Y As Integer) As Long
55| Declare Function SetWindowOrg Lib "GDI" (ByVal hDC As Integer, ByVal X As Integer,
    ByVal Y As Integer) As Long
56| Declare Function SetMapMode Lib "GDI" (ByVal hDC As Integer, ByVal nMapMode As
    Integer) As Integer
57| Global Const MM_ISOTROPIC = 7
58| Global Const MM_ANISOTROPIC = 8
59| Global Const MM_HIMETRIC = 3
60|
61|
62| Declare Function Ellipse Lib "GDI" (ByVal hDC As Integer, ByVal X1 As Integer, ByVal
    Y1 As Integer, ByVal X2 As Integer, ByVal Y2 As Integer) As Integer
63| Declare Function Arc Lib "GDI" (ByVal hDC As Integer, ByVal X1 As Integer, ByVal Y1
    As Integer, ByVal X2 As Integer, ByVal Y2 As Integer, ByVal X3 As Integer, ByVal Y3 As
    Integer, ByVal X4 As Integer, ByVal Y4 As Integer) As Integer
64| Declare Function Chord Lib "GDI" (ByVal hDC As Integer, ByVal X1 As Integer, ByVal Y1
    As Integer, ByVal X2 As Integer, ByVal Y2 As Integer, ByVal X3 As Integer, ByVal Y3 As
    Integer, ByVal X4 As Integer, ByVal Y4 As Integer) As Integer
65|
66| Declare Function GetStockObject Lib "GDI" (ByVal nIndex As Integer) As Integer
67| Global Const NULL_BRUSH = 5
68| Global Const NULL_PEN = 8
69|
70| Declare Function SetTextColor Lib "GDI" (ByVal hDC As Integer, ByVal crColor As Long)
    As Long
71| Declare Function TextOut Lib "GDI" (ByVal hDC As Integer, ByVal X As Integer, ByVal Y
    As Integer, ByVal lpString As String, ByVal nCount As Integer) As Integer
72|
73| Type METAFILEHEADER
74|  key As Long
75|  HANDLE As Integer
76| '  RECT   bbox;
77|   BoundsLeft As Integer
78|   BoundsTop As Integer
79|   BoundsRight As Integer
80|   BoundsBottom As Integer
81|   inchPerGDIUnit As Integer
82|   reserved As Long
83|   checksum As Integer
84| End Type
85| Type MetaFileHeaderAsBinary
86|  bin(0 To 11) As Integer
87| End Type
88|
89|
90| 'For copying the metafile to the clipboard using the Windows API
91| Declare Function OpenClipboard Lib "User" (ByVal hWnd As Integer) As Integer
92| Declare Function EmptyClipboard Lib "User" () As Integer
93| Declare Function SetClipboardData Lib "User" (ByVal wFormat As Integer, ByVal hMem
    As Integer) As Integer
```

```
Declare Function CloseClipboard Lib "User" () As Integer
Type METAFILEPICT
   mm As Integer
   xExt As Integer
   yExt As Integer
   hMF As Integer
End Type

Sub CopyWMFToClipboard (hMF As Integer)
Dim myMetafilePict As METAFILEPICT
Dim Dummy As Long, hGMemMetaPict As Integer, hSel As Integer, nOut As Long

Dummy = OpenClipboard(MDI_Wheat.hWnd)
If Dummy = 0 Then
  Beep
  MsgBox "Cannot copy data to clipboard"
  Exit Sub
End If


   Dummy = EmptyClipboard()

   myMetafilePict.mm = MM_ANISOTROPIC'  Global Const MM_HIMETRIC = 3
""

     "'Global const MM_ISOTROPIC=7,  Might want to use MM_ISOTROPIC
   myMetafilePict.xExt = (GDIBounds.Right - GDIBounds.Left) '* 1.05
   myMetafilePict.yExt = (GDIBounds.Top - GDIBounds.Bottom) '* 1.05' GDIBounds.Top
- GDIBounds.Bottom
   myMetafilePict.hMF = hMF
 hGMemMetaPict = GlobalAlloc(GHND, 8)
 hSel = GetSelector(hGMemMetaPict)
 Dummy = MemoryWrite(hSel, 0, myMetafilePict, 8)
 Dummy = SetClipboardData(3, hGMemMetaPict)  'Global Const CF_METAFILEPICT =
3
 Dummy = CloseClipboard()




End Sub

Function FindX_WMF (X As Single) As Integer
  FindX_WMF = GDIBounds.Left + (X - ViewBounds.Left) * ScaleFactor
End Function

Function FindY_WMF (Y As Single) As Integer
  FindY_WMF = GDIBounds.Bottom + (Y - ViewBounds.Bottom) * ScaleFactor
End Function

Function GetHDCMeta () As Integer
  GetHDCMeta = hDCMeta
End Function
```

```
145|
146| Sub GetMetafileGDIBounds (rctCopy As APIRECT)
147|   rctCopy = GDIBounds
148| End Sub
149|
150| Sub GetMetaFileWorldBounds (rctCopy As RealRect)
151|   rctCopy = ViewBounds
152| End Sub
153|
154| Sub SetBrushInWMF (FillStyle As Integer, FillColor As Long)
155| Static LastFillColor  As Long, LastFillStyle As Long, hBrush As Integer, hOldBrush As
    Integer
156|    'Leave if the brush is already right.
157|     If FillStyle = LastFillStyle And FillColor = LastFillColor Then Exit Sub
158|
159| LastFillStyle = FillStyle: LastFillColor = FillColor
160|
161|   Select Case LastFillStyle
162|     Case 0 'Solid fill
163|   LogicalBrush.lbStyle = 0  'BS_SOLID
164|     Case 1 'Null fill
165|   LogicalBrush.lbStyle = 1  'BS_NULL
166|     Case Else 'Hatch fill of some sort
167|   LogicalBrush.lbStyle = 2  'BS_HATCHED
168|   LogicalBrush.lbHatch = LastFillStyle - 2
169|   End Select
170|   LogicalBrush.lbColor = LastFillColor
171|   hBrush = CreateBrushIndirect(LogicalBrush)
172|   hOldBrush = SelectObject(hDCMeta, hBrush)
173|   hOldBrush = DeleteObject(hOldBrush)
174|
175| End Sub
176|
177| Sub SetFontInWMF (ZSize As Single, ZFontCode As Integer)
178| Static LastSize As Single, LastFontCode As Integer, nFontsTotal As Long
179|  Dim nChar As Integer
180| On Error GoTo ERRSetFontInWMF
181| 'Do nothing if the fontsize and fontcode are the same as last time.
182| If ZSize = LastSize And ZFontCode = LastFontCode Then Exit Sub
183| LastSize = ZSize: LastFontCode = ZFontCode
184| nFontsTotal = nFontsTotal + 1
185|
186| Static FontNameUsed As String, LastFontName As String, nF As Long, tmpCode As
    Integer
187|
188|
189| 'FONTCODE encodes Bold and Italic in the first two bits,
190| '  and a font ID # used in tblFonts to encode the font name
191| '  in the other 14 bits.
192|    If AND2(LastFontCode, BOLDTYPE) Then
193|   MFFont.lfWeight = FW_BOLD
194|      Else
195|   MFFont.lfWeight = FW_NORMAL
196|    End If
```

```
197|
198|    If AND2(LastFontCode, ITALICTYPE) Then
199|   MFFont.lfItalic = Chr$(255)
200|     Else
201|   MFFont.lfItalic = Chr$(0)
202|    End If
203|
204|
205|  tmpCode = MASK14LOBITS And LastFontCode
206|
207|  FontNameUsed = "Arial" 'Set default font to Arial
208|  For nF = 1 To UBound(FontTable)
209|
210|     If FontTable(nF).Number = tmpCode Then
211|   FontNameUsed = FontTable(nF).Name
212|   Exit For
213|     End If
214|  Next nF
215|  MFFont.lfFaceName = FontNameUsed
216|  'sStop
217|  nChar = 32
218|  Do Until Asc(Mid$(MFFont.lfFaceName, nChar, 1)) <> 32
219|    Mid$(MFFont.lfFaceName, nChar, 1) = Ascii0
220|    nChar = nChar - 1
221|  Loop
222|           'Experimental 940912
223|  MFFont.lfHeight = -LastSize * 34.0277 / 2.04'Size in hundredths of millimeters, for use
     with LO_METRIC mapping mode.
224| """dddDebug.Print nFontsTotal, LastSize, ZFontCode, FontNameUsed, Now
225|  hFont = CreateFontIndirect(MFFont)
226|  hOldFont = SelectObject(hDCMeta, hFont)
227|  hOldFont = DeleteObject(hOldFont)
228| Exit Sub
229|
230|
231| ERRSetFontInWMF:
232|  Select Case Err
233|   Case Else
234|    MsgBox "Error #" & Err & " occurred in SetFontInWMF." & CRLF & Error
235|    Err = 0
236|    Exit Sub
237|    Resume Next
238|  End Select
239| End Sub
240|
241| Sub SetPenInWMF (LineColor As Long, LineWidth As Integer, LineStyle As Integer)
242| Static LastColor As Long, LastWidth As Integer, LastStyle As Integer
243| Static hPen As Integer, hOldPen As Integer
244|
245| If (LastColor = LineColor) And (LastWidth = LineWidth) And (LastStyle = LineStyle) Then
     Exit Sub
246|    LastColor = LineColor: LastWidth = LineWidth: LastStyle = LineStyle
247|    If LastWidth <> 0 Then
248|   hPen = CreatePen(LastStyle, LastWidth, LastColor)
```

```
249|       Else
250|    hPen = CreatePen(5, 1, 0&)
251|     End If
252|     hOldPen = SelectObject(hDCMeta, hPen)
253|     hOldPen = DeleteObject(hOldPen)
254| End Sub
255|
256| Sub SetTextColorInWMF (Color As Long)
257|    DummyLong = SetTextColor(hDCMeta, Color)
258| End Sub
259|
260| Sub StartWMF (ViewRect As RealRect, SizeRect As APIRECT, Where As String)
261|    ViewBounds = ViewRect  'ViewBounds contains the RealWorld coords of the view
    region
262|     GDIBounds = SizeRect   'SizeRect contains the output dimensions
263|           '  of the metafile in 100ths of a millimeter
264|   If Where = "" Then
265|     hDCMeta = CreateMetaFile(ByVal 0&)
266|    Else
267|     hDCMeta = CreateMetaFile(ByVal Where)
268|   End If
269|  Dim Junk As Long
270|  If GDIBounds.Left = GDIBounds.Right Or GDIBounds.Top = GDIBounds.Bottom Then
271|    GDIBounds.Top = 10000: GDIBounds.Bottom = -10000'Force this to be about a 8" tall
272|    GDIBounds.Left = -10000: GDIBounds.Right = -10000 + 20000 * (ViewBounds.Top -
    ViewBounds.Bottom) / (ViewBounds.Right - ViewBounds.Left)
273|  End If
274|
275|   ScaleFactor = (GDIBounds.Top - GDIBounds.Bottom) / (ViewBounds.Top -
    ViewBounds.Bottom)
276|
277|  'Initialize the metafile for isotropic output
278| Junk = SetWindowExt(hDCMeta, GDIBounds.Top - GDIBounds.Bottom, GDIBounds.Left
    - GDIBounds.Right)
279| 'Junk = SetWindowOrg(hDCMeta, GDIBounds.Bottom, GDIBounds.Left) Original
280|   'Junk = SetMapMode(hDCMeta, MM_ISOTROPIC) Original form
281| 'Junk = SetWindowOrg(hDCMeta, GDIBounds.Top, GDIBounds.Left)
282|    Junk = SetMapMode(hDCMeta, MM_ISOTROPIC)
283|
284|
285|    Ascii0 = Chr$(0)
286|    MFFont.lfWidth = 0
287|    MFFont.lfEscapement = 0
288|    MFFont.lfOrientation = 0
289|    MFFont.lfUnderline = Ascii0
290|    MFFont.lfStrikeOut = Ascii0
291|    MFFont.lfCharSet = Ascii0
292|    MFFont.lfOutPrecision = Chr$(OUT_TT_PRECIS)
293|    MFFont.lfClipPrecision = Chr$(CLIP_TT_ALWAYS)
294|    MFFont.lfQuality = Chr$(DRAFT_QUALITY)
295|    MFFont.lfPitchAndFamily = Chr$(4)
296|
297|
298|    'Re-initialize the pen, font, and brush to unlikely values, so there
```

```
299|    ' is always some sort of GDI object available
300|    SetFontInWMF 2, 0
301|    SetPenInWMF 0&, 0, 5
302|    SetBrushInWMF 1, 0&
303| End Sub
304|
305| Sub StopWMF (hMFBack As Integer)
306|    hMFBack = DeleteObject(hFont)
307|    Dim hPen As Integer
308|    hPen = SelectObject(hDCMeta, GetStockObject(NULL_PEN))
309|    hPen = DeleteObject(hPen)
310|    hPen = SelectObject(hDCMeta, GetStockObject(NULL_BRUSH))
311|    hPen = DeleteObject(hPen)
312|    hPen = Rectangle(hDCMeta, GDIBounds.Left, GDIBounds.Top, GDIBounds.Right,
     GDIBounds.Bottom)
313|
314|    hMFBack = CloseMetaFile(hDCMeta)
315|    hDCMeta = 0
316| End Sub
317|
318| Sub WMFDrawFilledCircle (XCenter As Single, YCenter As Single, Radius As Single,
     Color As Long)
319|  SetPenInWMF Color, 1, 0
320|  SetBrushInWMF 0, Color
321|
322|   Dim XL As Integer, XR As Integer, YT As Integer, YB As Integer
323|      XR = FindX_WMF(XCenter + Radius)
324|      XL = FindX_WMF(XCenter - Radius)
325|      YT = FindY_WMF(YCenter + Radius)
326|      YB = FindY_WMF(YCenter - Radius)
327|
328|   DummyLong = Ellipse(hDCMeta, XL, YT, XR, YB)
329|
330| End Sub
331|
332| Sub WMFDrawGlobalHatchLine (hGMemAllPoints As Integer, nStart As Long, nPoints
     As Long)
333|  If nPoints = 0 Then Exit Sub
334| On Error Resume Next
335|  ReDim RlPts(1 To nPoints) As RealPoint
336|  Dim n  As Long
337|     Call GetRlPntsFromGlobalMem(hGMemAllPoints, RlPts(1), nStart, nPoints)
338|     For n = 1 To (nPoints - 1) Step 2
339|    Printer.Line (RlPts(n).X, RlPts(n).Y)-(RlPts(n + 1).X, RlPts(n + 1).Y)
340|     Next n
341| End Sub
342|
343| Sub WMFDrawHollowCircle (XCenter As Single, YCenter As Single, Radius As Single,
     Color As Long)
344|  SetPenInWMF Color, 1, 0
345|  SetBrushInWMF 1, Color
346|
347|   Dim XL As Integer, XR As Integer, YT As Integer, YB As Integer
348|      XR = FindX_WMF(XCenter + Radius)
```

```
349|       XL = FindX_WMF(XCenter - Radius)
350|       YT = FindY_WMF(YCenter + Radius)
351|       YB = FindY_WMF(YCenter - Radius)
352|
353|    DummyLong = Ellipse(hDCMeta, XL, YT, XR, YB)
354|  End Sub
355|
356|  '
357|  Sub WMFHatchLines3 (FirstLine As Long, LastLine As Long, nTheme As Long) '
358|  'GWPouch 940531  The ShowTYPE3 series of subroutines checks to see
359|  '  if the desired feature lies inside, or close to the viewport
360|  '  before trying to draw it.  This should avoid many overflow errors
361|  '  and the like.
362|  On Error GoTo ERRShowHatchLines3
363|
364|  Dim n As Long, nLines As Long, flgSkipIt As Integer
365|  If (UBound(HatchLines) = LBound(HatchLines)) Then Exit Sub
366|  nLines = LastLine - FirstLine + 1
367|  frmMap.pnl3dPercentDone.FloodPercent = 0
368|  frmMap.lblPercentDone = "Printing hatch lines from " & Themes(nTheme).ThemeName
369|
370|  For n = FirstLine To LastLine
371|   DoEvents
372|   If flgStopPrinting Then Exit Sub
373|   flgSkipIt = False
374|   ' Check whether the polygon, or whatever, of which this is the hatching
375|   '  lies outside the current viewport.
376|   If HatchLines(n).MaxX < ViewBounds.Left Then flgSkipIt = True
377|   If HatchLines(n).MinX > ViewBounds.Right Then flgSkipIt = True
378|   If HatchLines(n).MinY > ViewBounds.Top Then flgSkipIt = True
379|   If HatchLines(n).MaxY < ViewBounds.Bottom Then flgSkipIt = True
380|
381|     Printer.DrawWidth = HatchLines(n).ZWidth
382|     Printer.ForeColor = HatchLines(n).ZColor
383|     Printer.DrawStyle = HatchLines(n).ZStyle
384|
385|     If flgSkipIt = False Then
386|   Call PrintDrawGlobalHatchLine(hGMemAllPoints, HatchLines(n).nStart,
      HatchLines(n).nPoints)
387|     End If
388|
389|     frmMap.pnl3dPercentDone.FloodPercent = (n - FirstLine) / nLines * 100
390|  Next n
391|
392|  frmMap.pnl3dPercentDone.FloodPercent = 100
393|  frmMap.lblPercentDone = "Done PRINTING hatch lines from " &
      Themes(nTheme).ThemeName
394|  Exit Sub
395|
396|  ERRShowHatchLines3:
397|  flgSkipIt = True
398|  Resume Next
399|
400|
```

```
401|
402| End Sub
403|
404| Sub WMFLabels1 (FirstLabel As Long, LastLabel As Long, nTheme As Long)
405| Dim n As Long, Pt  As RealPoint
406| If (UBound(Labels) = LBound(Labels)) Then Exit Sub
407| Dim flgSkipIt As Integer, ViewBoundsPlus As RealRect
408| 'ViewBoundsPlus will contain a rectangle, ten percent larger on each side,
409| '  than the current ViewPort.  This will allow Labels near the edge to
410| '  be partially shown, even when they don't quite lie in the rectangle.
411| Call MagnifyRectangle(ViewBounds, ViewBoundsPlus, 0#)
412|
413|
414| On Error GoTo ERRPrintLabels1
415|
416|
417| nLabels = LastLabel - FirstLabel + 1
418| Dim nStart As Long, nP2 As Long, strOut As String, GDIPt As APIPoint
419| For n = FirstLabel To LastLabel
420|    flgSkipIt = False
421|   DoEvents
422|
423|    nStart = Labels(n).nStart
424|    Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pt, nStart, lngOne)
425| 'Check whether the point falls in the viewbounds.
426|
427|    If Pt.X < ViewBoundsPlus.Left Then flgSkipIt = True
428|    If Pt.X > ViewBoundsPlus.Right Then flgSkipIt = True
429|    If Pt.Y < ViewBoundsPlus.Bottom Then flgSkipIt = True
430|    If Pt.Y > ViewBoundsPlus.Top Then flgSkipIt = True
431|    If flgSkipIt Then GoTo BotShowLabels
432|    SetFontInWMF Labels(n).FontSize, Labels(n).FontCode
433|    nP2 = SetTextColor(hDCMeta, Labels(n).ZColor)
434|    DummyShort = SetTextAlign(hDCMeta, Labels(n).Alignment)
435|
436|    If flgSkipIt Then GoTo BotShowLabels
437|     GDIPt.X = GDIBounds.Left + (Pt.X - ViewBounds.Left) * ScaleFactor
438|     GDIPt.Y = GDIBounds.Bottom + (Pt.Y - ViewBounds.Bottom) * ScaleFactor
439|     strOut = Trim$(Labels(n).Text)
440|     nP2 = TextOut(hDCMeta, GDIPt.X, GDIPt.Y, ByVal strOut, Len(strOut))
441| BotShowLabels:
442|
443| Next n
444| Exit Sub
445| ERRPrintLabels1:
446| Select Case Err
447|   Case 6
448|     flgSkipIt = True
449|     Resume Next
450|   Case Else
451|     MsgBox "Error #" & Err & " occurred in ShowLabels1." & CRLF & Error
452|     Err = 0
453|     Exit Sub
454|     Resume Next
```

```
End Select

End Sub

Sub WMFLines5 (FirstLine As Long, LastLine As Long, nTheme As Long)
'940621  This comes full circle.  This is now a descendant of the
' ShowPolygon5, which is fast.
'GWPouch 940613
'Displays solid color polygons, I hope.
'Derived from ShowLines3.

'This will check to see whether a line overlaps the view rectangle
'  before plotting it.
Dim n As Long, nLines As Long, nStart As Long, nPoints As Long, nP2 As Long
Dim flgSkipToBottom As Integer, iPoints As Integer
On Error GoTo ERRShowLines5

If (UBound(Lines) = 0) Then Exit Sub
nLines = LastLine - FirstLine + 1
Static hOldPen As Integer, hPen As Integer
Static LastColor As Long, LastWidth As Long, LastStyle As Long

For n = FirstLine To LastLine
 flgSkipToBottom = False
 DoEvents
 If Lines(n).MaxX < ViewBounds.Left Then flgSkipToBottom = True
 If Lines(n).MinX > ViewBounds.Right Then flgSkipToBottom = True
 If Lines(n).MinY > ViewBounds.Top Then flgSkipToBottom = True
 If Lines(n).MaxY < ViewBounds.Bottom Then flgSkipToBottom = True
 nPoints = Lines(n).nPoints: iPoints = nPoints: nP2 = 3 * nPoints
 nStart = Lines(n).nStart
 If nPoints < 2 Then flgSkipToBottom = True
If flgSkipToBottom = False Then
 ReDim LineIn(1 To nPoints) As RealPoint, GDILine(1 To nP2) As APIPoint
 Call GetRlPntsFromGlobalMem(hGMemAllPoints, LineIn(1), nStart, nPoints)
 iPoints = SCScaleR4To2(ViewBounds, GDIBounds, nPoints, LineIn(1), nP2,
GDILine(1))

'Draw line
 'If (LastColor <> Lines(n).ZColor) Or (LastWidth <> Lines(n).ZWidth) Or (LastStyle <>
Lines(n).ZStyle) Then
 '  LastColor = Lines(n).ZColor
 '  LastWidth = Lines(n).ZWidth
 '  LastStyle = Lines(n).ZStyle
 '  hPen = DeleteObject(hPen)
 '  hPen = CreatePen(LastStyle, LastWidth, LastColor)' VBColor2GDIColor(LastColor))
 '  hOldPen = SelectObject(hDCMeta, hPen)
 ' 'hOldPen = DeleteObject(hOldPen)
 'End If
   If flgSkipToBottom = False Then
  SetPenInWMF Lines(n).ZColor, Lines(n).ZWidth, Lines(n).ZStyle
  ReturnVal = Polyline(hDCMeta, GDILine(1), iPoints)
  If ReturnVal = 0 Then Beep
   End If
```

```
507|
508| End If
509| Next n
510|
511| LeaveShowLines5:
512| hPen = DeleteObject(hPen)
513| Exit Sub
514|
515|
516| ERRShowLines5:
517| flgSkipToBottom = True
518| Resume Next
519|
520|
521|
522|
523|
524| End Sub
525|
526| Sub WMFPlotSymbol (X As Single, Y As Single, FontCode As Integer, FontSize As
     Single, SymbolCode As Integer)
527|   Dim iX As Integer, jY As Integer, strOut As String
528|
529|
530|   SetFontInWMF FontSize, FontCode
531|   iX = FindX_WMF(X)
532|   jY = FindY_WMF(Y)
533|   strOut = Chr$(SymbolCode)
534|   iX = TextOut(hDCMeta, iX, jY, ByVal strOut, 1)
535|
536| End Sub
537|
538| Sub WMFPoints3 (FirstPoint As Long, LastPoint As Long, nTheme As Long)
539| Dim n As Long, Pt  As RealPoint
540| If (UBound(Points) = 0) Then Exit Sub
541| Dim flgSkipIt As Integer, ViewBoundsPlus As RealRect, strOut As String
542| Dim nStart As Long, nPoints As Long, nP2 As Long, iPoints  As Long
543| Dim GDIPt As APIPoint
544| 'ViewBoundsPlus will contain a rectangle, ten percent larger on each side,
545| '  than the current ViewPort.  This will allow points near the edge to
546| '  be partially shown, even when they don't quite lie in the rectangle.
547| 'Call MagnifyRectangle(ViewBounds, ViewBoundsPlus, .1)
548| Call MagnifyRectangle(ViewBounds, ViewBoundsPlus, 0)
549|
550| On Error GoTo ERRShowPoints3
551|
552| DummyShort = SetTextAlign(hDCMeta, TA_BASELINE)
553|
554| nPoints = LastPoint - FirstPoint + 1
555| nStart = Points(FirstPoint).nStart
556| "Knock current font out of Metafile
557| 'SetFontInWMF 2, 0
558|
559| For n = FirstPoint To LastPoint
```

```
560|     flgSkipIt = False
561|   DoEvents
562|   If flgStopPrinting Then Exit Sub
563|
564|
565|  'Check whether the point falls in the viewbounds.
566|     nStart = Points(n).nStart
567|     Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pt, nStart, lngOne)
568|    If Pt.X < ViewBoundsPlus.Left Then flgSkipIt = True
569|    If Pt.X > ViewBoundsPlus.Right Then flgSkipIt = True
570|    If Pt.Y < ViewBoundsPlus.Bottom Then flgSkipIt = True
571|    If Pt.Y > ViewBoundsPlus.Top Then flgSkipIt = True
572|    If flgSkipIt Then GoTo BotShowPoints3
573|     GDIPt.X = GDIBounds.Left + (Pt.X - ViewBounds.Left) * ScaleFactor' /
      (ViewBounds.Right - ViewBounds.Left) * (GDIBounds.Right - GDIBounds.Left)
574|     GDIPt.Y = GDIBounds.Bottom + (Pt.Y - ViewBounds.Bottom) * ScaleFactor'/
      (ViewBounds.Top - ViewBounds.Bottom) * (GDIBounds.Top - GDIBounds.Bottom)
575|    SetFontInWMF Points(n).ZSize, Points(n).ZFontCode
576|     'ddDebug.Print n, GDIPt.X, GDIPt.Y
577|    nP2 = SetTextColor(hDCMeta, Points(n).ZColor)
578|
579|    If flgSkipIt Then GoTo BotShowPoints3
580|     strOut = Chr$(Points(n).ZSymbol)
581|     nP2 = TextOut(hDCMeta, GDIPt.X, GDIPt.Y, ByVal strOut, 1)
582|  BotShowPoints3:
583|
584|  Next n
585|
586|  Exit Sub
587|  ERRShowPoints3:
588|  Select Case Err
589|
590|   Case 6
591|     flgSkipIt = True
592|     Resume Next
593|   Case Else
594|     Beep
595|     MsgBox "Error #" & Err & " occurred in PrintPoints3." & CRLF & Error
596|     Err = 0
597|     Exit Sub
598|     Resume
599|  End Select
600|
601|
602|  End Sub
603|
604|  Sub WMFPolygons5 (FirstPoly As Long, LastPoly As Long, nTheme As Long)
605|  'GWPouch 940613
606|  'Displays solid color polygons, I hope.
607|  'Derived from ShowLines3.
608|
609|  'This will check to see whether a polygon overlaps the view rectangle
610|  '  before plotting it.
611|  Dim n As Long, nPolys As Long, nStart As Long, nPoints As Long, nP2 As Long
```

```
612  Dim flgSkipToBottom As Integer, iPoints As Integer
613  On Error GoTo ERRShowPolygons5
614
615  Dim hBrush As Integer, hOldBrush As Integer, hPen As Integer, hOldPen As Integer
616
617  Dim LastLineWidth As Integer, LastLineColor As Long, LastLineStyle As Integer
618
619
620  If (UBound(Polys) = 0) Then Exit Sub
621
622  nPolys = LastPoly - FirstPoly + 1
623   SetPolyFillMode hDCMeta, ALTERNATE 'Set fill mode to allow islands
624
625  For n = FirstPoly To LastPoly
626   flgSkipToBottom = False
627   DoEvents
628
629
630   If Polys(n).MaxX < ViewBounds.Left Then flgSkipToBottom = True
631   If Polys(n).MinX > ViewBounds.Right Then flgSkipToBottom = True
632   If Polys(n).MinY > ViewBounds.Top Then flgSkipToBottom = True
633   If Polys(n).MaxY < ViewBounds.Bottom Then flgSkipToBottom = True
634   nPoints = Polys(n).nPoints: iPoints = nPoints: nP2 = nPoints * 3
635   nStart = Polys(n).nStart
636   If nPoints < 3 Then flgSkipToBottom = True
637  If flgSkipToBottom = False Then
638   ReDim PolyIn(1 To nPoints) As RealPoint, GDIPoly(1 To nP2) As APIPoint
639   Call GetRlPntsFromGlobalMem(hGMemAllPoints, PolyIn(1), nStart, nPoints)
640   iPoints = SCScaleR4To12(ViewBounds, GDIBounds, nPoints, PolyIn(1), nP2,
      GDIPoly(1))
641
642  'Draw optional polygon interior
643
644    'Select Case Polys(n).FillStyle
645    ' Case 0 'Solid fill
646    '    LogicalBrush.lbStyle = 0  'BS_SOLID
647    ' Case 1 'Null fill
648    '    LogicalBrush.lbStyle = 1  'BS_NULL
649    ' Case Else 'Hatch fill of some sort
650    '    LogicalBrush.lbStyle = 2  'BS_HATCHED
651    '    LogicalBrush.lbHatch = Polys(n).FillStyle - 2
652    'End Select
653    'LogicalBrush.lbColor = Polys(n).FillColor
654    'hBrush = CreateBrushIndirect(LogicalBrush)
655    'hOldBrush = SelectObject(hDCMeta, hBrush)
656      'Possible outline
657    'If Polys(n).OutlineWidth > 0 Then
658    '       hPen = CreatePen(6, Polys(n).OutlineWidth, Polys(n).OutlineColor)
659    '       hOldPen = SelectObject(hDCMeta, hPen)
660    ' Else
661    '       hOldPen = SelectObject(hDCMeta, GetStockObject(NULL_PEN))
662    'End If
663    If flgSkipToBottom = False Then
664       SetBrushInWMF Polys(n).FillStyle, Polys(n).FillColor
```

```
665|       SetPenInWMF Polys(n).OutlineColor, Polys(n).OutlineWidth, 6
666|       ReturnVal = Polygon(hDCMeta, GDIPoly(1), iPoints)
667|       If ReturnVal = 0 Then Beep: Debug.Print "Problem writing polygon #" & n & " to
      metatfile"
668|     End If
669|     'hOldBrush = DeleteObject(hBrush)
670|     'hPen = DeleteObject(hPen)
671|
672|  End If
673|
674|
675|  Next n
676|
677|  LeaveShowPolygons5:
678|
679|  Exit Sub
680|
681|
682|  ERRShowPolygons5:
683|  flgSkipToBottom = True
684|  Resume Next
685|
686|  End Sub
687|
688|  Sub WMFRasterGrid (myIndex As Long, Junk As Long, nTheme As Long)
689|  On Error GoTo ERRShowRasterGrid
690|  Dim nR As Long, nC As Long, PtX As Single, PtY As Single, ColorOut  As Long
691|  If RasterGrids(myIndex).DataType <> DB_LONG Then Exit Sub
692|
693|  Dim CircleRadius As Integer, GDIPt As APIPoint
694|
695|
696|  CircleRadius = RasterGrids(myIndex).DeltaX * ScaleFactor + 1
697|
698|  SetPenInWMF 0&, 0, 5
699|  For nR = GridRowNumFromYCoord(ViewBounds.Bottom, RasterGrids(myIndex)) To
      GridRowNumFromYCoord(ViewBounds.Top, RasterGrids(myIndex))
700|    PtY = RasterGrids(myIndex).Y_Origin + nR * RasterGrids(myIndex).DeltaY
701|    For nC = GridColNumFromXCoord(ViewBounds.Left, RasterGrids(myIndex)) To
      GridColNumFromXCoord(ViewBounds.Right, RasterGrids(myIndex))
702|       DoEvents
703|       PtX = RasterGrids(myIndex).X_Origin + nC * RasterGrids(myIndex).DeltaX
704|       ColorOut = GetIJCellINT4FromGlobalMem(nC, nR, RasterGrids(myIndex),
      RasterGrids(myIndex).hGMem)
705|     ' ColorOut = GetXYCellINT4FromGlobalMem(PtX, PtY, RasterGrids(myIndex),
      RasterGrids(myIndex).hGMem)
706|       If ColorOut <> 0 Then
707|        SetBrushInWMF 0, ColorOut
708|        GDIPt.X = GDIBounds.Left + (PtX - ViewBounds.Left) * ScaleFactor
709|        GDIPt.Y = GDIBounds.Bottom + (PtY - ViewBounds.Bottom) * ScaleFactor
710|        ColorOut = Ellipse(hDCMeta, GDIPt.X - CircleRadius, GDIPt.Y - CircleRadius,
      GDIPt.X + CircleRadius, GDIPt.Y + CircleRadius)
711|       End If
712|    Next nC
```

```
Next nR
STOPShowRasterGrid:
Printer.DrawWidth = 1
Exit Sub
ERRShowRasterGrid:
Select Case Err
 Case Else
 MsgBox "Erro #" & Err & " in ShowRasterGrid." & CRLF
 Exit Sub
 Resume
End Select

End Sub

Sub WMFThemes (ThemesToPrint() As typTheme, ViewRect As RealRect, PrintRect As
RealRect, strTitle As String, posTitle As RealPoint, Caption, posCaption As RealPoint,
QSDataOut() As typTaggedIDNum, lngQSHighlightColor As Long,
sngQSHighlightRadius As Single, LastQS As RealPoint)

Dim nTh As Long, mm As Long
Dim nThemesShownHere As Long
  Screen.MousePointer = 11
'ThemesToPrint is an array full of ThemesToPrint to be printed.
'ViewRect is the area, in world coordinates, to be printed
'PrintRect is the location, on the Printer, where the map of the area
'       ViewRect should show up.
'The units of PrintRect are in MILLIMETERS.

Dim PrinterHeight As Single, PrinterWidth As Single
Printer.ScaleMode = 6'for millimeters
PrinterHeight = Printer.ScaleHeight
PrinterWidth = Printer.ScaleWidth

'These ranges leave some space on most printers, to accomodate the
' the rollers and such.  However, these will be used to define the
' the edges.

'Figure out whether the width or height will be the limiting
'  factor.
Dim XScale As Single, YScale As Single, UsedScale As Single
XScale = Abs((PrintRect.Right - PrintRect.Left) / (ViewRect.Right - ViewRect.Left))
YScale = Abs((PrintRect.Top - PrintRect.Bottom) / (ViewRect.Top - ViewRect.Bottom))
If XScale < YScale Then
   UsedScale = XScale
 Else
   UsedScale = YScale
End If



'Set the scale of the Printer up so that the corner of the ViewRect map into the corners of
' the PrintRect.
PrintScale.Left = ViewRect.Left - PrintRect.Left / UsedScale
PrintScale.Top = ViewRect.Top + PrintRect.Top / UsedScale
```

```
764| PrintScale.Bottom = PrintScale.Top - Printer.ScaleHeight / UsedScale
765| PrintScale.Right = PrintScale.Left + Printer.ScaleWidth / UsedScale
766|
767| Printer.Scale (PrintScale.Left, PrintScale.Top)-(PrintScale.Right, PrintScale.Bottom)
768|
769| nThemesShownHere = UBound(ThemesToPrint) - LBound(ThemesToPrint) + 1
770| 'ReDim mThemesToPrint(1 To nThemesShownHere)
771| 'Copy the themes to a module level array, so other subroutines
772| ' can get access to the names properly.
773| 'For mm = 1 To nThemesShownHere
774| '   CopyTYPTheme ThemesToPrint(mm), mThemesToPrint(mm)
775| 'Next mm
776|
777|
778| LSet ViewBounds = ViewRect
779|
780| If flgStopPrinting Then Exit Sub
781|
782| For mm = 1 To nThemesShownHere
783|     nTh = mm
784|
785|    DoEvents
786|    If flgStopPrinting Then GoTo ExitShowThemes
787| Select Case ThemesToPrint(nTh).ThemeType
788|  Case JOINTEDLINE:  Call PrintLines5(ThemesToPrint(nTh).FirstLoc,
    ThemesToPrint(nTh).LastLoc, nTh)
789|  Case ONEPOINT:    Call PrintPoints3(ThemesToPrint(nTh).FirstLoc,
    ThemesToPrint(nTh).LastLoc, nTh)
790|  Case TILEBLOCK:   Call PrintTiles4(ThemesToPrint(nTh).FirstLoc,
    ThemesToPrint(nTh).LastLoc, nTh)
791|  Case HATCHLINE:   Call PrintHatchLines3(ThemesToPrint(nTh).FirstLoc,
    ThemesToPrint(nTh).LastLoc, nTh)
792|  Case SOLIDPOLYGON: Call PrintPolygons5(ThemesToPrint(nTh).FirstLoc,
    ThemesToPrint(nTh).LastLoc, nTh)
793|  Case LABELTEXT:   Call PrintLabels1(Themes(nTh).FirstLoc, Themes(nTh).LastLoc,
    nTh)
794| End Select
795|
796| Next mm
797|
798| ExitShowThemes:
799|
800| 'Display the current spatial selection
801|  PrinterCircleQSResults QSDataOut(), lngQSHighlightColor, sngQSHighlightRadius,
    LastQS
802|
803|
804|
805| Printer.Line (ViewRect.Left, ViewRect.Top)-(ViewRect.Right, ViewRect.Bottom), RGB(0,
    0, 0), B
806| Printer.ForeColor = RGB(0, 0, 0)
807|
808| Dim Dummy As Integer
809| 'Print coordinates, whether the user wanted them or not.
```

```
810|  Printer.FontSize = 8
811|  Printer.FontName = "Arial"
812|  'Print bounds of top
813|  Dummy = SetTextAlign(Printer.hDC, TA_BOTTOM + TA_CENTER)
814|  Printer.CurrentX = (ViewRect.Left + ViewRect.Right) / 2
815|  Printer.CurrentY = ViewRect.Top
816|  Printer.Print ViewRect.Top
817|  'Print Bounds of bottom
818|  Dummy = SetTextAlign(Printer.hDC, TA_TOP + TA_CENTER)
819|  Printer.CurrentY = ViewRect.Bottom
820|  Printer.CurrentX = (ViewRect.Left + ViewRect.Right) / 2
821|  Printer.Print ViewRect.Bottom
822|  'Print Bounds of left side
823|  Dummy = SetTextAlign(Printer.hDC, TA_BASELINE + TA_RIGHT)
824|  Printer.CurrentY = (ViewRect.Bottom + ViewRect.Top) / 2
825|  Printer.CurrentX = (ViewRect.Left)
826|  Printer.Print ViewRect.Left
827|  'Print Bounds of left side
828|  Dummy = SetTextAlign(Printer.hDC, TA_BASELINE + TA_LEFT)
829|  Printer.CurrentY = (ViewRect.Bottom + ViewRect.Top) / 2
830|  Printer.CurrentX = (ViewRect.Right)
831|  Printer.Print ViewRect.Right
832|
833|
834|
835|
836|
837|  If strTitle <> "" Then
838|    Printer.ForeColor = RGB(0, 0, 0)
839|    Printer.ScaleMode = 6'Put the Printer back into millimeters
840|    Printer.CurrentX = posTitle.X
841|    Printer.CurrentY = posTitle.Y
842|    Printer.FontName = "Times New Roman"
843|    Printer.FontSize = 18
844|    Printer.FontBold = True
845|    Printer.Print strTitle
846|  End If
847|
848|  Printer.EndDoc
849|  Screen.MousePointer = 0
850|  Beep
851|  frmMap.lblPercentDone = "DONE PRINTING"
852|  Exit Sub
853|
854|
855|
856|  End Sub
857|
858|  Sub WMFTiles4 (FirstTile As Long, LastTile As Long, nTheme As Long)
859|  'GWPouch 940720 Converted to use GDI Rectangle, and full outline tiles.
860|
861|  Dim flgSkipPlotting As Integer
862|  ReDim Pts(1 To 2) As RealPoint, GDIPts(1 To 2) As APIPoint
863|  Dim n As Long, L  As Long, One As Long, Two  As Long, dummyInt As Integer
```

```
Dim hBrush  As Integer, hOldBrush As Integer, hPen As Integer, hOldPen As Integer
One = 1&: Two = 2&

nTiles = LastTile - FirstTile + 1

For n = FirstTile To LastTile
  flgSkipPlotting = False
  DoEvents

   L = Tiles(n).nStart
   Call GetRlPntsFromGlobalMem(hGMemAllPoints, Pts(1), L, Two)

'Before trying to plot the tile, check to see whether it falls in
' the current viewing rectangle.

   If Pts(1).X > ViewBounds.Right Then flgSkipPlotting = True
   If Pts(2).X < ViewBounds.Left Then flgSkipPlotting = True
   If Pts(1).Y < ViewBounds.Bottom Then flgSkipPlotting = True
   If Pts(2).Y > ViewBounds.Top Then flgSkipPlotting = True

   'Select Case Tiles(n).InteriorFillStyle
   ' Case 0 'Solid fill
   '    LogicalBrush.lbStyle = 0  'BS_SOLID
   ' Case 1 'Null fill
   '    LogicalBrush.lbStyle = 1  'BS_NULL
   ' Case Else 'Hatch fill of some sort
   '    LogicalBrush.lbStyle = 2  'BS_HATCHED
   '    LogicalBrush.lbHatch = Tiles(n).InteriorFillStyle - 2
   'End Select
   'LogicalBrush.lbColor = Tiles(n).InteriorColor
   'hBrush = CreateBrushIndirect(LogicalBrush)
   'hOldBrush = SelectObject(hDCMeta, hBrush)
   '  'Possible outline
   'If Tiles(n).OutlineWidth > 0 Then
   '       hPen = CreatePen(Tiles(n).OutlineStyle, Tiles(n).OutlineWidth,
    Tiles(n).OutlineColor)
   '       hOldPen = SelectObject(hDCMeta, hPen)
   ' Else
   '       hOldPen = SelectObject(hDCMeta, GetStockObject(NULL_PEN))
   'End If

   If Not flgSkipPlotting Then
   Call ScaleR4To2LOCAL(ViewBounds, GDIBounds, Two, Pts(1), GDIPts(1))
   SetBrushInWMF Tiles(n).InteriorFillStyle, Tiles(n).InteriorColor
   SetPenInWMF Tiles(n).OutlineColor, Tiles(n).OutlineWidth, Tiles(n).OutlineStyle
   dummyInt = Rectangle(hDCMeta, GDIPts(1).X, GDIPts(1).Y, GDIPts(2).X, GDIPts(2).Y)
   'ddDebug.Print n, hPen, hBrush, dummyInt
   End If
   hOldBrush = DeleteObject(hBrush)
   hPen = DeleteObject(hPen)


Next n
Exit Sub
```

```
917|
918| ERRPrintTiles4:
919| ' Ultra-primitive error-ignoring
920|  flgSkipPlotting = True
921|  Resume Next
922|
923|
924| End Sub
925|
926| Sub WritePlaceableWMFHeader_HIMETRIC (WMFName As String)
927| ' Unfortunately, these need to be converted to inches for further use.
928| 'Placeable Windows Metafiles
929| Dim PlaceableMetafileHeader As METAFILEHEADER
930| Dim Binstuff As MetaFileHeaderAsBinary
931| '
932| 'The following comes from the Windows 3.1 SDK documentation.
933| 'A placeable Windows metafile is a standard Windows metafile that has an additional
    22-byte
934| 'header. The header contains information about the aspect ratio and original size of the
    metafile,
935| 'permitting applications to display the metafile in its intended form.
936| 'The header for a placeable Windows metafile has the following form:
937| 'typedef struct {
938| '   DWORD   key;
939| '   HANDLE  hmf;
940| '   RECT    bbox;
941| '       'typedef struct tagRECT {    /* rc */
942| '       '  int left;
943| '       '  int top;
944| '       '  int right;
945| '       '  int bottom;
946| '       '} RECT
947| '   WORD    inch;
948| '   DWORD   reserved;
949| '   WORD    checksum;
950| '} METAFILEHEADER;
951| '
952| 'Following are the members of a placeable metafile header:
953| '
954| 'key Specifies the binary key that uniquely identifies this file type. This member must be
    set
955| 'to 0x9AC6CDD7L.
956| 'hmf Unused; must be zero.
957| 'bbox   Specifies the coordinates of the smallest rectangle that encloses the picture. The
958| 'coordinates are in metafile units as defined by the inch member.
959| 'inch   Specifies the number of metafile units to the inch. To avoid numeric overflow, this
    value
960| 'should be less than 1440. Most applications use 576 or 1000.
961| 'reserved   Unused; must be zero.
962| 'checksum   Specifies the checksum. It is the sum (using the XOR operator) of the first
    10 words of
963| 'the header.
964| 'The actual content of the Windows metafile immediately follows the header. The format
    for this content
```

```
'is identical to that for standard Windows metafiles. For some applications, a placeable Windows
'metafile must not exceed 64K.
'Note:  Placeable Windows metafiles are not compatible with the GetMetaFile function. Applications
'that intend to use the metafile functions to read and play placeable Windows metafiles must read
'the file by using an input function (such as _lread), strip the 22-byte header, and create a
'standard Windows metafile by using the remaining bytes and the SetMetaFileBits function.

  PlaceableMetafileHeader.key = &H9AC6CDD7
  PlaceableMetafileHeader.HANDLE = 0
      'The (BoundsLeft-BoundsRight)/InchesPerGDIUnit will give picture height in inches
  PlaceableMetafileHeader.BoundsLeft = GDIBounds.Left
  PlaceableMetafileHeader.BoundsTop = GDIBounds.Top
  PlaceableMetafileHeader.BoundsRight = GDIBounds.Right
  PlaceableMetafileHeader.BoundsBottom = GDIBounds.Bottom

  PlaceableMetafileHeader.inchPerGDIUnit = 2540   'hundredths of millimeter in one inch
  PlaceableMetafileHeader.reserved = 0
LSet Binstuff = PlaceableMetafileHeader
Dim n As Long
  PlaceableMetafileHeader.checksum = 0
For n = 0 To 9
  PlaceableMetafileHeader.checksum = PlaceableMetafileHeader.checksum Xor Binstuff.bin(n)
Next n

'Get a temporary file name
Dim TempName As String * 255
TempName = "ZRQLJEQ"
If Dir$(TempName) <> "" Then Kill TempName
'n = GetTempFileName(0, "MF", 0, TempName)
Open TempName For Binary Access Write As 255
Put 255, , PlaceableMetafileHeader

Open WMFName For Binary Access Read As 254
n = LOF(254) 'n now stores the length of file of the targeted metafile

Dim ch As String
If n > 4096 Then
  ch = String(4096, " ")
  Dim nChunk As Integer
  For nChunk = 1 To n \ 4096
  Get 254, , ch
  Put 255, , ch
  Next nChunk
  ch = String(n Mod 4096, " ")
  Get 254, , ch
  Put 255, , ch
  Close 254
```

```
1012|   Else
1013|     ch = String(n, " ")
1014|     Get 254, , ch
1015|     Put 255, , ch
1016|     Close 254
1017|  End If
1018|  Close 255
1019|   Kill WMFName
1020|     Beep
1021|   Name TempName As WMFName
1022| End Sub
1023|
1024| Sub ZWMFBeachBall (X As Single, Y As Single, Color As Long)
1025|   Dim TempColor As Long
1026|   Dim Radius As Single
1027|   Radius = (ViewBounds.Right - ViewBounds.Left) * .025
1028|   'TempColor = -Color
1029|   WMFDrawFilledCircle X, Y, Radius, Color
1030|   WMFDrawFilledCircle X, Y, .75 * Radius, Not Color
1031|
1032|   WMFDrawFilledCircle X, Y, .5 * Radius, Color
1033|   WMFDrawFilledCircle X, Y, .25 * Radius, Not Color
1034| Exit Sub
1035|
1036| '       On Error Resume Next
1037| '     'Make a beachball
1038| '   'Find the bounding rectangle for the circle, then draw the two wedges.
1039|       Dim XR As Integer, XL As Integer, YB As Integer, YT As Integer
1040|       Dim XX As Integer, YY As Integer, A As Single
1041| '     XX = FindX_WMF(X): YY = FindY_WMF(Y)
1042| '     A = X + (ViewBounds.Right - ViewBounds.Left) * .025
1043| '     XR = FindX_WMF(A)
1044| '     A = X - (ViewBounds.Right - ViewBounds.Left) * .025
1045| '     XL = FindX_WMF(A)
1046| '     A = Y + (ViewBounds.Right - ViewBounds.Left) * .025
1047| '     YT = FindY_WMF(A)
1048| '     A = Y - (ViewBounds.Right - ViewBounds.Left) * .025
1049| '     YB = FindY_WMF(A)
1050| '     dDebug.Print XX, YY, , XL, XR, , , YT, YB
1051| '     sStop
1052| '     SetPenInWMF Color, 1, 0
1053| '     SetBrushInWMF 0, Color
1054| '     DummyLong = Chord(hDCMeta, XL, YT, XR, YB, XR, YY, XX, YT)
1055| '     DummyLong = Chord(hDCMeta, XL, YT, XR, YB, XL, YY, XX, YB)
1056| ' SetBrushInWMF 0, RGB(255, 0, 255)
1057| '     DummyLong = Chord(hDCMeta, XL, YB, XR, YT, XR, YY, XX, YT)
1058| '     DummyLong = Chord(hDCMeta, XL, YB, XR, YT, XL, YY, XX, YB)
1059| '
1060| '
1061| End Sub
1062|
```

```
 1| Option Explicit
 2| Global One  As Long, Zero As Long, Two As Long
 3| Declare Function GlobalAlloc Lib "Kernel" (ByVal wFlags As Integer, ByVal dwBytes As
    Long) As Integer
 4| Declare Function GlobalReAlloc Lib "Kernel" (ByVal hMem As Integer, ByVal dwBytes
    As Long, ByVal wFlags As Integer) As Integer
 5| Declare Function GlobalHandleToSel Lib "Toolhelp.DLL" (ByVal Handle As Integer) As
    Integer
 6| Declare Function GlobalFree Lib "Kernel" (ByVal hMem As Integer) As Integer
 7| Declare Function GlobalSize Lib "Kernel" (ByVal hMem As Integer) As Long
 8| Declare Function MemoryRead Lib "Toolhelp.dll" (ByVal wSel As Integer, ByVal
    dwOffSet As Long, lpvBuf As Any, ByVal dwcb As Long) As Long
 9| Declare Function MemoryWrite Lib "Toolhelp.dll" (ByVal wSel As Integer, ByVal
    dwOffSet As Long, lpvBuf As Any, ByVal dwcb As Long) As Long
10| Declare Function GlobalLock Lib "Kernel" (ByVal hMem As Integer) As Long
11| Declare Function GlobalUnlock Lib "Kernel" (ByVal hMem As Integer) As Integer
12| Declare Function GetFreeSpace Lib "Kernel" (ByVal wFlags As Integer) As Long
13|
14|
15| 'Type RealPoint
16| '   X As Single
17| '   Y As Single
18| 'End Type
19| 'Global lngHGAllPointsMemSize As Long
20|
21|
22|
23|
24|
25|
26| Global Const GMEM_FIXED = &H0
27| Global Const GMEM_MOVEABLE = &H2
28| Global Const GMEM_NOCOMPACT = &H10
29| Global Const GMEM_NODISCARD = &H20
30| Global Const GMEM_INITZERO = &H40
31| Global Const GMEM_ZEROINIT = &H40
32| Global Const GMEM_MODIFY = &H80
33| Global Const GMEM_LOCKCOUNT = &HFF
34| Global Const GMEM_DISCARDABLE = &H100
35| Global Const GMEM_SHARE = &H2000
36| Global Const GMEM_DDESHARE = &H2000
37| Global Const GMEM_DISCARDED = &H4000
38| Global Const GHND = (GMEM_MOVEABLE Or GMEM_ZEROINIT)
39|
40|
41|
42| Type RlPnt
43|    X As Single
44|    Y As Single
45| End Type
46|
47|
48| Const NUM_ELEMENTS = 1000
49| Const ELEMENT_SIZE = 4
```

```
50|
51|
52|  Global Int1 As Integer, int2 As Integer
53|  Global Long1 As Long, Long2 As Long
54|  Global TestPoint(5) As RlPnt
55|
56|  Sub CheckAPIPointsR4PointsMatch (hGMemReal As Integer, hGMemAPI As Integer)
57|  Dim nPointsReal As Long, nPointsAPI
58|     nPointsReal = GlobalSize(hGMemReal)
59|     nPointsAPI = GlobalSize(hGMemAPI)
60|     If nPointsAPI * 2 < nPointsReal Then
61|        hGMemAPI = GetMoreWinMem(hGMemAPI, nPointsReal / 2)
62|     End If
63|
64|  End Sub
65|
66|  Sub GetAPIPntsFromGlobalMem (hGMem As Integer, FirstPoint As APIPOINT, nFirst As
     Long, nPoints As Long)
67|  ' Read XY (integer*2) points from global memory array.
68|  Static nBytes As Long, FirstLoc As Long, nOut As Long, hSel As Integer
69|
70|  nBytes = nPoints * 4
71|  FirstLoc = nFirst * 4
72|  hSel = GetSelector(hGMem)
73|  nOut = MemoryRead(hSel, FirstLoc, FirstPoint, nBytes)
74|
75|  End Sub
76|
77|  Function GetMoreWinMem (hGMem As Integer, lngTotalMemory As Long) As Integer
78|  Dim Temp As Integer
79|  Temp = hGMem
80|  Temp = GlobalReAlloc(Temp, lngTotalMemory, GMEM_INITZERO)
81|  If Temp = 0& Then
82|     'There is some reason to panic
83|     MsgBox "Could not allocate more memory.  Close some applictaions and/or data files
     to free up some memory.", 0, "DANGER! DANGER! WARNING WIL ROBINSON!
     DANGER!"
84|   Else
85|    hGMem = Temp
86|  End If
87|
88|  GetMoreWinMem = Temp
89|  'Note that changing the size of the memory block can change the
90|  '   the global memory handle.
91|  End Function
92|
93|  Sub GetRlPntsFromGlobalMem (hGMem As Integer, FirstPoint As RealPoint, nFirst As
     Long, nPoints As Long)
94|  ' Read XY (real*4) points from global memory array.
95|  Static nBytes As Long, FirstLoc As Long, nOut As Long, hSel As Integer
96|  Dim ptrBaseGlobal As Long
97|
98|
99|  nBytes = nPoints * 8
```

```
100  FirstLoc = nFirst * 8
101  '"ptrBaseGlobal = GlobalLock(hGMem)
102  '"Call HUGEMemCopyRIPts(FirstPoint, nBytes, Zero, ptrBaseGlobal,
       lngHGAllPointsMemSize, FirstLoc, nBytes)
103  '"ptrBaseGlobal = GlobalUnlock(hGMem)
104  '""Sub HUGEMemCopy (ByVal PTRBaseFrom As Long, nFromSizeInBytes As Long,
       nFromStartInBytes As Long, ByVal PTRBaseTo As Long, nFromSizeInBytes As Long,
       nFromStartInBytes As Long, nBytesToCopy As Long)
105
106
107
108  hSel = GetSelector(hGMem)
109  nOut = MemoryRead(hSel, FirstLoc, FirstPoint, nBytes)
110
111  End Sub
112
113  Function GetSelector (hGMem As Integer) As Integer
114  ' Get the selector.
115
116  GetSelector = GlobalHandleToSel(hGMem)
117  ' Write random values to global memory.
118  ' Free the global memory.
119  'X = GlobalFree(Handle)
120  End Function
121
122  Function GetWinMem (lngHowMuch As Long) As Integer
123  'This function does an initial allocation of memory, and returns the
124  ' hGMem (hGMEM).
125  'lngHowMuch is specified in bytes.
126  Dim hGMem As Integer, Selector As Integer
127  Dim X As Integer, Z As Single, I As Integer
128  Dim Offset As Long
129
130  ' Display hourglass mouse pointer.
131
132  Screen.MousePointer = 11
133
134  ' Allocate the needed memory.
135
136  hGMem = GlobalAlloc(GMEM_MOVEABLE Or GMEM_ZEROINIT, lngHowMuch)
137
138  ' Display error message if alloction failed.
139     Screen.MousePointer = 0
140  If hGMem = 0 Then
141     MsgBox "Could not allocate memory; close some applications or files or something."
142    Else
143     GetWinMem = hGMem
144  End If
145  End Function
146
147  Sub SetRIPntsToGlobalMem (hGMem As Integer, FirstPoint As RealPoint, nFirst As
       Long, nPoints As Long)
148  'This subroutine puts XY points (Single/ Real*4) into a global memory array.
149  ' Note that it is a zero-based global memory array (ie, the first element is in location 0
```

```
150| ' Note also that it modifies the value of nFirst to point at the memory location
151| '  after the last one it put something into, so this can be easily used
152| '  to load points using a location counter.  HOWEVER, this also means
153| '  that you shouldn't pass it an integer you need to use for something else.
154| Static FirstLoc As Long, nBytes As Long, nOut As Long, hSel  As Integer
155| FirstLoc = nFirst * 8
156| nBytes = nPoints * 8
157| If FirstLoc + nBytes > lngHGAllPointsMemSize Then
158|    lngHGAllPointsMemSize = lngHGAllPointsMemSize + 1048576 'Grab another MByte
159|    hGMem = GetMoreWinMem(hGMem, lngHGAllPointsMemSize)
160| End If
161| """Dim ptrBaseGlobal As Long
162| """ptrBaseGlobal = GlobalLock(hGMem)
163| """Call HUGEMemCopyRlPts(FirstPoint, nBytes, Zero, ptrBaseGlobal,
    lngHGAllPointsMemSize, FirstLoc, nBytes)
164| """ptrBaseGlobal = GlobalUnlock(hGMem)
165| """""""""Sub HUGEMemCopy (ByVal PTRBaseFrom As Long, nFromSizeInBytes As Long,
    nFromStartInBytes As Long, ByVal PTRBaseTo As Long, nFromSizeInBytes As Long,
    nFromStartInBytes As Long, nBytesToCopy As Long)
166|
167| hSel = GetSelector(hGMem)
168| nOut = MemoryWrite(hSel, FirstLoc, FirstPoint, nBytes)
169| nFirst = nFirst + nPoints
170| End Sub
171|
172| Sub ShutDownAndFreeMem ()
173| 'Provides exiting for Wheat.
174| '940722 GWPouch.  Discovered that I had a severe (read:total) memory
175| '  leak from Wheat.  This turned out to be caused by trying to release
176| '  the global memory in frmMap_Unload and frmMap_queryUnload, which
177| '  do not seem to get called.  I think this has to do with it being a
178| '  MDI child now.  Anyway, MDIWheat_Unload doesn't get called either,
179| '  so instead of relying on form_unload procedures to free the memory,
180| '  you should exit by calling this function in the form_mfExit event.
181|
182|  Do While (GlobalUnlock(hGMemAllPoints)): Loop
183|  Do While (GlobalFree(hGMemAllPoints)): Loop
184|  Dim n As Long
185|  For n = 1 To UBound(RasterGrids)
186|     Do While GlobalFree(RasterGrids(n).hGMem): Loop
187|  Next n
188| End
189|    ZapCursors
190|    ZapIcons
191|
192| End Sub
193|
```

```
 1| Option Explicit
 2|
 3| Type typGridDescription
 4|   DataType As Long 'The type of data in the grid.  This how the numbers will be pulled
    out.
 5|                    ' Use the same datacodes as in MSAccess (ie, DB_LONG=4 for integer*4)
 6|   DataSize As Long 'Always set this field, its used in retrieving cell values
 7|                    'Probably should not do this, but if you want to store structures, this
 8|                    ' could come in very handy.
 9|   X_Origin As Single
10|   Y_Origin As Single
11|   DeltaX As Single
12|   DeltaY As Single
13|   nColsX As Long
14|   nRowsY As Long
15|   BlankVal As String * 8'A value that the user can flag null values with.
16|                        ' not supported, and the next programmer's responsibility.
17|   LayerName As String * 80
18|   hGMem As Integer
19| End Type
20|
21| Global ZZZGridDescription As typGridDescription 'An extra one kept on hand for
    debugging
22|
23| Type stringg8
24|   f As String * 8
25| End Type
26| Global ZZZTempString As String
27|
28| Global ALLTypesAllowed(1 To 8)  As Integer
29|
30| Function CellOffsetFromXY (X As Single, Y As Single, GridDesc As typGridDescription)
    As Long
31|  On Error GoTo ERRCellOffsetFromXY
32|  Dim RowNum As Long, ColNum As Long
33|  RowNum = GridRowNumFromYCoord(Y, GridDesc)
34|   'Check that we're in the area covered by the grid.
35|   If RowNum < 0 Then GoSub 100
36|   If RowNum >= GridDesc.nRowsY Then GoSub 100
37|  ColNum = GridColNumFromXCoord(X, GridDesc)
38|   'Check that we're in the area covered by the grid.
39|   If ColNum < 0 Then GoSub 100
40|   If ColNum >= GridDesc.nColsX Then GoSub 100
41|
42|  CellOffsetFromXY = (RowNum * GridDesc.nColsX + ColNum) * GridDesc.DataSize
43|
44|
45|
46| Exit Function
47| ERRCellOffsetFromXY:
48|   Select Case Err
49|     Case Else
50|      MsgBox "Error #" & Err & " occured in CellOffsetFromXY.    " & Error
51|      Err = 0
```

```
52|         Exit Function
53|         Resume
54|     End Select
55|
56|
57| 100 :
58|  CellOffsetFromXY = -1
59|  Exit Function
60|  Return
61|
62|
63| End Function
64|
65| Sub ColorGridFromOtherGrid (GridNameIn As String, GridNameOut As String,
    ColorTableName As String, DB As Database)
66| On Error GoTo ERRColorGridFromR4Grid
67|
68| ReDim ColorLookup(0 To 0) As ColorLUT
69| ZLoadColorLUT ColorTableName, DB, ColorLookup()
70|
71| Dim dnsGrids  As Dynaset
72| Dim hGMemIn As Integer, hGMemOut As Integer, GridSize As Long
73| Dim GridInDesc As typGridDescription, GridOutDesc As typGridDescription
74|
75|
76| Set dnsGrids = DB.CreateDynaset("zWHEATtblRasterGrids")
77| Call LoadRasterGridIntoGlobalMem(GridNameIn, GridInDesc, DB)
78|
79| If GridInDesc.DataType > DB_DATE Then MsgBox "Cannot create color grid from the
    input grid.  Exiting subroutine.": Exit Sub
80| GridOutDesc = GridInDesc
81| 'Make the output a color grid.
82| GridOutDesc.DataType = DB_LONG: GridOutDesc.DataSize = 4
83|
84|
85| GridSize = GridOutDesc.DataSize * GridOutDesc.nColsX * GridOutDesc.nRowsY
86| GridOutDesc.hGMem = GetWinMem(GridSize)
87|
88| Dim nR As Long, nC As Long
89| Dim ValueIn As Variant, ColorOut As Long
90| For nR = 0 To GridInDesc.nRowsY
91|  For nC = 0 To GridInDesc.nColsX
92|   DoEvents
93|     ValueIn = GetIJCellValueGenericFromGlobalMem(nC, nR, GridInDesc,
    GridInDesc.hGMem)
94|     'If ValueIn <> 0 Then S top
95|     ColorOut = ZApplyCLUT(ValueIn, ColorLookup())
96|     Call SetIJCellInt4InGlobalMem(nC, nR, GridOutDesc, GridOutDesc.hGMem,
    ColorOut)
97|  Next nC
98| Next nR
99|
100|  dnsGrids.AddNew
101|  dnsGrids("DataType") = GridOutDesc.DataType
```

```
102|    dnsGrids("DataSize") = GridOutDesc.DataSize
103|    dnsGrids("X_Origin") = GridOutDesc.X_Origin
104|    dnsGrids("Y_Origin") = GridOutDesc.Y_Origin
105|    dnsGrids("DeltaX") = GridOutDesc.DeltaX
106|    dnsGrids("DeltaY") = GridOutDesc.DeltaY
107|    dnsGrids("nColsX") = GridOutDesc.nColsX
108|    dnsGrids("nRowsY") = GridOutDesc.nRowsY
109|    dnsGrids("BlankVal") = GridOutDesc.BlankVal
110|    dnsGrids("GridName") = GridNameOut
111|    dnsGrids("DateModified") = Now
112|    PullGlobalMemBlockIntoDB GridOutDesc.hGMem, dnsGrids, "GridData"
113|    dnsGrids.Update
114|    DummyLong = GlobalFree(GridOutDesc.hGMem): DummyLong =
     GlobalFree(GridInDesc.hGMem)
115|
116| Exit Sub
117|
118|
119| ERRColorGridFromR4Grid:
120|  Select Case Err
121|   Case 3011 'No table errors
122|    ZDefineGridTable DB
123|    MsgBox "There were no RasterGrids in this database.  Exiting subroutine."
124|    Err = 0
125|    Exit Sub
126|    Resume
127|   Case Else
128|    MsgBox "Error #" & Err & " occurred in PolygonQTToGrid.  " & CRLF & Error
129|    Err = 0
130|    Exit Sub
131|    Resume
132|  End Select
133|
134|
135|
136|
137|
138| End Sub
139|
140| Function GetIJCellINT4FromGlobalMem (IX As Long, JY As Long, GridDesc As
     typGridDescription, hGMem As Integer) As Long
141| Static Offset As Long, wSel  As Integer, DummyLong As Long, I4 As Long
142|
143| If GridDesc.DataType <> DB_LONG Then Beep: Exit Function
144| If (IX < 0) Or (IX >= GridDesc.nColsX) Then
145|    GetIJCellINT4FromGlobalMem = 0
146|    Exit Function
147| End If
148| If (JY < 0) Or (JY >= GridDesc.nRowsY) Then
149|    GetIJCellINT4FromGlobalMem = 0
150|    Exit Function
151| End If
152|
153|
```

```
154|
155| Offset = (JY * GridDesc.nColsX + IX) * GridDesc.DataSize
156| ' IX could put a function in here to check the size of the global memory block
157| '  before trying to do this, but it would slow it down to much and ther
158| '  should be other bounds checking before getting here.
159| '
160| 'If dnsIn(FieldName).FieldSize() < Offset Then
161| '   'There is not as much data as there ought to be for this data type.
162| '   Beep
163| '   Exit Function
164| 'End If
165| If Offset < 0 Then Exit Function
166| wSel = GlobalHandleToSel(hGMem)
167|
168| DummyLong = MemoryRead(wSel, Offset, I4, GridDesc.DataSize)
169| 'dDebug.Print I4, Offset
170| GetIJCellINT4FromGlobalMem = I4
171|
172|
173| End Function
174|
175| Function GetIJCellValueGenericFromGlobalMem (IColsX As Long, JRowsY As Long,
     GridDesc As typGridDescription, hGMem As Integer) As Variant
176| Static Offset As Long, wSel  As Integer, DummyLong As Long
177|
178| If IColsX < 0 Or IColsX >= GridDesc.nColsX Then Exit Function
179| If JRowsY < 0 Or JRowsY >= GridDesc.nRowsY Then Exit Function
180| Dim CurChunk As String
181| Offset = GridDesc.DataSize * (GridDesc.nColsX * JRowsY + IColsX)
182| ' IX could put a function in here to check the size of the global memory block
183| '  before trying to do this, but it would slow it down to much and ther
184| '  should be other bounds checking before getting here.
185| '
186| 'If dnsIn(FieldName).FieldSize() < Offset Then
187| '   'There is not as much data as there ought to be for this data type.
188| '   Beep
189| '   Exit Function
190| 'End If
191|
192| wSel = GlobalHandleToSel(hGMem)
193| CurChunk = Space(255)
194| DummyLong = MemoryRead(wSel, Offset, ByVal CurChunk, GridDesc.DataSize)
195| GetIJCellValueGenericFromGlobalMem = ExtractGenericData(CurChunk,
     GridDesc.DataType)
196|
197| End Function
198|
199| Sub GetRowFromI4RasterGrid (JY As Long, GridDesc As typGridDescription, DataOut()
     As Long)
200| If JY < 0 Or JY >= GridDesc.nRowsY Then Exit Sub
201| If GridDesc.DataType <> DB_LONG Then Exit Sub 'Maybe gripe at user, too.
202| If UBound(DataOut) - LBound(DataOut) + 1 <> GridDesc.nColsX Then
203|    'we have a problem in that we don't have a full line of data, or too much data
204|    ' halt it and give user an error message
```

```
    Beep
    MsgBox "Wrong dimensions on input array to SetRowIntoR4RasterGrid"
    Exit Sub
  End If
Dim wSel As Integer, Offset As Long, RowSize As Long
 Offset = (JY * GridDesc.nColsX) * GridDesc.DataSize
 wSel = GlobalHandleToSel(GridDesc.hGMem)
 RowSize = GridDesc.DataSize * GridDesc.nColsX
 DummyLong = MemoryRead(wSel, Offset, DataOut(LBound(DataOut)), RowSize)
End Sub

Sub GetRowFromR4RasterGrid (JY As Long, GridDesc As typGridDescription,
DataOut() As Single)
If JY < 0 Or JY >= GridDesc.nRowsY Then Exit Sub
If GridDesc.DataType <> DB_SINGLE Then Exit Sub 'Maybe gripe at user, too.
If UBound(DataOut) - LBound(DataOut) + 1 <> GridDesc.nColsX Then
    'we have a problem in that we don't have a full line of data, or too much data
    ' halt it and give user an error message
    Beep
    MsgBox "Wrong dimensions on input array to SetRowIntoR4RasterGrid"
    Exit Sub
  End If

 Dim wSel As Integer, Offset As Long, RowSize As Long
 Offset = (JY * GridDesc.nColsX) * GridDesc.DataSize
 wSel = GlobalHandleToSel(GridDesc.hGMem)
 RowSize = GridDesc.DataSize * GridDesc.nColsX
 DummyLong = MemoryRead(wSel, Offset, DataOut(LBound(DataOut)), RowSize)
End Sub

Sub GetRowFromRasterGrid ()

End Sub

Function GetXYCellINT1FromGlobalMem (X As Single, Y As Single, GridDesc As
typGridDescription, hGMem As Integer) As Integer
Static Offset As Long, wSel  As Integer, DummyLong As Long
Dim CurChunk As String
Offset = CellOffsetFromXY(X, Y, GridDesc)
' IX could put a function in here to check the size of the global memory block
'  before trying to do this, but it would slow it down to much and ther
'  should be other bounds checking before getting here.
'
'If dnsIn(FieldName).FieldSize() < Offset Then
'   'There is not as much data as there ought to be for this data type.
'   Beep
'   Exit Function
'End If

wSel = GlobalHandleToSel(hGMem)
CurChunk = Space(255)
DummyLong = MemoryRead(wSel, Offset, ByVal CurChunk, GridDesc.DataSize)
GetXYCellINT1FromGlobalMem = Asc(Left$(CurChunk, 1))

```

```
257| End Function
258|
259| Function GetXYCellINT2FromGlobalMem (X As Single, Y As Single, GridDesc As
     typGridDescription, hGMem As Integer) As Integer
260| Static Offset As Long, wSel  As Integer, DummyLong As Long, I2 As Long
261|
262| Offset = CellOffsetFromXY(X, Y, GridDesc)
263| ' IX could put a function in here to check the size of the global memory block
264| '  before trying to do this, but it would slow it down to much and ther
265| '  should be other bounds checking before getting here.
266| '
267| 'If dnsIn(FieldName).FieldSize() < Offset Then
268| '   'There is not as much data as there ought to be for this data type.
269| '   Beep
270| '   Exit Function
271| 'End If
272|
273| wSel = GlobalHandleToSel(hGMem)
274|
275| DummyLong = MemoryRead(wSel, Offset, I2, GridDesc.DataSize)
276| GetXYCellINT2FromGlobalMem = I2
277|
278|
279| End Function
280|
281| Function GetXYCellINT4FromGlobalMem (X As Single, Y As Single, GridDesc As
     typGridDescription, hGMem As Integer) As Long
282| Static Offset As Long, wSel  As Integer, DummyLong As Long, I4 As Long
283|
284| Offset = CellOffsetFromXY(X, Y, GridDesc)
285| ' IX could put a function in here to check the size of the global memory block
286| '  before trying to do this, but it would slow it down to much and ther
287| '  should be other bounds checking before getting here.
288| '
289| 'If dnsIn(FieldName).FieldSize() < Offset Then
290| '   'There is not as much data as there ought to be for this data type.
291| '   Beep
292| '   Exit Function
293| 'End If
294| If Offset < 0 Then Exit Function
295| wSel = GlobalHandleToSel(hGMem)
296|
297| DummyLong = MemoryRead(wSel, Offset, I4, GridDesc.DataSize)
298| 'dDebug.Print I4, Offset
299| GetXYCellINT4FromGlobalMem = I4
300|
301| End Function
302|
303| Function GetXYCellReal4FromGlobalMem (X As Single, Y As Single, GridDesc As
     typGridDescription, hGMem As Integer) As Single
304| Static Offset As Long, wSel  As Integer, DummyLong As Long, R4 As Single
305|
306| Offset = CellOffsetFromXY(X, Y, GridDesc)
307| ' IX could put a function in here to check the size of the global memory block
```

```
308|  ' before trying to do this, but it would slow it down to much and ther
309|  ' should be other bounds checking before getting here.
310|  '
311|  'If dnsIn(FieldName).FieldSize() < Offset Then
312|  '   'There is not as much data as there ought to be for this data type.
313|  '   Beep
314|  '   Exit Function
315|  'End If
316|
317|  wSel = GlobalHandleToSel(hGMem)
318|
319|  DummyLong = MemoryRead(wSel, Offset, R4, GridDesc.DataSize)
320|  GetXYCellReal4FromGlobalMem = R4
321|
322|  End Function
323|
324|  Function GetXYCellREAL8FromGlobalMem (X As Single, Y As Single, GridDesc As
       typGridDescription, hGMem As Integer) As Double
325|  Static Offset As Long, wSel  As Integer, DummyLong As Long, R8 As Double
326|
327|  Offset = CellOffsetFromXY(X, Y, GridDesc)
328|  ' IX could put a function in here to check the size of the global memory block
329|  '  before trying to do this, but it would slow it down to much and ther
330|  '  should be other bounds checking before getting here.
331|  '
332|  'If dnsIn(FieldName).FieldSize() < Offset Then
333|  '   'There is not as much data as there ought to be for this data type.
334|  '   Beep
335|  '   Exit Function
336|  'End If
337|
338|  wSel = GlobalHandleToSel(hGMem)
339|
340|  DummyLong = MemoryRead(wSel, Offset, R8, GridDesc.DataSize)
341|  GetXYCellREAL8FromGlobalMem = R8
342|
343|  End Function
344|
345|  Function GetXYCellTEXTFromGlobalMem (X As Single, Y As Single, GridDesc As
       typGridDescription, hGMem As Integer) As String
346|  Static Offset As Long, wSel  As Integer, DummyLong As Long
347|  Dim CurChunk As String
348|  Offset = CellOffsetFromXY(X, Y, GridDesc)
349|  ' IX could put a function in here to check the size of the global memory block
350|  '  before trying to do this, but it would slow it down to much and ther
351|  '  should be other bounds checking before getting here.
352|  '
353|  'If dnsIn(FieldName).FieldSize() < Offset Then
354|  '   'There is not as much data as there ought to be for this data type.
355|  '   Beep
356|  '   Exit Function
357|  'End If
358|
359|  wSel = GlobalHandleToSel(hGMem)
```

```
360  CurChunk = Space(GridDesc.DataSize)
361  DummyLong = MemoryRead(wSel, Offset, ByVal CurChunk, GridDesc.DataSize)
362  GetXYCellTEXTFromGlobalMem = Left$(CurChunk, GridDesc.DataSize)
363
364
365  End Function
366
367  Function GetXYCellValueGenericFromDNS (X As Single, Y As Single, GridDesc As
     typGridDescription, dnsIn As Dynaset, FieldName As String) As Variant
368  Static Offset As Long
369  Offset = CellOffsetFromXY(X, Y, GridDesc)
370
371  If Offset = -1 Then Exit Function
372
373  If dnsIn(FieldName).FieldSize() < Offset Then
374     'There is not as much data as there ought to be for this data type.
375     Beep
376     Exit Function
377  End If
378
379  GetXYCellValueGenericFromDNS =
     ExtractGenericData(dnsIn(FieldName).GetChunk(Offset, GridDesc.DataSize),
     GridDesc.DataType)
380
381  End Function
382
383  Function GetXYCellValueGenericFromGlobalMem (X As Single, Y As Single, GridDesc
     As typGridDescription, hGMem As Integer) As Variant
384  Static Offset As Long, wSel  As Integer, DummyLong As Long
385  Dim CurChunk As String
386  Offset = CellOffsetFromXY(X, Y, GridDesc)
387  If Offset < 0 Then Exit Function
388  ' IX could put a function in here to check the size of the global memory block
389  '  before trying to do this, but it would slow it down to much and ther
390  '  should be other bounds checking before getting here.
391  '
392  'If dnsIn(FieldName).FieldSize() < Offset Then
393  '   'There is not as much data as there ought to be for this data type.
394  '   Beep
395  '   Exit Function
396  'End If
397
398  wSel = GlobalHandleToSel(hGMem)
399  CurChunk = Space(255)
400  DummyLong = MemoryRead(wSel, Offset, ByVal CurChunk, GridDesc.DataSize)
401  GetXYCellValueGenericFromGlobalMem = ExtractGenericData(CurChunk,
     GridDesc.DataType)
402
403  End Function
404
405  Function GridColNumFromXCoord (X As Single, GridDesc As typGridDescription) As
     Long
406  'GWPouch 13August94 at home on a Saturday, Becky is now in Ohio
407  ' By definition (mine) the coordinates of a grid cell refer to its center point.
```

```
408| ' The cell's value should represent the value of the parameter in the neighborhood
409| ' of the cell-center
410| GridColNumFromXCoord = (X - GridDesc.X_Origin) / GridDesc.DeltaX
411|
412| End Function
413|
414| Function GridRowNumFromYCoord (Y As Single, GridDesc As typGridDescription) As
     Long
415| 'GWPouch 13August94 at home on a Saturday, Becky is now in Ohio
416| ' By definition (mine) the coordinates of a grid cell refer to its center point.
417| ' The cell's value should represent the value of the parameter in the neighborhood
418| ' of the cell-center
419| GridRowNumFromYCoord = (Y - GridDesc.Y_Origin) / GridDesc.DeltaY
420| 'Note also that the first row in the file/BLOB is numbered as zero
421| ' Sorry for following the CS convention, but it avoid a lot of n-1/n+1
422| ' problems later on, and cuts down on the number of multiplies to get data.
423|
424|
425| End Function
426|
427| Sub LoadBLOBIntoGlobalMem (dnsIn As Dynaset, FieldName As String, hGMem As
     Integer, nStartAt As Long)
428|    Dim NumChunks As Long, TotalSize As Long, Offset As Long
429|    Dim RemChunk As Long, Cursize As Long
430|    Dim IX As Long, ChunkSize As Long, CurChunk As String
431|    ChunkSize = 8192  ' Set size of chunk.
432|    ' Get field size.
433|    TotalSize = dnsIn(FieldName).FieldSize()
434|    NumChunks = TotalSize \ ChunkSize   ' Set number of chunks.
435|    ' Set number of remaining bytes.
436|    RemChunk = TotalSize Mod ChunkSize
437|    ' Set starting size of chunk.
438|    Cursize = ChunkSize
439|
440| Dim wSel As Integer, DummyLong As Long
441|    wSel = GlobalHandleToSel(hGMem)
442|    For IX = 0 To NumChunks
443|       If IX = NumChunks Then Cursize = RemChunk
444|       Offset = IX * ChunkSize
445|       CurChunk = dnsIn(FieldName).GetChunk(Offset, Cursize)
446|       DummyLong = MemoryWrite(wSel, Offset, ByVal CurChunk, Cursize)
447|       ' Write chunk to global memory using MemoryWrite
448|    Next IX
449|
450| End Sub
451|
452| Sub LoadRasterGridIntoGlobalMem (GridName As String, GridOutDesc As
     typGridDescription, DB As Database)
453| On Error GoTo ERRLoadGridIntoGlobalMem
454|
455| Dim dnsGridsFrom As Dynaset
456| Set dnsGridsFrom = DB.CreateDynaset("SELECT * FROM [zWHEATtblRasterGrids]
     WHERE [GridName]='" & GridName & "';")
457| dnsGridsFrom.MoveFirst
```

```
458| 'dnsGridsFrom.FindFirst "GridName='" & GridName & "'"
459|    GridOutDesc.LayerName = dnsGridsFrom("GridName")
460|    GridOutDesc.DataType = dnsGridsFrom("DataType")
461|    GridOutDesc.DataSize = dnsGridsFrom("DataSize")
462|    GridOutDesc.X_Origin = dnsGridsFrom("X_Origin")
463|    GridOutDesc.Y_Origin = dnsGridsFrom("Y_Origin")
464|    GridOutDesc.DeltaX = dnsGridsFrom("DeltaX")
465|    GridOutDesc.DeltaY = dnsGridsFrom("DeltaY")
466|    GridOutDesc.nColsX = dnsGridsFrom("nColsX")
467|    GridOutDesc.nRowsY = dnsGridsFrom("nRowsY")
468|    If IsNull(dnsGridsFrom("BlankVal")) Then
469|       'DO NOTHING
470|      Else
471|       GridOutDesc.BlankVal = dnsGridsFrom("BlankVal")
472|    End If
473|
474|    GridOutDesc.hGMem = GetWinMem(GridOutDesc.DataSize * GridOutDesc.nColsX *
     GridOutDesc.nRowsY)
475|    Call LoadBLOBIntoGlobalMem(dnsGridsFrom, "GridData", GridOutDesc.hGMem, 0&)
476|
477|
478| Exit Sub
479| ERRLoadGridIntoGlobalMem:
480|  Select Case Err
481|   Case 3011 'No table errors
482|    ZDefineGridTable DB
483|    Resume
484|   Case Else
485|    MsgBox "Error #" & Err & " occurred in LoadGridIntoGlobalMem.  " & CRLF & Error
486|    Err = 0
487|    Exit Sub
488|    Resume
489|  End Select
490|
491| End Sub
492|
493| Sub LoadRstGrdR4FromArray (GridDesc As typGridDescription, ZIn() As Single)
494| If GridDesc.DataType <> DB_SINGLE Then Exit Sub 'Maybe gripe at user, too.
495| Dim LIX As Long, UIX As Long, LJY As Long, UJY As Long, nColsX As Long, nRowsY
     As Long
496| LIX = LBound(ZIn, 1): UIX = UBound(ZIn, 1): nColsX = UIX - LIX + 1
497| LJY = LBound(ZIn, 2): UJY = UBound(ZIn, 2): nRowsY = UJY - LJY + 1
498| If nColsX <> GridDesc.nColsX Then
499|    MsgBox "There are a different number of columns in this array than there are in the
     RasterGrid."
500|    Exit Sub
501|   End If
502| If nRowsY <> GridDesc.nRowsY Then
503|    MsgBox "There are a different number of rows in this array than there are in the
     RasterGrid."
504|    Exit Sub
505|   End If
506|
507| Dim wSel As Integer, Offset As Long, RowSize As Long, JY As Long, IX As Long
```

```
508| wSel = GlobalHandleToSel(GridDesc.hGMem)
509| RowSize = GridDesc.DataSize * GridDesc.nColsX
510|
511| ReDim ZRow(LIX To UIX) As Single
512|
513|  For JY = LJY To UJY
514|    Offset = (JY * GridDesc.nColsX) * GridDesc.DataSize
515|    For IX = LIX To UIX
516|       ZRow(IX) = ZIn(IX, JY)
517|    Next
518|    DummyLong = MemoryRead(wSel, Offset, ZRow(LIX), RowSize)
519|  Next
520|
521| End Sub
522|
523| Sub LoadRstGrdR4ToArray (GridDesc As typGridDescription, ZOut() As Single)
524|
525| If GridDesc.DataType <> DB_SINGLE Then Exit Sub 'Maybe gripe at user, too.
526| ReDim ZOut(0 To GridDesc.nColsX - 1, 0 To GridDesc.nRowsY - 1)
527| Dim wSel As Integer, Offset As Long, RowSize As Long, JY As Long, IX As Long
528| wSel = GlobalHandleToSel(GridDesc.hGMem)
529| RowSize = GridDesc.DataSize * GridDesc.nColsX
530|
531| ReDim ZRow(0 To GridDesc.nColsX - 1) As Single
532|  For JY = 0 To GridDesc.nRowsY - 1
533|    Offset = (JY * GridDesc.nColsX) * GridDesc.DataSize
534|    DummyLong = MemoryRead(wSel, Offset, ZRow(0), RowSize)
535|    For IX = 0 To GridDesc.nColsX - 1
536|       ZOut(IX, JY) = ZRow(IX)
537|    Next
538|  Next
539| End Sub
540|
541| Sub PolygonQTToGrid (strSQLSource As String, strOutlineField As String, strValueField
    As String, GridOutDesc As typGridDescription, GridName As String, DB As Database)
542| Dim snpIn As snapshot, dnsOut As Dynaset
543| Dim hGMemGrid As Integer, GridSize As Long, Value As Variant
544|
545| On Error GoTo ERRPolygonQTToGrid
546| Set snpIn = DB.CreateSnapshot(strSQLSource)
547| Set dnsOut = DB.CreateDynaset("zWHEATtblRasterGrids")
548| GridSize = GridOutDesc.DataSize * GridOutDesc.nColsX * GridOutDesc.nRowsY
549| hGMemGrid = GetWinMem(GridSize)
550| Dim nPoints As Long, nPolys As Long, mPoly  As Long, Temp  As String
551|
552|   snpIn.MoveFirst : snpIn.MoveLast : snpIn.MoveFirst
553|   nPolys = snpIn.RecordCount
554|   For mPoly = 1 To nPolys
555|      nPoints = snpIn(strOutlineField).FieldSize()
556|      If nPoints < 12 Then GoTo BottomPolygonLoop1
557|      'There are no coordinates for this polygon, so skip to the the bottom of this loop.
558|       Temp = snpIn(strOutlineField): Temp = ForceBLOBToBePolygon(Temp)
559|       nPoints = Len(Temp) \ 8
560|      ReDim Pts(1 To nPoints) As RealPoint 'This will be properly dimensioned by
```

```
    BLOB2RealPoints
561|        Call ZLoadBLOB2RealPoints(Temp, Pts()) 'This implicitly updates
    MinX,MaxX,MinY, MaxY
562|        Value = snpIn(strValueField)
563|        Call PolygonToGrid(Pts(), Value, hGMemGrid, GridOutDesc)
564|        'dDebug.Print "Polygon # " & mPoly & " of " & nPolys & ".  nPoints=" & nPoints
565| BottomPolygonLoop1:
566|     snpIn.MoveNext
567|    'frmRasterGrid.pnlStatus = "On polygon " & mPoly & " of " & nPolys
568|    Next mPoly
569|
570|    dnsOut.AddNew
571|    dnsOut("DataType") = GridOutDesc.DataType
572|    dnsOut("DataSize") = GridOutDesc.DataSize
573|    dnsOut("X_Origin") = GridOutDesc.X_Origin
574|    dnsOut("Y_Origin") = GridOutDesc.Y_Origin
575|    dnsOut("DeltaX") = GridOutDesc.DeltaX
576|    dnsOut("DeltaY") = GridOutDesc.DeltaY
577|    dnsOut("nColsX") = GridOutDesc.nColsX
578|    dnsOut("nRowsY") = GridOutDesc.nRowsY
579|    dnsOut("BlankVal") = GridOutDesc.BlankVal
580|    dnsOut("GridName") = GridName
581|    dnsOut("DateModified") = Now
582|      Dim SourceDescription As String
583|      SourceDescription = "Created by GRIDHNDL.BAS:PolygonQTToGrid " & Now &
    CRLF
584|      SourceDescription = SourceDescription & "Source Query or Table was [" &
    strSQLSource & "]  in Database " & DB.Name & CRLF
585|      SourceDescription = SourceDescription & "Source field for polygon outlines was [" &
    strOutlineField & "] ." & CRLF
586|      SourceDescription = SourceDescription & "Values are those from [" & strValueField &
    "] ."
587|    dnsOut("GridComesFrom") = SourceDescription
588|    PullGlobalMemBlockIntoDB hGMemGrid, dnsOut, "GridData"
589|    dnsOut.Update
590|    DummyLong = GlobalFree(hGMemGrid)
591| Exit Sub
592| ERRPolygonQTToGrid:
593|  Select Case Err
594|   Case 3011 'No table errors
595|    ZDefineGridTable DB
596|    Resume
597|   Case Else
598|    MsgBox "Error #" & Err & " occurred in PolygonQTToGrid.  " & CRLF & Error
599|    Err = 0
600|    Exit Sub
601|    Resume
602|  End Select
603|
604| End Sub
605|
606| Sub PullGlobalMemBlockIntoDB (hGMemSource As Integer, dnsToGetIt As Dynaset,
    fldToGetIt As String)
607|      Dim NumChunks As Long, TotalSize As Long, wSel As Integer, DummyLong As Long
```

```
    Dim RemChunk As Integer, Cursize As Integer
    Dim Offset As Long
    Dim IX As Long, ChunkSize As Long, CurChunk As String
    ChunkSize = 8192 ' Set size of chunk.
    ' Get file size.
    TotalSize = GlobalSize(hGMemSource)
    wSel = GlobalHandleToSel(hGMemSource)
    NumChunks = TotalSize \ ChunkSize   ' Set number of chunks.
    ' Set number of remaining bytes.
    RemChunk = TotalSize Mod ChunkSize
    ' Set starting size of chunk.
    Cursize = ChunkSize
    Offset = 0
    For IX = 0 To NumChunks
       If IX = NumChunks Then Cursize = RemChunk
       If Cursize = 0 Then Exit For
       CurChunk = Space$(Cursize)
       DummyLong = MemoryRead(wSel, Offset, ByVal CurChunk, Cursize)
       dnsToGetIt(fldToGetIt).AppendChunk (CurChunk)
       Offset = Offset + Cursize
    Next IX

End Sub

Sub PullLargeFileIntoDB (FileIn As String, dnsToGetIt As Dynaset, fldToGetIt As String)
    Dim NumChunks As Long, TotalSize As Long, FNum As Integer
    Dim RemChunk As Integer, Cursize As Integer
    Dim IX As Long, ChunkSize As Long, CurChunk As String
    ChunkSize = 8192 ' Set size of chunk.
    ' Get file size.
    FNum = FreeFile
    Open FileIn For Binary As #FNum  ' Open the file.
    TotalSize = LOF(FNum)
    NumChunks = TotalSize \ ChunkSize   ' Set number of chunks.
    ' Set number of remaining bytes.
    RemChunk = TotalSize Mod ChunkSize
    ' Set starting size of chunk.
    Cursize = ChunkSize
    For IX = 0 To NumChunks
       If IX = NumChunks Then Cursize = RemChunk
       If Cursize = 0 Then Exit For
       CurChunk = Space$(Cursize)
       Get #FNum, , CurChunk   ' Read chunk from file.
       dnsToGetIt(fldToGetIt).AppendChunk (CurChunk)
    Next IX

End Sub

Sub SaveRasterGridFromGlobalMem (GridName As String, GridOutDesc As
    typGridDescription, DB As Database, Comments As String)
On Error GoTo ERRSaveRasterGridFromGlobalMem

Dim dnsOut As Dynaset
Set dnsOut = DB.CreateDynaset("zWHEATtblRasterGrids")
```

```
661|    dnsOut.AddNew
662|    dnsOut("DataType") = GridOutDesc.DataType
663|    dnsOut("DataSize") = GridOutDesc.DataSize
664|    dnsOut("X_Origin") = GridOutDesc.X_Origin
665|    dnsOut("Y_Origin") = GridOutDesc.Y_Origin
666|    dnsOut("DeltaX") = GridOutDesc.DeltaX
667|    dnsOut("DeltaY") = GridOutDesc.DeltaY
668|    dnsOut("nColsX") = GridOutDesc.nColsX
669|    dnsOut("nRowsY") = GridOutDesc.nRowsY
670|    dnsOut("BlankVal") = GridOutDesc.BlankVal
671|    dnsOut("GridName") = GridName
672|    dnsOut("DateModified") = Now
673|    dnsOut("GridComesFrom") = Comments
674|    PullGlobalMemBlockIntoDB GridOutDesc.hGMem, dnsOut, "GridData"
675|    dnsOut.Update
676|
677|
678|
679|
680|  Exit Sub
681|  ERRSaveRasterGridFromGlobalMem:
682|   Select Case Err
683|    Case 3011 'No table errors
684|     ZDefineGridTable DB
685|     Resume
686|    Case Else
687|     MsgBox "Error #" & Err & " occurred in SaveRasterGridFromGlobalMem.  " & CRLF &
      Error
688|     Err = 0
689|     Exit Sub
690|     Resume
691|   End Select
692|
693|
694|  End Sub
695|
696|  Sub SetIJCellGenericInGlobalMem (IX As Long, JY As Long, GridDesc As
      typGridDescription, hGMem As Integer, Value As Variant)
697|  If IX < 0 Or IX >= GridDesc.nColsX Then Exit Sub
698|  If JY < 0 Or JY >= GridDesc.nRowsY Then Exit Sub
699|
700|
701|   Select Case GridDesc.DataType
702|     Case DB_BOOLEAN, DB_BYTE, DB_INTEGER
703|        SetIJCellInt2InGlobalMem IX, JY, GridDesc, hGMem, CInt(Value)
704|        Exit Sub
705|     Case DB_LONG
706|        SetIJCellInt4InGlobalMem IX, JY, GridDesc, hGMem, CLng(Value)
707|        Exit Sub
708|     Case DB_CURRENCY
709|        SetIJCellInt8InGlobalMem IX, JY, GridDesc, hGMem, CCur(Value)
710|        Exit Sub
711|     Case DB_SINGLE
712|        SetIJCellReal4InGlobalMem IX, JY, GridDesc, hGMem, CSng(Value)
```

```
713|       Exit Sub
714|    Case DB_DOUBLE
715|       SetIJCellReal8InGlobalMem IX, JY, GridDesc, hGMem, CDbl(Value)
716|       Exit Sub
717|    Case DB_DATE
718|       Beep
719|       Exit Sub
720|    Case DB_TEXT
721|       SetIJCellTXTInGlobalMem IX, JY, GridDesc, hGMem, CStr(Value)
722|       Exit Sub
723|    Case Else
724|       Beep
725|       Exit Sub
726|  End Select
727|
728| End Sub
729|
730| Sub SetIJCellInt2InGlobalMem (IX As Long, JY As Long, GridDesc As
    typGridDescription, hGMem As Integer, Value As Integer)
731| Static Offset As Long, wSel  As Integer, DummyLong As Long
732| If GridDesc.DataSize <> 2 Then
733|    Beep
734|    MsgBox "Cannot put a 2-byte integer into a grid with datasize of " &
    GridDesc.DataSize
735|    Exit Sub
736| End If
737| If IX < 0 Or IX >= GridDesc.nColsX Then Exit Sub
738| If JY < 0 Or JY >= GridDesc.nRowsY Then Exit Sub
739|
740|    Offset = (JY * GridDesc.nColsX + IX) * GridDesc.DataSize
741| ' IX could put a function in here to check the size of the global memory block
742| '  before trying to do this, but it would slow it down to much and ther
743| '  should be other bounds checking before getting here.
744|    wSel = GlobalHandleToSel(hGMem)
745|    DummyLong = MemoryWrite(wSel, Offset, Value, GridDesc.DataSize)
746| End Sub
747|
748| Sub SetIJCellInt4InGlobalMem (IX As Long, JY As Long, GridDesc As
    typGridDescription, hGMem As Integer, Value As Long)
749| Static Offset As Long, wSel  As Integer, DummyLong As Long
750| If GridDesc.DataSize <> 4 Then
751|    Beep
752|    MsgBox "Cannot put a 4-byte integer into a grid with datasize of " &
    GridDesc.DataSize
753|    Exit Sub
754| End If
755| If IX < 0 Or IX >= GridDesc.nColsX Then Exit Sub
756| If JY < 0 Or JY >= GridDesc.nRowsY Then Exit Sub
757|
758|    Offset = (JY * GridDesc.nColsX + IX) * GridDesc.DataSize
759|    If Offset < 0 Then Exit Sub
760| ' IX could put a function in here to check the size of the global memory block
761| '  before trying to do this, but it would slow it down to much and ther
762| '  should be other bounds checking before getting here.
```

```
763|     wSel = GlobalHandleToSel(hGMem)
764|     DummyLong = MemoryWrite(wSel, Offset, Value, GridDesc.DataSize)
765|
766| End Sub
767|
768| Sub SetIJCellInt8InGlobalMem (IX As Long, JY As Long, GridDesc As
    typGridDescription, hGMem As Integer, Value As Currency)
769| Static Offset As Long, wSel  As Integer, DummyLong As Long
770| If GridDesc.DataSize <> 8 Then
771|     Beep
772|     MsgBox "Cannot put a 8-byte integer into a grid with datasize of " &
    GridDesc.DataSize
773|     Exit Sub
774| End If
775| If IX < 0 Or IX >= GridDesc.nColsX Then Exit Sub
776| If JY < 0 Or JY >= GridDesc.nRowsY Then Exit Sub
777|
778|     Offset = (JY * GridDesc.nColsX + IX) * GridDesc.DataSize
779| ' IX could put a function in here to check the size of the global memory block
780| '  before trying to do this, but it would slow it down to much and ther
781| '  should be other bounds checking before getting here.
782|     wSel = GlobalHandleToSel(hGMem)
783|     DummyLong = MemoryWrite(wSel, Offset, Value, GridDesc.DataSize)
784|
785| End Sub
786|
787| Sub SetIJCellReal4InGlobalMem (IX As Long, JY As Long, GridDesc As
    typGridDescription, hGMem As Integer, Value As Single)
788| Static Offset As Long, wSel  As Integer, DummyLong As Long
789| If GridDesc.DataSize <> 4 Then
790|     Beep
791|     MsgBox "Cannot put a 4-byte real into a grid with datasize of " & GridDesc.DataSize
792|     Exit Sub
793| End If
794| If IX < 0 Or IX >= GridDesc.nColsX Then Exit Sub
795| If JY < 0 Or JY >= GridDesc.nRowsY Then Exit Sub
796|
797|     Offset = (JY * GridDesc.nColsX + IX) * GridDesc.DataSize
798| ' IX could put a function in here to check the size of the global memory block
799| '  before trying to do this, but it would slow it down to much and ther
800| '  should be other bounds checking before getting here.
801|     wSel = GlobalHandleToSel(hGMem)
802|     DummyLong = MemoryWrite(wSel, Offset, Value, GridDesc.DataSize)
803|
804| End Sub
805|
806| Sub SetIJCellReal8InGlobalMem (IX As Long, JY As Long, GridDesc As
    typGridDescription, hGMem As Integer, Value As Double)
807| Static Offset As Long, wSel  As Integer, DummyLong As Long
808| If GridDesc.DataSize <> 8 Then
809|     Beep
810|     MsgBox "Cannot put a 8-byte real into a grid with datasize of " & GridDesc.DataSize
811|     Exit Sub
812| End If
```

```
813| If IX < 0 Or IX >= GridDesc.nColsX Then Exit Sub
814| If JY < 0 Or JY >= GridDesc.nRowsY Then Exit Sub
815|
816|    Offset = (JY * GridDesc.nColsX + IX) * GridDesc.DataSize
817| ' IX could put a function in here to check the size of the global memory block
818| '  before trying to do this, but it would slow it down to much and ther
819| '  should be other bounds checking before getting here.
820|    wSel = GlobalHandleToSel(hGMem)
821|    DummyLong = MemoryWrite(wSel, Offset, Value, GridDesc.DataSize)
822|
823| End Sub
824|
825| Sub SetIJCellTXTInGlobalMem (IX As Long, JY As Long, GridDesc As
    typGridDescription, hGMem As Integer, Value As String)
826| Static Offset As Long, wSel  As Integer, DummyLong As Long
827| If IX < 0 Or IX >= GridDesc.nColsX Then Exit Sub
828| If JY < 0 Or JY >= GridDesc.nRowsY Then Exit Sub
829|
830|
831|    Offset = (JY * GridDesc.nColsX + IX) * GridDesc.DataSize
832| ' IX could put a function in here to check the size of the global memory block
833| '  before trying to do this, but it would slow it down to much and ther
834| '  should be other bounds checking before getting here.
835|    wSel = GlobalHandleToSel(hGMem)
836|    DummyLong = MemoryWrite(wSel, Offset, ByVal Value, GridDesc.DataSize)
837|
838| End Sub
839|
840| Sub SetRowIntoI4RasterGrid (JY As Long, GridDesc As typGridDescription, DataIn() As
    Long)
841| If JY < 0 Or JY >= GridDesc.nRowsY Then Exit Sub
842| If GridDesc.DataType <> DB_LONG Then Exit Sub 'Maybe gripe at user, too.
843| If UBound(DataIn) - LBound(DataIn) + 1 <> GridDesc.nColsX Then
844|    'we have a problem in that we don't have a full line of data, or too much data
845|    ' halt it and give user an error message
846|    Beep
847|    MsgBox "Wrong dimensions on input array to SetRowIntoR4RasterGrid"
848|    Exit Sub
849|  End If
850| Dim wSel As Integer, Offset As Long, RowSize As Long
851| Offset = (JY * GridDesc.nColsX) * GridDesc.DataSize
852| wSel = GlobalHandleToSel(GridDesc.hGMem)
853| RowSize = GridDesc.DataSize * GridDesc.nColsX
854| DummyLong = MemoryWrite(wSel, Offset, DataIn(LBound(DataIn)), RowSize)
855|
856| End Sub
857|
858| Sub SetRowIntoR4RasterGrid (JY As Long, GridDesc As typGridDescription, DataIn() As
    Single)
859|
860| If JY < 0 Or JY >= GridDesc.nRowsY Then Exit Sub
861| If GridDesc.DataType <> DB_SINGLE Then Exit Sub 'Maybe gripe at user, too.
862| If UBound(DataIn) - LBound(DataIn) + 1 <> GridDesc.nColsX Then
863|    'we have a problem in that we don't have a full line of data, or too much data
```

```
864|     ' halt it and give user an error message
865|     Beep
866|     MsgBox "Wrong dimensions on input array to SetRowIntoR4RasterGrid"
867|     Exit Sub
868|   End If
869|  Dim wSel As Integer, Offset As Long, RowSize As Long
870|  Offset = (JY * GridDesc.nColsX) * GridDesc.DataSize
871|  wSel = GlobalHandleToSel(GridDesc.hGMem)
872|
873|  RowSize = GridDesc.DataSize * GridDesc.nColsX
874|  DummyLong = MemoryWrite(wSel, Offset, DataIn(LBound(DataIn)), RowSize)
875|
876| End Sub
877|
878| Sub SetXYCellGenericInGlobalMem (X As Single, Y As Single, GridDesc As
     typGridDescription, hGMem As Integer, Value As Variant)
879|  Select Case GridDesc.DataType
880|    Case DB_BOOLEAN, DB_BYTE, DB_INTEGER
881|       SetXYCellInt2InGlobalMem X, Y, GridDesc, hGMem, CInt(Value)
882|       Exit Sub
883|    Case DB_LONG
884|       SetXYCellInt4InGlobalMem X, Y, GridDesc, hGMem, CLng(Value)
885|       Exit Sub
886|    Case DB_CURRENCY
887|       SetXYCellInt8InGlobalMem X, Y, GridDesc, hGMem, CCur(Value)
888|       Exit Sub
889|    Case DB_SINGLE
890|       SetXYCellReal4InGlobalMem X, Y, GridDesc, hGMem, CSng(Value)
891|       Exit Sub
892|    Case DB_DOUBLE
893|       SetXYCellReal8InGlobalMem X, Y, GridDesc, hGMem, CDbl(Value)
894|       Exit Sub
895|    Case DB_DATE
896|       Beep
897|       Exit Sub
898|    Case DB_TEXT
899|       SetXYCellTXTInGlobalMem X, Y, GridDesc, hGMem, CStr(Value)
900|       Exit Sub
901|    Case DB_LONGBINARY, DB_MEMO
902|       Beep
903|    Case Else
904|       Beep
905|  End Select
906| End Sub
907|
908| Sub SetXYCellInt1InGlobalMem (X As Single, Y As Single, GridDesc As
     typGridDescription, hGMem As Integer, Value As String)
909| Static Offset As Long, wSel  As Integer, DummyLong As Long
910| If GridDesc.DataSize <> 1 Then
911|    Beep
912|    MsgBox "Cannot put a 1-byte integer into a grid with datasize of " &
     GridDesc.DataSize
913|    Exit Sub
914| End If
```

```
 915|
 916| Offset = CellOffsetFromXY(X, Y, GridDesc)
 917| If Offset < 0 Then Exit Sub
 918| ' IX could put a function in here to check the size of the global memory block
 919| '  before trying to do this, but it would slow it down to much and ther
 920| '  should be other bounds checking before getting here.
 921| '
 922| 'If dnsIn(FieldName).FieldSize() < Offset Then
 923| '   'There is not as much data as there ought to be for this data type.
 924| '   Beep
 925| '   Exit Function
 926| 'End If
 927|
 928| wSel = GlobalHandleToSel(hGMem)
 929|
 930| DummyLong = MemoryWrite(wSel, Offset, Value, GridDesc.DataSize)
 931|
 932|
 933|
 934| End Sub
 935|
 936| Sub SetXYCellInt2InGlobalMem (X As Single, Y As Single, GridDesc As
      typGridDescription, hGMem As Integer, Value As Integer)
 937| Static Offset As Long, wSel  As Integer, DummyLong As Long
 938| If GridDesc.DataSize <> 2 Then
 939|    Beep
 940|    MsgBox "Cannot put a 2-byte integer into a grid with datasize of " &
      GridDesc.DataSize
 941|    Exit Sub
 942| End If
 943|
 944| Offset = CellOffsetFromXY(X, Y, GridDesc)
 945| If Offset < 0 Then Exit Sub
 946| ' IX could put a function in here to check the size of the global memory block
 947| '  before trying to do this, but it would slow it down to much and ther
 948| '  should be other bounds checking before getting here.
 949| '
 950| 'If dnsIn(FieldName).FieldSize() < Offset Then
 951| '   'There is not as much data as there ought to be for this data type.
 952| '   Beep
 953| '   Exit Function
 954| 'End If
 955|
 956| wSel = GlobalHandleToSel(hGMem)
 957|
 958| DummyLong = MemoryWrite(wSel, Offset, Value, GridDesc.DataSize)
 959|
 960|
 961| End Sub
 962|
 963| Sub SetXYCellInt4InGlobalMem (X As Single, Y As Single, GridDesc As
      typGridDescription, hGMem As Integer, Value As Long)
 964| Static Offset As Long, wSel  As Integer, DummyLong As Long
 965| If GridDesc.DataSize <> 4 Then
```

```
 966|    Beep
 967|    MsgBox "Cannot put a 4-byte integer into a grid with datasize of " &
      GridDesc.DataSize
 968|    Exit Sub
 969| End If
 970|
 971| Offset = CellOffsetFromXY(X, Y, GridDesc)
 972| If Offset < 0 Then Exit Sub
 973| ' IX could put a function in here to check the size of the global memory block
 974|
 975| ' before trying to do this, but it would slow it down to much and ther
 976| ' should be other bounds checking before getting here.
 977| '
 978| 'If dnsIn(FieldName).FieldSize() < Offset Then
 979| '   'There is not as much data as there ought to be for this data type.
 980| '   Beep
 981| '   Exit Function
 982| 'End If
 983|
 984| wSel = GlobalHandleToSel(hGMem)
 985|
 986| DummyLong = MemoryWrite(wSel, Offset, Value, GridDesc.DataSize)
 987|
 988|
 989| End Sub
 990|
 991| Sub SetXYCellInt8InGlobalMem (X As Single, Y As Single, GridDesc As
      typGridDescription, hGMem As Integer, Value As Currency)
 992| Static Offset As Long, wSel  As Integer, DummyLong As Long
 993| If GridDesc.DataSize <> 8 Then
 994|    Beep
 995|    MsgBox "Cannot put a 8-byte integer into a grid with datasize of " &
      GridDesc.DataSize
 996|    Exit Sub
 997| End If
 998|
 999| Offset = CellOffsetFromXY(X, Y, GridDesc)
1000| If Offset < 0 Then Exit Sub
1001| ' IX could put a function in here to check the size of the global memory block
1002| ' before trying to do this, but it would slow it down to much and ther
1003| ' should be other bounds checking before getting here.
1004| '
1005| 'If dnsIn(FieldName).FieldSize() < Offset Then
1006| '   'There is not as much data as there ought to be for this data type.
1007| '   Beep
1008| '   Exit Function
1009| 'End If
1010|
1011| wSel = GlobalHandleToSel(hGMem)
1012|
1013| DummyLong = MemoryWrite(wSel, Offset, Value, GridDesc.DataSize)
1014|
1015|
1016|
```

```
1017  End Sub
1018
1019  Sub SetXYCellReal4InGlobalMem (X As Single, Y As Single, GridDesc As
      typGridDescription, hGMem As Integer, Value As Single)
1020  Static Offset As Long, wSel  As Integer, DummyLong As Long
1021  If GridDesc.DataSize <> 4 Then
1022     Beep
1023     MsgBox "Cannot put a 4-byte real into a grid with datasize of " & GridDesc.DataSize
1024     Exit Sub
1025  End If
1026
1027  Offset = CellOffsetFromXY(X, Y, GridDesc)
1028  If Offset < 0 Then Exit Sub
1029  ' IX could put a function in here to check the size of the global memory block
1030  '  before trying to do this, but it would slow it down to much and ther
1031  '  should be other bounds checking before getting here.
1032  '
1033  'If dnsIn(FieldName).FieldSize() < Offset Then
1034  '    'There is not as much data as there ought to be for this data type.
1035  '    Beep
1036  '    Exit Function
1037  'End If
1038
1039  wSel = GlobalHandleToSel(hGMem)
1040
1041  DummyLong = MemoryWrite(wSel, Offset, Value, GridDesc.DataSize)
1042
1043
1044
1045  End Sub
1046
1047  Sub SetXYCellReal8InGlobalMem (X As Single, Y As Single, GridDesc As
      typGridDescription, hGMem As Integer, Value As Double)
1048  Static Offset As Long, wSel  As Integer, DummyLong As Long
1049  If GridDesc.DataSize <> 8 Then
1050     Beep
1051     MsgBox "Cannot put a 8-byte real into a grid with datasize of " & GridDesc.DataSize
1052     Exit Sub
1053  End If
1054
1055  Offset = CellOffsetFromXY(X, Y, GridDesc)
1056  If Offset < 0 Then Exit Sub
1057  ' IX could put a function in here to check the size of the global memory block
1058  '  before trying to do this, but it would slow it down to much and ther
1059  '  should be other bounds checking before getting here.
1060  '
1061  'If dnsIn(FieldName).FieldSize() < Offset Then
1062  '    'There is not as much data as there ought to be for this data type.
1063  '    Beep
1064  '    Exit Function
1065  'End If
1066
1067  wSel = GlobalHandleToSel(hGMem)
1068
```

```
1069|  DummyLong = MemoryWrite(wSel, Offset, Value, GridDesc.DataSize)
1070|
1071|  End Sub
1072|
1073|  Sub SetXYCellTXTInGlobalMem (X As Single, Y As Single, GridDesc As
       typGridDescription, hGMem As Integer, Value As String)
1074|  Static Offset As Long, wSel  As Integer, DummyLong As Long
1075|
1076|  Offset = CellOffsetFromXY(X, Y, GridDesc)
1077|  If Offset < 0 Then Exit Sub
1078|  ' IX could put a function in here to check the size of the global memory block
1079|  '  before trying to do this, but it would slow it down to much and ther
1080|  '  should be other bounds checking before getting here.
1081|  '
1082|  'If dnsIn(FieldName).FieldSize() < Offset Then
1083|  '    'There is not as much data as there ought to be for this data type.
1084|  '    Beep
1085|  '    Exit Function
1086|  'End If
1087|
1088|  wSel = GlobalHandleToSel(hGMem)
1089|
1090|  DummyLong = MemoryWrite(wSel, Offset, ByVal Value, GridDesc.DataSize)
1091|
1092|
1093|
1094|
1095|  End Sub
1096|
1097|  Sub ShowProblemPolygon (InPoly() As RealPoint, ClippedLines() As RealPoint,
       nCrossings As Long)
1098|  ' Dim n As Long, BoundBox  As RealRect
1099|  'Dim nPoly As Long, lbPoly As Long
1100|  'lbPoly = LBound(InPoly)
1101|  'nPoly = UBound(InPoly) - lbPoly + 1
1102|  '
1103|  'Call GetPolyBounds(InPoly(lbPoly), nPoly + 1, BoundBox)
1104|  'Call MagnifyRectangle(BoundBox, BoundBox, .1)
1105|  '
1106|  'frmShowPoly.Scale (BoundBox.Left, BoundBox.Top)-(BoundBox.RIght,
       BoundBox.Bottom)
1107|  '
1108|  '  frmShowPoly.Cls
1109|  '  frmShowPoly.Show
1110|  '  frmShowPoly.DrawWidth = 1
1111|  '  frmShowPoly.ForeColor = RGB(255, 0, 0)
1112|  '  For n = LBound(InPoly) To UBound(InPoly) - 1
1113|  '    frmShowPoly.Line (InPoly(n).X, InPoly(n).Y)-(InPoly(n + 1).X, InPoly(n + 1).Y)
1114|  '  Next n
1115|  '  frmShowPoly.DrawWidth = 2
1116|  '  frmShowPoly.ForeColor = RGB(0, 255, 0)
1117|  '  For n = 1 To nCrossings
1118|  '    frmShowPoly.Circle (ClippedLines(n).X, ClippedLines(n).Y), 30
1119|  '  Next n
```

```
1120|  ' frmShowPoly.CurrentX = BoundBox.Left
1121|  " frmShowPoly.CurrentY = BoundBox.Top
1122|  '  frmShowPoly.DrawWidth = 1
1123|  '  frmShowPoly.ForeColor = RGB(255, 0, 255)
1124|  '  For n = 1 To nCrossings - 1 Step 2
1125|  '    frmShowPoly.Line (ClippedLines(n).X, ClippedLines(n).Y)-(ClippedLines(n + 1).X,
       ClippedLines(n + 1).Y), RGB(255, 0, 255)
1126|  '  Next n
1127|  '
1128|  '  For n = 1 To nCrossings
1129|  '    dDebug.Print "X=" & ClippedLines(n).X & ",  Y=" & ClippedLines(n).Y
1130|  '    frmShowPoly.CurrentY = frmShowPoly.CurrentY + frmShowPoly.TextHeight("123") *
       2
1131|  '  Next n
1132|  '
1133|  Beep
1134|  Beep
1135|  End Sub
1136|
1137|  Function UserChoiceOfRasterGrids (intTypesAllowed() As Integer, DB As Database,
       Prompt As String) As String
1138|  Dim n As Long
1139|  On Error GoTo ERRUserChoiceOfRasterGrids
1140|  Dim snpListGrids As snapshot
1141|  Set snpListGrids = DB.CreateSnapshot("SELECT DISTINCT [GridName], [DataType]
       FROM [zWHEATtblRasterGrids];")
1142|  frmRasterGridList.lblPrompt = Prompt
1143|  frmRasterGridList.lstGrids.Clear
1144|  snpListGrids.MoveFirst
1145|  Do Until snpListGrids.EOF
1146|    For n = LBound(intTypesAllowed) To UBound(intTypesAllowed)
1147|        If snpListGrids("DataType") = intTypesAllowed(n) Then
       frmRasterGridList.lstGrids.AddItem snpListGrids("GridName")
1148|    Next
1149|
1150|    snpListGrids.MoveNext
1151|  Loop 'While Not snpListGrids.EOF
1152|  snpListGrids.Close
1153|  frmRasterGridList.Show 1
1154|  If frmRasterGridList.cmdOK.FontItalic Then
1155|     UserChoiceOfRasterGrids = frmRasterGridList.lstGrids
1156|    Else
1157|     UserChoiceOfRasterGrids = ""
1158|  End If
1159|  Unload frmRasterGridList
1160|  Exit Function
1161|
1162|  ERRUserChoiceOfRasterGrids:
1163|  Select Case Err
1164|     Case Else
1165|      MsgBox "Error #" & Err & " occurred in UserChoiceOfRasterGrids." & CRLF & Error
1166|      Err = 0
1167|      Exit Function
1168|      Resume
```

```
1169| End Select
1170|
1171| End Function
1172|
1173| Sub WriteLargeFileFromBLOB (FileOut As String, dnsFrom As Dynaset, fldFrom As
     String)
1174|   Dim NumChunks As Long, TotalSize As Long, Offset As Long
1175|   Dim RemChunk As Long, Cursize As Long
1176|   Dim IX As Long, ChunkSize As Long, CurChunk As String
1177|   Close 92
1178|   Open FileOut For Binary As 92
1179|
1180|
1181|
1182|   ChunkSize = 8192  ' Set size of chunk.
1183|   ' Get field size.
1184|   TotalSize = dnsFrom(fldFrom).FieldSize()
1185|   NumChunks = TotalSize \ ChunkSize   ' Set number of chunks.
1186|   ' Set number of remaining bytes.
1187|   RemChunk = TotalSize Mod ChunkSize
1188|   ' Set starting size of chunk.
1189|   Cursize = ChunkSize
1190|
1191|   For IX = 0 To NumChunks
1192|     If IX = NumChunks Then Cursize = RemChunk
1193|     Offset = IX * ChunkSize
1194|     CurChunk = dnsFrom(fldFrom).GetChunk(Offset, Cursize)
1195|     'Write to disk file
1196|     Put #92, , CurChunk
1197|   Next IX
1198|
1199|  Close 92
1200| End Sub
1201|
1202| Function ZApplyCLUT (ValueIn As Variant, CLUT() As ColorLUT) As Long
1203| Dim nL As Long
1204|
1205| For nL = LBound(CLUT) To UBound(CLUT)
1206|   If (ValueIn < CLUT(nL).Max And ValueIn >= CLUT(nL).Min) Then
1207|   'dDebug.Print nL, ValueIn, CLUT(nL).Max, CLUT(nL).Min
1208|     ZApplyCLUT = CLUT(nL).VBColor
1209|     Exit For
1210|   End If
1211| Next nL
1212|
1213| End Function
1214|
1215| Sub ZDefineGridTable (DB As Database)
1216|
1217| Dim ntdfGrids As New tabledef
1218| Dim nfldDT As New field, nfldDS As New field
1219| Dim nfldX0 As New field, nfldnC As New field, nflddX As New field
1220| Dim nfldY0 As New field, nfldnR As New field, nflddY As New field
1221| Dim nfldGridData As New field, nfldName As New field, nfldNullVal As New field
```

```
1222| Dim nfldSerialNum As New field, nfldDate As New field
1223| ntdfGrids.Name = "zWHEATtblRasterGrids"
1224|
1225| 'fields.append fields to tabledef.
1226| nfldName.Name = "GridName": nfldName.Type = DB_TEXT: nfldName.Size = 80:
      ntdfGrids.Fields.Append nfldName
1227| nfldDT.Name = "DataType": nfldDT.Type = DB_LONG: ntdfGrids.Fields.Append nfldDT
1228| nfldDS.Name = "DataSize": nfldDS.Type = DB_LONG: ntdfGrids.Fields.Append nfldDS
1229| nfldX0.Name = "X_Origin": nfldX0.Type = DB_SINGLE: ntdfGrids.Fields.Append nfldX0
1230| nfldY0.Name = "Y_Origin": nfldY0.Type = DB_SINGLE: ntdfGrids.Fields.Append nfldY0
1231| nflddX.Name = "DeltaX": nflddX.Type = DB_SINGLE: ntdfGrids.Fields.Append nflddX
1232| nflddY.Name = "DeltaY": nflddY.Type = DB_SINGLE: ntdfGrids.Fields.Append nflddY
1233| nfldnC.Name = "nColsX": nfldnC.Type = DB_LONG: ntdfGrids.Fields.Append nfldnC
1234| nfldnR.Name = "nRowsY": nfldnR.Type = DB_LONG: ntdfGrids.Fields.Append nfldnR
1235| nfldNullVal.Name = "BlankVal": nfldNullVal.Type = DB_TEXT: nfldNullVal.Size = 8:
      ntdfGrids.Fields.Append nfldNullVal
1236|
1237| nfldSerialNum.Name = "SerialNumber": nfldSerialNum.Type = DB_LONG:
      ntdfGrids.Fields.Append nfldSerialNum
1238| nfldDate.Name = "DateModified": nfldDate.Type = DB_DATE:  ntdfGrids.Fields.Append
      nfldDate
1239| nfldGridData.Name = "GridData": nfldGridData.Type = DB_LONGBINARY:
      ntdfGrids.Fields.Append nfldGridData
1240|
1241| Dim nfSourceComments As New field, nfUserComments As New field
1242| nfSourceComments.Name = "GridComesFrom": nfSourceComments.Type =
      DB_MEMO: ntdfGrids.Fields.Append nfSourceComments
1243| nfUserComments.Name = "UserNotes": nfUserComments.Type = DB_MEMO:
      ntdfGrids.Fields.Append nfUserComments
1244|
1245| DB.TableDefs.Append ntdfGrids
1246| End Sub
1247|
1248| Private Sub ZLoadBLOB2RealPoints (BLOBin As String, DataPts() As RealPoint)
1249| Static S8 As stringg8, RlPt As RealPoint
1250| Dim n As Long, nPoints As Long, NPos As Long
1251| nPoints = Len(BLOBin) / 8 '8 Bytes to a RealPoint
1252|     'This implicitly updates MinX,MaxX,MinY, MaxY
1253|
1254| NPos = 1  'NPos will keep track of where in the BLOB we are
1255| For n = 1 To nPoints
1256|    S8.f = Mid$(BLOBin, NPos, 8)
1257|    LSet RlPt = S8
1258|    DataPts(n) = RlPt
1259|    'MinX = MinR4(MinX, RlPt.X)
1260|    'MinY = MinR4(MinY, RlPt.Y)
1261|    'MaxX = MaxR4(MaxX, RlPt.X)
1262|    'MaxY = MaxR4(MaxY, RlPt.Y)
1263|
1264|    NPos = NPos + 8
1265| Next n
1266|
1267| End Sub
1268|
```

```
1269| Sub ZLoadBLOBIntoGlobalMem (hGMemTarget As Integer, dnsToGetFrom As Dynaset,
      fldToGetFrom As String)
1270|    Dim NumChunks As Long, TotalSize As Long, wSel As Integer, DummyLong As Long
1271|    Dim RemChunk As Integer, Cursize As Integer
1272|    Dim Offset As Long
1273|    Dim IX As Long, ChunkSize As Long, CurChunk As String
1274|    ChunkSize = 8192 ' Set size of chunk.
1275|    ' Get file size.
1276|    TotalSize = dnsToGetFrom(fldToGetFrom).FieldSize()
1277|    If TotalSize > GlobalSize(hGMemTarget) Then
1278|       MsgBox "The global memory block allocated for loading this grid into is too small."
      & CRLF & "ZLoadBLOBIntoGlobalMem"
1279|       Exit Sub
1280|    End If
1281|    wSel = GlobalHandleToSel(hGMemTarget)
1282|    NumChunks = TotalSize \ ChunkSize   ' Set number of chunks.
1283|    ' Set number of remaining bytes.
1284|    RemChunk = TotalSize Mod ChunkSize
1285|    ' Set starting size of chunk.
1286|    Cursize = ChunkSize
1287|    Offset = 0
1288|    For IX = 0 To NumChunks
1289|       If IX = NumChunks Then Cursize = RemChunk
1290|       If Cursize = 0 Then Exit For
1291|       CurChunk = dnsToGetFrom(fldToGetFrom).GetChunk(Offset, Cursize)
1292|       DummyLong = MemoryWrite(wSel, Offset, ByVal CurChunk, Cursize)
1293|
1294|       Offset = Offset + Cursize
1295|    Next IX
1296|
1297| End Sub
1298|
1299| Sub ZLoadColorLUT (strCLUTName As String, DB As Database, CLUT() As ColorLUT)
1300|    Dim snpLUT As snapshot
1301|    Set snpLUT = DB.CreateSnapshot(strCLUTName)
1302|    snpLUT.MoveFirst
1303|    snpLUT.MoveLast
1304|    Dim n As Long, nLevels As Long
1305|    nLevels = snpLUT.RecordCount
1306|    ReDim CLUT(1 To nLevels)
1307|    snpLUT.MoveFirst
1308|    For n = 1 To nLevels
1309|       CLUT(n).Min = snpLUT("Minimum")
1310|       CLUT(n).Max = snpLUT("Maximum")
1311|       CLUT(n).Class = snpLUT("Class")
1312|       CLUT(n).VBColor = snpLUT("VBColor")
1313|       CLUT(n).GDIColor = snpLUT("GDIColor")
1314|     snpLUT.MoveNext
1315|    Next n
1316|    snpLUT.Close
1317|    Set snpLUT = Nothing
1318| End Sub
1319|
```

```
 1| Option Explicit
 2|
 3| Sub SpatialSearch (X As Single, Y As Single, Theme As typTheme, hGMem As Integer,
    lngSearchType As Long, dynLookin As Dynaset, QSOut() As typTaggedIDNum,
    lngHighlightColor As Long, Target As PictureBox)
 4| Select Case lngSearchType
 5|    Case QS_NONE 'Do nothing
 6|    Case QS_CLOSESTPOINT
 7|       Call srchClosest1(X, Y, hGMem, Theme, dynLookin, QSOut(), lngHighlightColor,
    Target)
 8|    Case QS_NEARCURSOR
 9|    Case QS_RADIUSSEARCH
10|       Call srchWithinRadius(X, Y, hGMem, Theme, dynLookin, QSOut(),
    lngHighlightColor, Target)
11|    Case QS_WITHINBOX
12|    Case QS_WITHINPOLY
13|       Call srchEnclosingPolygons(X, Y, hGMem, Theme, dynLookin, QSOut(),
    lngHighlightColor, Target)
14|    Case QS_WITHINTILE
15|       Call srchEnclosingTile(X, Y, hGMem, Theme, dynLookin, QSOut(),
    lngHighlightColor, Target)
16|    Case QS_LINEDIST
17|       Call srchClosestLine(X, Y, hGMem, Theme, dynLookin, QSOut(), lngHighlightColor,
    Target)
18| End Select
19| If dynLookin.Filter = "" Then dynLookin.Filter = "1=0"   '"ZQJLink='~!@#$%^&*()_+'"
20| End Sub
21|
22| Sub srchClosest1 (X As Single, Y As Single, hGMem As Integer, Theme As typTheme,
    dynIn As Dynaset, QSOut() As typTaggedIDNum, lngHighlightColor As Long, Target As
    PictureBox)
23|  Dim nStart As Long, nEnd As Long
24|  Dim m1 As Long, m2 As Long
25|  Dim MinDist As Single, Dist As Single, ClosestPt As RealPoint
26|  Dim N As Long, nClosest As Long, M  As Long, L As Long
27|  Dim nPoints As Long, nChunks As Long, nCh As Long, nPts As Long
28|  Dim QPoint As RealPoint
29|  QPoint.X = X
30|  QPoint.Y = Y
31|  'Find closest will always return exactly one feature, so redimension the
32|  ' array accordingly.
33|  ReDim Preserve QSOut(0 To 1)
34| Select Case Theme.ThemeType
35|  Case SkipThis
36|    'Exit Subroutine automatically
37|  Case OnePoint
38|     nStart = Points(Theme.FirstLoc).nStart
39|     nEnd = Points(Theme.LastLoc).nStart
40|     GoSub FindClosestPointLabel
41|     For m1 = Theme.FirstLoc To Theme.LastLoc
42|      If Points(m1).nStart = nClosest Then
43|         dynIn.Filter = "zqFeatureID = " & Points(m1).FeatureID
44|         QSOut(1).IDNum = m1
45|         QSOut(1).FeatureID = Points(m1).FeatureID
```

```
46|          Exit For
47|        End If
48|       Next m1
49|       'Target.Circle (ClosestPt.X, ClosestPt.Y), 500, lngHighlightColor
50|
51|    Case JointedLine
52|       nStart = Lines(Theme.FirstLoc).nStart
53|       nEnd = Lines(Theme.LastLoc).nStart + Lines(Theme.LastLoc).nPoints
54|       GoSub FindClosestPointLabel
55|       nClosest = nClosest
56|       For m1 = Theme.FirstLoc To Theme.LastLoc
57|        If Lines(m1).nStart <= nClosest And Lines(m1).nStart + Lines(m1).nPoints >
    nClosest Then
58|           dynIn.Filter = "zqFeatureID = " & Lines(m1).FeatureID
59|           QSOut(1).IDNum = m1
60|           QSOut(1).FeatureID = Lines(m1).FeatureID
61|           Exit For
62|        End If
63|       Next m1
64|       """frmMap.picMap.Circle (ClosestPt.X, ClosestPt.Y), 1500, RGB(255, 0, 0)
65|       'Target.Circle (ClosestPt.X, ClosestPt.Y), 500, lngHighlightColor
66|    Case OutLine
67|
68|    Case BoundBlock
69|
70|    Case HatchLine
71|    Case TileBlock
72|       nStart = Tiles(Theme.FirstLoc).nStart
73|       nEnd = Tiles(Theme.LastLoc).nStart + 1
74|       GoSub FindClosestPointLabel
75|       For m1 = Theme.FirstLoc To Theme.LastLoc
76|        If Tiles(m1).nStart = nClosest Or Tiles(m1).nStart + 1 = nClosest Then
77|           dynIn.Filter = "ZQFeatureID = " & Tiles(m1).FeatureID
78|           QSOut(1).IDNum = m1
79|           QSOut(1).FeatureID = Tiles(m1).FeatureID
80|           Exit For
81|        End If
82|       Next m1
83|
84|
85|  '  Case NodePoint
86|  '  Case LabelText
87|  '  Case CCCircle
88|  '  Case EEElipse
89|  '  Case PPPixel
90|  '  Case AAArc
91|  '  Case RRRectangle
92|     Case Else
93|       MsgBox ("Need to add appropriate logic to subroutine SpatialSearch for
    ThemeType=" & Format(Theme.ThemeType))
94|  End Select
95|  Exit Sub
96|  FindClosestPointLabel:
97|
```

```
 98| Dim ptrRealPoints As Long, IndexClosest As Long
 99|
100| nPts = nEnd - nStart + 1
101| ptrRealPoints = GlobalLock(hGMem)
102| If ptrRealPoints = 0 Then MsgBox "Could not get global memory block.  End-users
     should never see this message", , "WARNING! WARNING! DANGER, WIL ROBINSON,
     DANGER! WARNING!!"
103|
104| MinDist = 1E+34
105| 'Sub HUGEFindClosestPoint (TestPoint As RealPoint, ByVal ptrToRealPoints As Long,
106| '                          nPtsTotal As Long, nStartAt As Long, nPointsToLookAt As
     Long,
107| '                                                IndexClosest As Long, Dist2Closest As
     Single, PointClosest As RealPoint)
108| Dim nFirstIndex As Long
109| nFirstIndex = nStart + 1
110| Call HUGEFindClosestPoint(QPoint, ptrRealPoints, hGMemLastPoint, nFirstIndex, nPts,
     IndexClosest, MinDist, ClosestPt)
111| N = GlobalUnlock(hGMem)
112| nClosest = nStart + IndexClosest - 1
113|
114| QSOut(1).Dist1 = MinDist
115| QSOut(1).X = ClosestPt.X
116| QSOut(1).Y = ClosestPt.Y
117|  'Target.Circle (ClosestPt.X, ClosestPt.Y), 500, lngHighlightColor
118| Return
119|
120|
121| End Sub
122|
123| Sub srchClosestLine (X As Single, Y As Single, hGMem As Integer, Theme As
     typTheme, dynIn As Dynaset, QSOut() As typTaggedIDNum, lngHighlightColor As Long,
     Target As PictureBox)
124| On Error GoTo ERRsrchClosestLine
125|  Dim nStart As Long, nEnd As Long, m1 As Long, nClosest As Long, nBase As Long
126|  Dim ptrRealPoints As Long, N As Long, strFilter As String
127|  Dim MinDist As Single, Dist As Single
128|  'Find closest will always return exactly one feature, so redimension the
129|  ' array accordingly.
130|  ReDim Preserve QSOut(0 To 1)
131|  Dim QPoint As RealPoint
132|  QPoint.X = X              'Copy search point into structure.
133|  QPoint.Y = Y
134|
135|  MinDist = 1E+34           'Set MinDist to absurdly high value.
136|  nBase = GlobalLock(hGMem)   'Lock the global array of points down
137|  If nBase = 0 Then MsgBox "Could not get global memory block.  End-users should
     never see this message", , "WARNING! WARNING! DANGER, WIL ROBINSON,
     DANGER! WARNING!!"
138|   'The above line is for panicking if the global memory lock fails, and we're in deep
     trouble.
139|   Dim nPTotal As Long
140|   Dim P1 As RealPoint, P2 As RealPoint
141| Select Case Theme.ThemeType
```

```
142|    Case SkipThis            'Exit Subroutine automatically
143|    Case OnePoint             'Exit Subroutine automatically
144|    Case JointedLine
145|     Dim nLines As Long
146|     nLines = Theme.LastLoc - Theme.FirstLoc + 1
147|     frmMap.lblPercentDone = "Searching Lines"
148|     frmMap.pnl3dPercentDone.FloodPercent = 0
149|       For m1 = Theme.FirstLoc To Theme.LastLoc
150|        frmMap.pnl3dPercentDone.FloodPercent = (m1 - Theme.FirstLoc) / nLines * 100
151|        Call HUGEDist2JointedLine(QPoint, nBase, hGMemLastPoint, Lines(m1).nStart,
    Lines(m1).nPoints, nClosest, Dist)
152|
153|          If Dist < MinDist Then
154|            MinDist = Dist
155|            strFilter = "zqFeatureID = " & Lines(m1).FeatureID
156|            QSOut(1).IDNum = m1
157|            QSOut(1).Dist1 = Dist
158|            QSOut(1).FeatureID = Lines(m1).FeatureID
159|            'Maybe later, add the calculations to find the point
160|            ' so I can highlight it.
161|            'QSOut(1).X = ClosestPt.X
162|            'QSOut(1).Y = ClosestPt.Y
163|          End If
164|       Next m1
165|            'Could draw the closest line here with, except that drawline doesn't take a control
    as a parameter.
166|          ' nLine=QSOut(1).IDNum
167|          ' Call DrawLine(Lines(nLine), Lines(nLine).ZWidth, lngQSHighlightColor,
    Lines(nLine).ZStyle)
168|       dynIn.Filter = strFilter
169|
170|
171|    Case SOLIDPOLYGON
172|     Dim nPolys As Long
173|     nPolys = Theme.LastLoc - Theme.FirstLoc + 1
174|     frmMap.lblPercentDone = "Searching Polygons"
175|     frmMap.pnl3dPercentDone.FloodPercent = 0
176|       For m1 = Theme.FirstLoc To Theme.LastLoc
177|        frmMap.pnl3dPercentDone.FloodPercent = (m1 - Theme.FirstLoc) / nPolys * 100
178|        Call HUGEDist2JointedLine(QPoint, nBase, hGMemLastPoint, Polys(m1).nStart,
    Polys(m1).nPoints, nClosest, Dist)
179|
180|          If Dist < MinDist Then
181|            MinDist = Dist
182|            strFilter = "zqFeatureID = " & Polys(m1).FeatureID
183|            QSOut(1).IDNum = m1
184|            QSOut(1).Dist1 = Dist
185|            QSOut(1).FeatureID = Lines(m1).FeatureID
186|            'Maybe later, add the calculations to find the point
187|            ' so I can highlight it.
188|            'QSOut(1).X = ClosestPt.X
189|            'QSOut(1).Y = ClosestPt.Y
190|          End If
191|       Next m1
```

```
192|          'Could draw the closest line here with, except that drawline doesn't take a control
     as a parameter.
193|          ' nLine=QSOut(1).IDNum
194|          ' Call DrawLine(Lines(nLine), Lines(nLine).ZWidth, lngQSHighlightColor,
     Lines(nLine).ZStyle)
195|        dynIn.Filter = strFilter
196|
197|    Case BoundBlock
198|
199|    Case HatchLine
200|    Case TileBlock
201| '  Case NodePoint
202| '  Case LabelText
203| '  Case CCCircle
204| '  Case EEElipse
205| '  Case PPPixel
206| '  Case AAArc
207| '  Case RRRectangle
208|    Case Else
209|      MsgBox ("Need to add appropriate logic to subroutine SpatialSearch for
     ThemeType=" & Format(Theme.ThemeType))
210| End Select
211|    nBase = GlobalUnlock(hGMem)      'Unlock the global memory
212| Exit Sub
213| 'Space for a gosub block, or maybe error handling.
214| ERRsrchClosestLine:
215|  Select Case Err
216|   Case Else
217|    MsgBox "Error #" & Err & " occurred in SPATQUERY.BAS:srchClosestLine." & CRLF
     & Error
218|    Exit Sub
219|    Resume
220|  End Select
221| End Sub
222|
223| Sub srchEnclosingPolygons (X As Single, Y As Single, hGMem As Integer, Theme As
     typTheme, dynIn As Dynaset, QSOut() As typTaggedIDNum, lngHighlightColor As Long,
     Target As PictureBox)
224|  On Error GoTo ERRsrchEnclosingPolygon
225|  Dim strFilter As String
226|  strFilter = "1=0"
227|  Dim N  As Long, nBase As Long
228|  Dim QPoint As RealPoint
229|  QPoint.X = X              'Copy search point into structure.
230|  QPoint.Y = Y
231|  ReDim Preserve QSOut(0 To 0) 'Kill previous output
232|
233|  nBase = GlobalLock(hGMem)   'Lock the global array of points down
234|  If nBase = 0 Then MsgBox "Could not get global memory block.  End-users should
     never see this message", , "WARNING! WARNING! DANGER, WIL ROBINSON,
     DANGER! WARNING!!"
235|   'The above line is for panicking if the global memory lock fails, and we're in deep
     trouble.
236|   Dim nPTotal As Long
```

```
237|   Dim IsItIn  As Integer, nPts As Long, nStarts As Long, nIn As Long
238|    Dim NN As Long
239|
240|
241|    nPolys = Theme.LastLoc - Theme.FirstLoc + 1
242|    frmMap.lblPercentDone = "Searching Polygons"
243|    frmMap.pnl3dPercentDone.FloodPercent = 0
244|
245| Select Case Theme.ThemeType
246|  Case SOLIDPOLYGON
247|      For N = Theme.FirstLoc To Theme.LastLoc
248|        NN = N
249|        frmMap.pnl3dPercentDone.FloodPercent = (NN - Theme.FirstLoc) / nPolys * 100
250|        'Call HUGEDist2JointedLine(QPoint, nBase, hGMemLastPoint, Lines(N).nStart,
    Lines(N).nPoints, nClosest, Dist)
251|         nStarts = Polys(NN).nStart
252|         nPts = Polys(NN).nPoints
253|         IsItIn = HUGEIsPointInPoly(QPoint, nBase, hGMemLastPoint, nStarts, nPts)
254|        'dDebug.Print nBase, hGMemLastPoint, nStarts, nPts
255|        'dDebug.Print NN, N, IsItIn
256|        If IsItIn Then
257|          nIn = nIn + 1
258|
259|          ReDim Preserve QSOut(0 To nIn)
260|            If strFilter = "1=0" Then
261|              strFilter = "(zqFeatureID = " & Polys(NN).FeatureID & ")"
262|              QSOut(nIn).FeatureID = Polys(NN).FeatureID
263|              QSOut(nIn).IDNum = NN
264|             Else
265|              strFilter = strFilter & " or (zqFeatureID = " & Polys(NN).FeatureID & ")"
266|              QSOut(nIn).FeatureID = Polys(NN).FeatureID
267|              QSOut(nIn).IDNum = NN
268|            End If
269|        End If
270|      Next N
271|      dynIn.Filter = strFilter
272|    Case Else
273|      Beep
274| End Select
275|
276| nBase = GlobalUnlock(hGMem)
277|
278|
279| Exit Sub
280| ERRsrchEnclosingPolygon:
281|  Select Case Err
282|    Case Else
283|      MsgBox "Error #" & Err & " occurred in srchEnclosingPolygons." & CRLF & Error
284|      Err = 0
285|      nBase = GlobalUnlock(hGMem)
286|      Exit Sub
287|      Resume
288|  End Select
289| End Sub
```

```
290|
291| Sub srchEnclosingTile (X As Single, Y As Single, hGMem As Integer, Theme As
    typTheme, dynIn As Dynaset, QSOut() As typTaggedIDNum, lngHighlightColor As Long,
    Target As PictureBox)
292|  Dim strFilter As String
293|  strFilter = "1=0"
294|
295| ReDim TileCorners(1 To 2) As RealPoint
296| Dim m1 As Long, nIn As Long
297|   ReDim Preserve QSOut(0 To 0)
298|   'On 940831, GWPouch modified this to allow it to return 0 or many tiles.
299| Select Case Theme.ThemeType
300|   Case TileBlock
301|     For m1 = Theme.FirstLoc To Theme.LastLoc
302|
303|       Call GetRlPntsFromGlobalMem(hGMem, TileCorners(1), Tiles(m1).nStart, 2&)
304|       If (TileCorners(1).X - X) * (TileCorners(2).X - X) <= 0 And (TileCorners(1).Y - Y) *
    (TileCorners(2).Y - Y) <= 0 Then
305|        'we are in this tile
306|        nIn = nIn + 1
307|        ReDim Preserve QSOut(0 To nIn)
308|          If strFilter = "1=0" Then
309|            strFilter = "(zqFeatureID = " & Tiles(m1).FeatureID & ")"
310|            QSOut(nIn).FeatureID = Tiles(m1).FeatureID
311|            QSOut(nIn).IDNum = m1
312|           Else
313|            strFilter = strFilter & " or (zqFeatureID = " & Tiles(m1).FeatureID & ")"
314|            QSOut(nIn).FeatureID = Tiles(m1).FeatureID
315|            QSOut(nIn).IDNum = m1
316|          End If
317|
318|        "dynIn.Filter = "ZQFeatureID = '" & Tiles(m1).FeatureName & "'"
319|        "QSOut(nIn).IDNum = m1
320|        "QSOut(nIn).Name = Tiles(m1).FeatureName
321|        """"Exit For
322|       End If
323|     Next m1
324|     dynIn.Filter = strFilter
325|   Case Else
326|    Beep
327| End Select
328| End Sub
329|
330| Sub srchWithinBox (Theme As typTheme)
331| Select Case Theme.ThemeType
332|   Case SkipThis
333|     'Exit Subroutine automatically
334|   Case OnePoint
335|
336|   Case JointedLine
337|
338|   Case OutLine
339|
340|   Case BoundBlock
```

```
341|
342|    Case HatchLine
343|    Case TileBlock
344|
345|
346| '  Case NodePoint
347| '  Case LabelText
348| '  Case CCCircle
349| '  Case EEElipse
350| '  Case PPPixel
351| '  Case AAArc
352| '  Case RRRectangle
353|    Case Else
354|      MsgBox ("Need to add appropriate logic to subroutine SpatialSearch for
     ThemeType=" & Format(Theme.ThemeType))
355| End Select
356| End Sub
357|
358| Sub srchWithinRadius (X As Single, Y As Single, hGMem As Integer, Theme As
     typTheme, dynIn As Dynaset, QSOut() As typTaggedIDNum, lngHighlightColor As Long,
     Target As PictureBox)
359|  Dim nStart As Long, nEnd As Long
360|  Dim m1 As Long, m2 As Long
361|  Dim MinDist As Single, Dist As Single, r2 As Single
362|  Dim N As Long, nClosest As Long, M  As Long, L As Long
363|  Dim nPoints As Long, nChunks As Long, nCh As Long, nPos As Long
364|  Dim strFilter As String, nFound As Long
365|  r2 = QSOut(0).Dist1 ^ 2
366|  strFilter = "1=0"
367|
368|
369| Select Case Theme.ThemeType
370|   Case SkipThis
371|     'Exit Subroutine automatically
372|   Case OnePoint
373|      nStart = Points(Theme.FirstLoc).nStart
374|      nEnd = Points(Theme.LastLoc).nStart
375|      GoSub FindWithinRadius
376|      For m1 = Theme.FirstLoc To Theme.LastLoc
377|       For m2 = 1 To nFound
378|          If QSOut(m2).IDNum = Points(m1).nStart And QSOut(m2).GeometricFlags = 0
     Then
379|             QSOut(m2).IDNum = m1
380|             QSOut(m2).GeometricFlags = 1
381|             If strFilter = "1=0" Then
382|                strFilter = "(zqFeatureID = " & Points(m1).FeatureID & ")"
383|                QSOut(m2).FeatureID = Points(m1).FeatureID
384|               Else
385|                strFilter = strFilter & " or (zqFeatureID = " & Points(m1).FeatureID & ")"
386|                QSOut(m2).FeatureID = Points(m1).FeatureID
387|             End If
388|             Exit For
389|          End If
390|       Next m2
```

```
391|       Next m1
392|
393|    Case JointedLine
394|       nStart = Lines(Theme.FirstLoc).nStart
395|       nEnd = Lines(Theme.LastLoc).nStart
396|       GoSub FindWithinRadius
397|       For m1 = Theme.FirstLoc To Theme.LastLoc
398|        For m2 = 1 To nFound
399|           If Lines(m1).nStart <= QSOut(m2).IDNum And Lines(m1).nStart +
      Lines(m1).nPoints - 1 >= QSOut(m2).IDNum And QSOut(m2).GeometricFlags = 0 Then
400|              QSOut(m2).IDNum = m1
401|              QSOut(m2).GeometricFlags = 1
402|              If strFilter = "1=0" Then
403|                 strFilter = "(zqFeatureID = " & Lines(m1).FeatureID & ")"
404|                 QSOut(m2).FeatureID = Lines(m1).FeatureID
405|               Else
406|                 strFilter = strFilter & " or (zqFeatureID = " & Lines(m1).FeatureID & ")"
407|                 QSOut(m2).FeatureID = Lines(m1).FeatureID
408|              End If
409|              Exit For
410|           End If
411|        Next m2
412|       Next m1
413|    Case OutLine
414|
415|    Case BoundBlock
416|
417|    Case HatchLine
418|    Case TileBlock
419|       nStart = Tiles(Theme.FirstLoc).nStart
420|       nEnd = Tiles(Theme.LastLoc).nStart + 1
421|       GoSub FindWithinRadius
422|       For m1 = Theme.FirstLoc To Theme.LastLoc
423|        For m2 = 1 To nFound
424|           If (QSOut(m2).IDNum = Tiles(m1).nStart Or QSOut(m2).IDNum =
      Tiles(m1).nStart + 1) And QSOut(m2).GeometricFlags = 0 Then
425|              QSOut(m2).GeometricFlags = 1 'Don't repeat this point
426|              QSOut(m2).IDNum = m1
427|              If strFilter = "1=0" Then
428|                 strFilter = "(zqFeatureID = " & Tiles(m1).FeatureID & ")"
429|                 QSOut(m2).FeatureID = Tiles(m1).FeatureID
430|               Else
431|                 strFilter = strFilter & " or (zqFeatureID= " & Tiles(m1).FeatureID & ")"
432|                 QSOut(m2).FeatureID = Tiles(m1).FeatureID
433|              End If
434|              Exit For
435|           End If
436|        Next m2
437|       Next m1
438| '  Case NodePoint
439| '  Case LabelText
440| '  Case CCCircle
441| '  Case EEElipse
442| '  Case PPPixel
```

```
443|  '  Case AAArc
444|  '  Case RRRectangle
445|     Case Else
446|       MsgBox ("Need to add appropriate logic to subroutine SpatialSearch for
      ThemeType=" & Format(Theme.ThemeType))
447|  End Select
448|  dynIn.Filter = strFilter
449|  Exit Sub
450|  'LOGIC FOR OUTPUTTING THE DATA TO GRID
451|  FindWithinRadius:
452|  'X As Single, Y As Single, nStart As Long, nEnd As Long, hGMem As Integer)
453|   MinDist = 1E+34
454|   nPoints = nEnd - nStart + 1
455|   nChunks = nPoints \ 4096
456|   nPos = 0
457|   nFound = 0
458|   frmMap.lblPercentDone = "Spatial Query within Radius " & Format(QSOut(0).Dist1)
459|   frmMap.pnl3dPercentDone.FloodPercent = 0
460|   ReDim Pts(0 To 4095) As RealPoint
461|   For nCh = 1 To nChunks
462|     Call GetRlPntsFromGlobalMem(hGMem, Pts(0), nStart + nPos, 4096)
463|       For M = 0 To 4095
464|         Dist = (X - Pts(M).X) * (X - Pts(M).X) + (Y - Pts(M).Y) * (Y - Pts(M).Y)
465|         If Dist < r2 Then
466|           nFound = nFound + 1
467|           ReDim Preserve QSOut(0 To nFound)
468|           QSOut(nFound).IDNum = M + nPos + nStart
469|           QSOut(nFound).Dist1 = Sqr(Dist)
470|           QSOut(nFound).X = Pts(M).X
471|           QSOut(nFound).Y = Pts(M).Y
472|
473|           'Target.Circle (Pts(M).X, Pts(M).Y), QSOut(0).Dist1 / 4, lngHighlightColor
474|
475|         End If
476|       Next M
477|       frmMap.pnl3dPercentDone.FloodPercent = nCh / nChunks * 100
478|       nPos = nPos + 4096
479|   Next nCh
480|   L = nPoints - nPos
481|   ReDim Pts(0 To L - 1)
482|   Call GetRlPntsFromGlobalMem(hGMem, Pts(0), nStart + nPos, L)
483|       For M = 0 To L - 1
484|         Dist = (X - Pts(M).X) * (X - Pts(M).X) + (Y - Pts(M).Y) * (Y - Pts(M).Y)
485|         If Dist < r2 Then
486|           nFound = nFound + 1
487|           ReDim Preserve QSOut(0 To nFound)
488|           QSOut(nFound).IDNum = M + nPos + nStart
489|           QSOut(nFound).Dist1 = Sqr(Dist)
490|           QSOut(nFound).X = Pts(M).X
491|           QSOut(nFound).Y = Pts(M).Y
492|           'Target.Circle (Pts(M).X, Pts(M).Y), QSOut(0).Dist1 / 4, lngHighlightColor
493|         End If
494|       Next M
495|
```

```
496|   frmMap.pnl3dPercentDone.FloodPercent = 100
497|   frmMap.lblPercentDone = "Done finding points within radius " &
       Format(QSOut(0).Dist1)
498|
499|
500| Return
501|
502| End Sub
503|
504| Sub zzOldsrchClosest1 (X As Single, Y As Single, hGMem As Integer, Theme As
       typTheme, dynIn As Dynaset, QSOut() As typTaggedIDNum)
505|   Dim nStart As Long, nEnd As Long
506|   Dim m1 As Long, m2 As Long
507|   Dim MinDist As Single, Dist As Single, ClosestPt As RealPoint
508|   Dim N As Long, nClosest As Long, M  As Long, L As Long
509|   Dim nPoints As Long, nChunks As Long, nCh As Long, nPos As Long
510|
511|
512|
513| Select Case Theme.ThemeType
514|   Case SkipThis
515|     'Exit Subroutine automatically
516|   Case OnePoint
517|      nStart = Points(Theme.FirstLoc).nStart
518|      nEnd = Points(Theme.LastLoc).nStart
519|      GoSub FindClosestPoint
520|      For m1 = Theme.FirstLoc To Theme.LastLoc
521|       If Points(m1).nStart = nClosest Then
522|          dynIn.Filter = "ZQFeatureID = " & Points(m1).FeatureID & ""
523|          Exit For
524|       End If
525|      Next m1
526|      frmMap.picMap.Circle (ClosestPt.X, ClosestPt.Y), 500, RGB(0, 0, 200)
527|
528|   Case JointedLine
529|      nStart = Lines(Theme.FirstLoc).nStart
530|      nEnd = Lines(Theme.LastLoc).nStart + Lines(Theme.LastLoc).nPoints
531|      GoSub FindClosestPoint
532|      For m1 = Theme.FirstLoc To Theme.LastLoc
533|       If Lines(m1).nStart <= nClosest And Lines(m1).nStart + Lines(m1).nPoints - 1 >=
     nClosest Then
534|          dynIn.Filter = "ZQFeatureID = " & Lines(m1 - 1).FeatureID
535|          Exit For
536|       End If
537|      Next m1
538|      frmMap.picMap.Circle (ClosestPt.X, ClosestPt.Y), 500, RGB(255, 128, 0)
539|
540|   Case OutLine
541|
542|   Case BoundBlock
543|
544|   Case HatchLine
545|   Case TileBlock
546|      nStart = Tiles(Theme.FirstLoc).nStart
```

```
547|       nEnd = Tiles(Theme.LastLoc).nStart
548|       GoSub FindClosestPoint
549|       For m1 = Theme.FirstLoc To Theme.LastLoc
550|        If Tiles(m1).nStart = nClosest Then
551|           dynIn.Filter = "ZQFeatureID = ' & Tiles(m1).FeatureID"
552|           frmMap.picMap.Circle (ClosestPt.X, ClosestPt.Y), 500, RGB(0, 0, 200)
553|          Exit For
554|        End If
555|       Next m1
556|
557|
558| '  Case NodePoint
559| '  Case LabelText
560| '  Case CCCircle
561| '  Case EEElipse
562| '  Case PPPixel
563| '  Case AAArc
564| '  Case RRRectangle
565|     Case Else
566|       MsgBox ("Need to add appropriate logic to subroutine SpatialSearch for
      ThemeType=" & Format(Theme.ThemeType))
567| End Select
568| Exit Sub
569| FindClosestPoint:
570| 'X As Single, Y As Single, nStart As Long, nEnd As Long, hGMem As Integer)
571|  MinDist = 1E+34
572|  nPoints = nEnd - nStart + 1
573|  nChunks = nPoints \ 4096
574|  nPos = 0
575|  frmMap.lblPercentDone = "Spatial Query by Closest Point"
576|  frmMap.pnl3dPercentDone.FloodPercent = 0
577|  ReDim Pts(0 To 4095) As RealPoint
578|  For nCh = 1 To nChunks
579|    Call GetRlPntsFromGlobalMem(hGMem, Pts(0), nStart + nPos, 4096)
580|     For M = 0 To 4095
581|      Dist = (X - Pts(M).X) * (X - Pts(M).X) + (Y - Pts(M).Y) * (Y - Pts(M).Y)
582|      If Dist < MinDist Then
583|        MinDist = Dist
584|        nClosest = M + nPos + nStart
585|        LSet ClosestPt = Pts(M)
586|      End If
587|     Next M
588|     frmMap.pnl3dPercentDone.FloodPercent = nCh / nChunks * 100
589|     nPos = nPos + 4096
590|  Next nCh
591|  L = nPoints - nPos
592|  ReDim Pts(0 To L - 1)
593|  Call GetRlPntsFromGlobalMem(hGMem, Pts(0), nStart + nPos, L)
594|     For M = 0 To L - 1
595|      Dist = (X - Pts(M).X) * (X - Pts(M).X) + (Y - Pts(M).Y) * (Y - Pts(M).Y)
596|      If Dist < MinDist Then
597|        MinDist = Dist
598|        nClosest = M + nPos + nStart
599|        LSet ClosestPt = Pts(M)
```

```
600|         End If
601|       Next M
602|    frmMap.pnl3dPercentDone.FloodPercent = 100
603|    frmMap.lblPercentDone = "Done finding closest point"
604| 'In the end, Pt contains the location of the closest point,
605| ' and nClosest is its index in the global point array.
606| '   FindClosestGlobalPoint = nClosest 'This is its location in the global point list.
607|
608| Return
609|
610| End Sub
611|
```

```
 1| Option Explicit
 2| Option Compare Text
 3|
 4|
 5| 'Global Zero As Long, One As Long, Two As Long
 6| '
 7| "  PolyFill() Modes
 8| 'Global Const ALTERNATE = 1
 9| 'Global Const WINDING = 2
10| '
11| "Declare Sub Polygon Lib "GDI" (ByVal hDC As Integer, lpPoints As APIPoint, ByVal
    nCount As Integer)
12| "Declare Sub HUGEPolygon Lib "GDI" Alias "Polygon" (ByVal hDC As Integer, ByVal
    ptrToAPIPoints As Long, ByVal nCount As Integer)
13| 'Declare Function Polygon Lib "GDI" (ByVal hDC As Integer, lpPoints As APIPoint, ByVal
    nCount As Integer) As Integer
14| '
15| '
16| 'Declare Sub SetPolyFillMode Lib "GDI" (ByVal hDC As Integer, ByVal nPolyFillMode As
    Integer)
17| "Declare Sub Polyline Lib "GDI" (ByVal hDC As Integer, lpPoints As APIPoint, ByVal
    nCount As Integer)
18| "Declare Sub HUGEPolyline Lib "GDI" Alias "PolyLine" (ByVal hDC As Integer, ByVal
    ptrToAPIPoints As Long, ByVal nCount As Integer)
19| 'Declare Function Polyline Lib "GDI" (ByVal hDC As Integer, lpPoints As APIPoint, ByVal
    nCount As Integer) As Integer
20| '
21| '
22| 'Declare Function Rectangle Lib "GDI" (ByVal hDC As Integer, ByVal X1 As Integer,
    ByVal Y1 As Integer, ByVal X2 As Integer, ByVal Y2 As Integer) As Integer
23|
24| Private Function ClearSQLMineField (strSQLIn As String) As String
25| "ClearSQLMineField will  replace characters inside quotes or brackets with tilde's ~
26| " This is a pre-processing step so that subsequent routines don't have to worry
27| " about whether the FROM they just found was in a field name, table name, or in quotes,
28| "  If it's left standing after this, it isn't in any "shell" so it is probably
29| " an SQL keyword  (or an inoffensive part of a field name or what have you.)
30| Dim chTilde As String, chSpace As String, Temp As String
31| chTilde = "~"
32|
33| Dim n As Long, nChars As Long, nCh As Long
34|
35| 'First, make sure all white space is spaces
36| nChars = Len(strSQLIn)
37| Temp = ""
38| For n = 1 To nChars
39|   If Asc(Mid$(strSQLIn, n, 1)) < 32 Then
40|      Temp = Temp & " "
41|    Else
42|      Temp = Temp & Mid$(strSQLIn, n, 1)
43|   End If
44| Next n
45| Dim iState As Integer, ThisChar As String, TempOut  As String
46| Const InsideNothing = 0
```

```
Const InsideSingleQuote = 1
Const InsideDoubleQuote = 2
Const InsideBrackets = 4

nChars = Len(Temp)
iState = InsideNothing: TempOut = ""
For n = 1 To nChars
   ThisChar = Mid$(Temp, n, 1)
   Select Case iState
     Case InsideNothing
   Select Case ThisChar
   Case """" 'double quote
     ThisChar = "~"
     iState = InsideDoubleQuote
   Case "'"  'single quote
      ThisChar = "~"
      iState = InsideSingleQuote
   Case "["  'bracket
      ThisChar = "~"
      iState = InsideBrackets
   Case Else
     'Do nothing
   End Select
     Case InsideSingleQuote
   Select Case ThisChar
     Case "'"
        ThisChar = "~"
        iState = InsideNothing
     Case Else
        ThisChar = "~"
   End Select
    Case InsideDoubleQuote
   Select Case ThisChar
     Case """"
        ThisChar = "~"
        iState = InsideNothing
     Case Else
        ThisChar = "~"
   End Select
    Case InsideBrackets
   Select Case ThisChar
     Case "]"
        ThisChar = "~"
        iState = InsideNothing
     Case Else
        ThisChar = "~"
   End Select
   End Select
   TempOut = TempOut & ThisChar
Next n

ClearSQLMineField = TempOut

End Function
```

```
101|
102| Function ExtractFieldName (ByVal InString As String) As String
103| 'Used to pull a field name out of SQL Statement.
104| 'The field name can be in any of several forms.
105| ' FieldName   [Field Name] TableName.FieldName
106| '         [Table Name].FieldName [Table Name].[Field Name]
107| ' It will include a trailing comma, if one is concatenated onto the
108| '    end of a field name.
109| ' Thus, FieldName, Field2Name, would return FieldName,   and not FieldName
110|
111|
112| Dim Temp As String
113| 'Stop
114| Temp = InString
115| Temp = Trim$(InString) & " "
116| If (Mid$(Temp, 1, 1) = "[") Then 'special handling for brackets
117|  'Find next ]
118|  Dim RBrackLoc As Integer
119|  RBrackLoc = InStr(Temp, "]")
120|  If (Mid$(Temp, RBrackLoc + 1, 1) = " ") Then
121|    'This possibility was for a string starting with a [, and ending with the next ]
122|    ' This would get "[Field Name]" from " [Field Name] AS FieldName"
123|    ExtractFieldName = Mid$(Temp, 1, RBrackLoc)
124|    Exit Function
125|   Else
126|    If (Mid$(Temp, RBrackLoc + 2, 1) = "[") Then 'special handling for brackets
127|      'Find next ]
128|      ' This would get "[List of Tables].[Field Name]" from " [List of Tables].[Field Name] AS
    FieldName"
129|      RBrackLoc = InStr(RBrackLoc + 2, Temp, "]")
130|      ExtractFieldName = Mid$(Temp, 1, RBrackLoc)
131|      Exit Function
132|     Else
133|      'Started with a [ ]. style field, and ends with a single word
134|      ' This would get "[List of Tables].FieldName" from " [List of Tables].FieldName AS
    FieldName"
135|      RBrackLoc = InStr(RBrackLoc + 2, Temp, " ")
136|      ExtractFieldName = Mid$(Temp, 1, RBrackLoc)
137|      Exit Function
138|    End If
139|  End If
140| Else
141|  'The string does not start with a [, so look for the next space
142|    Dim SpaceLoc As Integer
143|    SpaceLoc = InStr(Temp, " ")
144|    Dim PeriodLoc As Integer
145|    PeriodLoc = InStr(Temp, ".")
146|    If (PeriodLoc = 0) Or (PeriodLoc > SpaceLoc) Then
147|      'Just a string of characters with no brackets or periods
148|      ExtractFieldName = Mid$(Temp, 1, SpaceLoc - 1)
149|      Exit Function
150|     Else
151|      'we have a simple string to start, followed by a period.
152|      'So see if the next character after the period is a bracket.
```

```
153|      If (Mid$(Temp, PeriodLoc + 1, 1) = "[") Then
154|   'We've got a bracketed second string like "TableName.[Field Name] as qwerty"
155|   ExtractFieldName = Mid$(Temp, 1, InStr(Temp, "]"))
156|   Exit Function
157|      Else
158|   'A word followed by a period followed by a word
159|   ExtractFieldName = Mid$(Temp, 1, SpaceLoc - 1)
160|   Exit Function
161|     End If
162|    End If
163| End If
164|
165| End Function
166|
167| Private Function GetColorWithCmnDialog (ByVal ColorIsNow As Long) As Long
168| '   Dim Temp As Long
169| '   Temp = ColorIsNow
170| '
171| '   If ColorIsNow = 0 Then ColorIsNow = 128
172| '   frmThemeViewEdit!CMDialog1.Color = ColorIsNow
173| '   frmThemeViewEdit!CMDialog1.CancelError = True
174| 'On Error Resume Next 'This will only skip the color setting, so we're safe.
175| '   frmThemeViewEdit!CMDialog1.Flags = CC_RGBINIT Or CC_FULLOPEN 'Show the
    full dialog box (allowing custom colors) and initialize the value.
176| '   frmThemeViewEdit!CMDialog1.Action = 3 ' so open the color common dialog box.
177| '   Temp = frmThemeViewEdit!CMDialog1.Color'Only executed if user chose OK.
178| '   frmThemeViewEdit!CMDialog1.CancelError = False
179| '   GetColorWithCmnDialog = Temp
180| '
181| End Function
182|
183| Function GetExpressionInBrackets (ByVal strIn As String) As String
184| "Gets everything in brackets
185| Dim strTemp As String
186|   strIn = Trim$(strIn)
187|   If Left$(strIn, 1) <> "[" Then
188|     strTemp = strIn
189|     Else
190|     strTemp = Mid$(strIn, 1, InStr(strIn, "]"))
191|   End If
192|
193|   GetExpressionInBrackets = strTemp
194|
195| End Function
196|
197| Function GetExpressionInParentheses (ByVal strIn As String) As String
198| Dim nRt As Long, nLft As Long, n As Long, strTemp As String, str1 As String
199|
200| strIn = Trim$(strIn)
201| If strIn = "" Then GetExpressionInParentheses = " "
202| If Left$(strIn, 1) <> "(" Then
203|   GetExpressionInParentheses = strIn
204|   Exit Function
205| End If
```

```
206|
207| For n = 1 To Len(strIn)
208|   str1 = Mid$(strIn, n, 1)
209|   strTemp = strTemp & str1
210|   If str1 = "(" Then nLft = nLft + 1
211|   If str1 = ")" Then nRt = nRt + 1
212|   If nRt = nLft Then Exit For
213| Next n
214|
215|  GetExpressionInParentheses = strTemp
216|
217| End Function
218|
219| Function GetExpressionInQuotes (ByVal strIn As String) As String
220| Dim nRt As Long, nLft As Long, n As Long
221| Dim strTemp As String, str1 As String, strLookFor As String
222|
223| strIn = Trim$(strIn)
224| If strIn = "" Then GetExpressionInQuotes = " "
225| Select Case Left$(strIn, 1)
226|   Case Chr$(34) 'double quotes
227|     strLookFor = Chr$(34)
228|   Case Chr$(39) 'single quotes
229|     strLookFor = Chr$(39)
230|   Case Else
231|     GetExpressionInQuotes = strIn
232|     Exit Function
233| End Select
234|
235|   strTemp = strLookFor
236| For n = 2 To Len(strIn)
237|   str1 = Mid$(strIn, n, 1)
238|   strTemp = strTemp & str1
239|   If str1 = strLookFor Then Exit For
240| Next n
241|
242| GetExpressionInQuotes = strTemp
243|
244|
245|
246|
247| End Function
248|
249| Function GetFieldListFromSQL (strSQLIn As String) As String
250| " This will return the part of an SQL statement that contains the field list
251|
252| Dim Temp1 As String, Temp2 As String, Temp3 As String
253| Dim n1 As Long, n2 As Long
254|
255|   Temp1 = UCase$(ClearSQLMineField(strSQLIn))
256|
257|   'Find the word SELECT and the word FROM
258|   'Use of spaces before and after avoids some booby traps
259|   n1 = InStr(Temp1, " SELECT ")
```

```
260|    If n1 = 0 Then n1 = InStr(Temp1, "SELECT ") 'Kludge to account for SELECT being
       first word in statement
261|
262|    n2 = InStr(Temp1, " FROM ")
263|    If n1 = 0 Or n2 = 0 Then Exit Function 'We've got trouble, right here in River City....
264|    If n2 < n1 Then Exit Function ' with a capital T that rhymes with P that stands for Pool...
265|    n1 = n1 + Len(" SELECT ") - 2
266|    Temp2 = Mid$(strSQLIn, n1, n2 - n1 + 1)
267|    Temp3 = UCase$(Temp2)
268| 'Get rid of any leading distinct or distinctrow.
269|    n1 = InStr(Temp3, " DISTINCTROW ")
270|
271|    If n1 <> 0 Then
272|      Temp2 = Mid$(Temp2, n1 + Len(" DISTINCTROW ") - 1)
273|     Else 'Also check for DISTINCT
274|      n1 = InStr(Temp3, " DISTINCT ")
275|     If n1 <> 0 Then
276|       Temp2 = Mid$(Temp2, n1 + Len(" DISTINCT ") - 1)
277|     Else
278|       'We are happy with it as is.
279|     End If
280|    End If
281|    GetFieldListFromSQL = Temp2
282|
283| End Function
284|
285| Function GetNextWord (ByVal InString As String) As String
286|  Dim Temp As String, n As Long, nLen As Long, iStart As Long
287|  On Error Resume Next
288|
289|  nLen = Len(Trim$(InString))
290|  If nLen = 0 Then Exit Function
291|  'Assume everything with an ASCII value below 33 is a space (CRLFs, tabs, spaces...)
292|
293|  'First, get to the first non-whitespace character at iStartChar or onward
294|  iStart = 1
295|  For n = 1 To nLen
296|    If Asc(Mid$(InString, n, 1)) > 32 Then Exit For
297|    iStart = n + 1
298|  Next n
299|
300|  For n = iStart To nLen
301|     If Asc(Mid$(InString, n, 1)) > 32 Then
302|     Temp = Temp & Mid$(InString, n, 1)
303|       Else
304|     Exit For
305|     End If
306|  Next n
307|
308|  GetNextWord = Temp
309|
310|  End Function
311|
312|  Function GetRestOfStringAfter (ByVal strMain As String, ByVal strPart As String) As
```

```
      String
313|
314| "If the string strPart occurs in the string strMain, this will return the
315| " the rest of the string after the last location in strPart.
316| " Otherwise it returns an empty string
317|   On Error Resume Next
318|   GetRestOfStringAfter = Mid$(strMain, InStr(1, strMain, strPart, 1) + Len(strPart))
319|
320| End Function
321|
322| Function NextWord (ByVal InString As String, iStartChar As Long) As String
323|  Dim Temp As String, n As Long, nLen As Long, iStartCharTemp As Long
324|  On Error Resume Next
325|
326|  nLen = Len(InString)
327|  If iStartChar = 0 Then iStartChar = 1
328|  If iStartChar >= nLen Then Exit Function
329| 'Assume everything with an ASCII value below 33 is a space (CRLFs, tabs, spaces...)
330|
331| 'First, get to the first non-whitespace character at iStartChar or onward
332| iStartCharTemp = iStartChar
333| Temp = ""
334| For n = iStartChar To nLen
335|    If Asc(Mid$(InString, n, 1)) > 32 Then Exit For
336|    iStartCharTemp = n + 1
337| Next n
338| iStartChar = iStartCharTemp
339|
340|
341| For n = iStartChar To nLen
342|     If Asc(Mid$(InString, n, 1)) > 32 Then
343|     Temp = Temp & Mid$(InString, n, 1)
344|     iStartCharTemp = n
345|       Else
346|     Exit For
347|     End If
348| Next n
349|
350|  iStartChar = iStartCharTemp + 1
351|  If iStartChar > nLen Then iStartChar = nLen
352|
353|  NextWord = Temp
354|
355|
356|
357| End Function
358|
359| Sub PullFieldNamesFromString (ByVal InString As String, FieldsIn() As String,
      FieldsOut() As String, nFields As Long)  'ExpressionsIn As String,
360| 'This subroutine parses field names out of a string, and puts them into two
361| ' other strings
362| 'The input string, InString, is not modified by this procedure
363| 'The output string arrays, FieldsIn and FieldsOut, contain the field names
364| 'FieldsIn() is the name of the field in the underlying table/query
```

```
365| 'FieldsOut() is the name of the resultant field in the query.
366|
367| 'PullFieldNamesFromString is meant to be used in parsing SQL statements,
368| '  in particular "N parameters expected, but only M supplied", which occurs
369| '  whenever a fieldname doesn't exist in the current table/query.
370| '
371| ' Greg Pouch 12 January 1993
372| ' Greg Pouch 1 December 1995
373| 'KGS
374|
375| Dim strSQL_Remaining As String, str1 As String, strFieldIn_Temp  As String, strTemp3
    As String
376| Dim n As Long, nF As Long, nCh  As Long, nChars As Long
377|
378| Dim strFieldIn As String, strFieldOut As String
379|
380| strSQL_Remaining = Trim$(InString) & " "
381|
382| 'Replace ALL low ascii character with blanks
383| For n = 1 To Len(strSQL_Remaining)
384|   If Asc(Mid$(strSQL_Remaining, n, 1)) < 33 Then Mid$(strSQL_Remaining, n, 1) = " "
385| Next n
386|
387|
388|
389|
390|
391|
392| Do While Len(strSQL_Remaining) > 0
393|   nFields = nFields + 1
394|   If nFields > 500 Then
395|     Beep
396|     MsgBox "Error in PullFieldNamesFromString.  After more than 500 iterations, failed
    to terminate.  The troubling statement follows." & CRLF & InString
397|     Exit Do
398|   End If
399|  'Get the Field Name IN
400|   strSQL_Remaining = Trim$(strSQL_Remaining) & " "
401|   str1 = Left$(strSQL_Remaining, 1)
402|   Select Case str1
403|     Case "["    'Find the next matching bracket
404|      strFieldIn = GetExpressionInBrackets(strSQL_Remaining)
405|      strSQL_Remaining = GetRestOfStringAfter(strSQL_Remaining, strFieldIn)
406|   If Left$(strSQL_Remaining, 1) = "." Then
407|    'We have a name of the form [table].Field  or [Table].[field]
408|    strFieldIn = strFieldIn & "."
409|    strSQL_Remaining = Mid$(strSQL_Remaining, 2)'Get rid of the dot
410|    If Left$(strSQL_Remaining, 1) = "[" Then
411|      strFieldIn_Temp = GetExpressionInBrackets(strSQL_Remaining)
412|      strSQL_Remaining = GetRestOfStringAfter(strSQL_Remaining, strFieldIn_Temp)
413|      strFieldIn = strFieldIn & strFieldIn_Temp: strFieldIn_Temp = ""
414|    Else
415|      strFieldIn_Temp = GetNextWord(strSQL_Remaining)
416|      strSQL_Remaining = GetRestOfStringAfter(strSQL_Remaining, strFieldIn_Temp)
```

```
417|       strFieldIn = strFieldIn & strFieldIn_Temp: strFieldIn_Temp = ""
418|     End If
419|    End If
420|      Case "("
421|     strFieldIn = GetExpressionInParentheses(strSQL_Remaining)
422|     strSQL_Remaining = GetRestOfStringAfter(strSQL_Remaining, strFieldIn)
423|      Case """"   'One double quote
424|     strFieldIn = GetExpressionInQuotes(strSQL_Remaining)
425|     strSQL_Remaining = GetRestOfStringAfter(strSQL_Remaining, strFieldIn)
426|      Case "'"    'one single quote
427|     strFieldIn = GetExpressionInQuotes(strSQL_Remaining)
428|     strSQL_Remaining = GetRestOfStringAfter(strSQL_Remaining, strFieldIn)
429|      Case Else
430|     strFieldIn = GetNextWord(strSQL_Remaining)
431|     strSQL_Remaining = GetRestOfStringAfter(strSQL_Remaining, strFieldIn)
432|    End Select
433|
434|    strSQL_Remaining = Trim$(strSQL_Remaining)
435|  'Now grab the next word, which should be either AS, or a comma
436|   'dDebug.Print strFieldIn, strSQL_Remaining
437|    str1 = Left$(strSQL_Remaining, 1)
438|    Select Case str1
439|    Case ","
440|      'no AS name, just use the old name
441|      strSQL_Remaining = Mid$(strSQL_Remaining, 2)'Get rid of the comma
442|      strFieldOut = GetNextWord(strSQL_Remaining)
443|      strSQL_Remaining = GetRestOfStringAfter(strSQL_Remaining, strFieldOut)
444|    Case Else 'might be as
445|      If (UCase$(Left$(strSQL_Remaining, 2)) = "AS") Then
446|     strSQL_Remaining = Trim$(Mid$(strSQL_Remaining, 3))'Get rid of the AS
447|     str1 = Left$(strSQL_Remaining, 1)
448|     If str1 = "[" Then
449|      strFieldOut = GetExpressionInBrackets(strSQL_Remaining)
450|      strSQL_Remaining = GetRestOfStringAfter(strSQL_Remaining, strFieldOut)
451|     Else
452|      strFieldOut = GetNextWord(strSQL_Remaining)
453|      strSQL_Remaining = GetRestOfStringAfter(strSQL_Remaining, strFieldOut)
454|     End If
455|    Else
456|     strFieldOut = "Was not able to parse"
457|     strFieldIn_Temp = GetNextWord(strSQL_Remaining)
458|     strSQL_Remaining = GetRestOfStringAfter(strSQL_Remaining, strFieldIn_Temp)
459|     strFieldIn_Temp = ""
460|       'sStop 'This would be a good time to panic because
461|         'after a field name, we have neither a comma nor an AS clause
462|
463|      End If
464|    End Select
465|    If GetNextWord(strSQL_Remaining) = "," Then strSQL_Remaining =
     GetRestOfStringAfter(strSQL_Remaining, ",")
466|
467|    ReDim Preserve FieldsIn(1 To nFields), FieldsOut(1 To nFields)
468|      strFieldIn = Trim$(strFieldIn)
469|    FieldsIn(nFields) = Trim$(strFieldIn)
```

```
470|      strFieldOut = Trim$(strFieldOut)
471|      If Right$(strFieldOut, 1) = "," Then strFieldOut = Left$(strFieldOut, Len(strFieldOut) -
     1)
472|    FieldsOut(nFields) = strFieldOut
473| 'dDebug.Print strFieldIn, strFieldOut, strSQL_Remaining
474|
475| Loop
476|
477|
478|
479|
480|
481|
482|
483|
484|
485| End Sub
486|
487| Function SplitSQLStatement (strSQLIn As String, strParameters As String, strFields As
     String, strFrom As String, strWhere As String, strGroupBy As String, strHaving As String,
     strOrderBy As String) As Integer
488| " Given an SQL simple select SQL statement, this routine will break it apart (hopefully)
489| " into a the parts listed in the call statement.
490| "Return value of true indicates this routine thought it was a valid SQL statement,
491| " false indicates this routine could not parse it.
492|
493| ' if SplitSQLStatement (strSQLIn , strParameters , strFields, strFrom , strWhere,
     strGroupBy, strHaving , strOrderBy ) = false then exit sub
494| ' Dim strSQLIn As String, strParameters As String, strFields As String, strFrom As String,
     strWhere As String, strGroupBy As String, strHaving As String, strOrderBy As String
495|
496| 'strSQLIn        The input SQL statement to parse
497| 'strParameters    parameters required. Can be blank
498| 'strFields       list of fields, [field] As AliasName or whatever
499| 'strFromClause    the from clause.  Could include the join statements
500| 'strWhereClause   the Where clause  Can be blank
501| 'strGroupBy       the GroupBy Clause  Can be blank
502| 'strOrderBy       guess! Can be blank
503| Dim my_strSQLIn As String
504| Dim Temp1 As String, Temp2 As String, Temp3 As String
505|
506| Dim nParam As Integer, nSelect As Integer, nFrom As Integer, nLast As Integer
507| Dim nWhere As Integer, nGroupBy As Integer, nOrderBy As Integer, nHaving  As Integer
508| Dim nStart As Integer, nChars As Integer
509|
510| my_strSQLIn = strSQLIn
511| Temp1 = UCase$(ClearSQLMineField(my_strSQLIn))
512| nLast = InStr(Temp1, ";")
513| If nLast <> 0 Then 'Get rid of the trailing semi-colon.
514|    Mid$(my_strSQLIn, nLast, 1) = " "
515| End If
516|
517|
518|    'nParam and nSelect are substrings that could occur at the beginning of
```

```
519|    ' of the SQL statement, so precede the statement with a blank and substract
520|    ' 1 from the position for further use.
521|
522|    nParam = InStr(" " & Temp1, " PARAMETERS ") - 1
523|    nSelect = InStr(" " & Temp1, " SELECT ") - 1
524|    nFrom = InStr(Temp1, " FROM ")
525|    nWhere = InStr(Temp1, " WHERE ")
526|    nGroupBy = InStr(Temp1, " GROUP BY ")
527|    nOrderBy = InStr(Temp1, " ORDER BY ")
528|    nHaving = InStr(Temp1, " HAVING ")
529|
530|
531|    'This subroutine ignores INTERSECT, UNION, EXCEPT, ALL, CORRESPONDING
       clauses
532|
533|
534|    If nSelect = -1 Then 'Big trouble
535|       SplitSQLStatement = False
536|       Exit Function
537|     Else
538|      strFields = Trim$(GetFieldListFromSQL(my_strSQLIn))
539|    End If
540|
541|    If nParam = -1 Then 'no parameters statement
542|      strParameters = ""
543|     Else
544|      nStart = nParam + Len(" PARAMETERS ") - 1
545|      nChars = nSelect - nStart + 1
546|      strParameters = Trim$(Mid$(my_strSQLIn, nStart, nChars))
547|    End If
548|
549|
550|    'Read the string backwards, keeping track of the parts already used.
551|    nLast = Len(my_strSQLIn)
552|    If nOrderBy = 0 Then
553|      strOrderBy = ""
554|     Else
555|      nStart = nOrderBy + Len(" ORDER BY ") - 1
556|      nChars = nLast - nStart + 1
557|      nLast = nOrderBy
558|      strOrderBy = Trim$(Mid$(my_strSQLIn, nStart, nChars))
559|    End If
560|
561|    If nHaving = 0 Then
562|      strHaving = ""
563|    Else
564|      nStart = nHaving + Len(" HAVING ") - 1
565|      nChars = nLast - nStart + 1
566|      nLast = nHaving
567|      strHaving = Trim$(Mid$(my_strSQLIn, nStart, nChars))
568|    End If
569|
570|    If nGroupBy = 0 Then
571|      strGroupBy = ""
```

```
572|    Else
573|      nStart = nGroupBy + Len(" GROUP BY ") - 1
574|      nChars = nLast - nStart + 1
575|      nLast = nGroupBy
576|      strGroupBy = Trim$(Mid$(my_strSQLIn, nStart, nChars))
577|    End If
578|
579|    If nWhere = 0 Then
580|      strWhere = ""
581|    Else
582|      nStart = nWhere + Len(" WHERE ") - 1
583|      nChars = nLast - nStart + 1
584|      nLast = nWhere
585|      strWhere = Trim$(Mid$(my_strSQLIn, nStart, nChars))
586|    End If
587|
588|    If nFrom = 0 Then 'This clause is required
589|      SplitSQLStatement = 0
590|      Exit Function
591|      strFrom = ""
592|    Else
593|      nStart = nFrom + Len(" FROM ") - 1
594|      nChars = nLast - nStart + 1
595|      nLast = nFrom
596|      strFrom = Trim$(Mid$(my_strSQLIn, nStart, nChars))
597|    End If
598|
599| SplitSQLStatement = True
600| End Function
601|
602| Sub ZOLD_PullFieldNamesFromString (ByVal InString As String, FieldsIn() As String,
      FieldsOut() As String, nFields As Long)  'ExpressionsIn As String,
603| 'This subroutine parses field names out of a string, and puts them into two
604| ' other strings
605| 'The input string, InString, is not modified by this procedure
606| 'The output strings, FieldsIn and FieldsOut, contain the field names
607| 'FieldsIn is the name of the field in the underlying table/query
608| 'FieldsOut is the name of the resultant field in the query.
609| 'Field names are delimited with a CR-LF in both output strings.
610| '
611| 'PullFieldNamesFromString is meant to be used in error handling of SQL statements,
612| '  in particular "N parameters expected, but only M supplied", which occurs
613| '  whenever a fieldname doesn't exist in the current table/query.
614| '
615| ' Greg Pouch 12 January 1993
616| 'KGS
617|
618| Dim strTemp1 As String
619| Dim strTemp2 As String
620| Dim strTemp3 As String
621|
622| Dim jNextChar As Long
623| Dim jCurrChar As Long
624| Dim jLenInString As Long
```

```
625| Dim i_RtBrackLoc As Integer
626| Dim i_SpaceLoc As Integer
627| Dim i_PeriodLoc As Integer
628| Dim strNameIn As String
629| Dim strNameOut As String
630| Dim TheNextWord As String
631| Dim j_OldNextChar As Long
632| Dim lPassNumber As Long
633|
634|
635| strTemp2 = InString
636| 'Bad handling of parentheses, Replace with Blanks
637| strTemp2 = ReplaceString(strTemp2, "(", " ")
638| strTemp2 = ReplaceString(strTemp2, ")", " ")
639|
640| strTemp2 = Trim$(strTemp2) & " "
641|
642|
643|
644| jLenInString = Len(strTemp2)
645| nFields = 0
646| jCurrChar = 1
647|
648| Do Until jNextChar >= jLenInString
649|
650|  lPassNumber = lPassNumber + 1
651| 'This section gets an Input field name
652|   strTemp1 = Trim$(Mid$(strTemp2, jCurrChar)) & " "
653|   If Len(strTemp1) < 1 Then Exit Sub
654|
655| 'Figure out how to handle parentheses here.
656|  If (Left$(strTemp1, 1) = ")") Then
657|     'strNameIn = GetParenthesizedExpression(strTemp1, jCurrChar)
658|  End If
659|
660|   If (Left$(strTemp1, 1) = "[") Then 'special handling for brackets
661|   'Find next ]
662|
663|  i_RtBrackLoc = InStr(strTemp1, "]")
664|  If (Mid$(strTemp1, i_RtBrackLoc + 1, 1) = " ") Then
665|    'This possibility was for a string starting with a [, and ending with the next ]
666|    ' This would get "[Field Name]" from " [Field Name] AS FieldName"
667|    strNameIn = Mid$(strTemp1, 1, i_RtBrackLoc)
668|    jNextChar = i_RtBrackLoc + 1
669|   Else
670|    If (Mid$(strTemp1, i_RtBrackLoc + 1, 1) = ",") Then
671|     'This possibility was for a string starting with a [, and ending with the next ]
672|     ' This would get "[Field Name]," from " [Field Name], qrewr AS FieldName"
673|     strNameIn = Mid$(strTemp1, 1, i_RtBrackLoc + 1)
674|     jNextChar = i_RtBrackLoc + 2
675|     'Slight chance this might not be mutually excluseive of some of the
676|     ' other possibilities.
677|    End If
678|    If (Mid$(strTemp1, i_RtBrackLoc + 2, 1) = "[") Then 'special handling for brackets
```

```
679|      'Find next ]
680|      ' This would get "[List of Tables].[Field Name]" from " [List of Tables].[Field Name] AS
     FieldName"
681|      i_RtBrackLoc = InStr(i_RtBrackLoc + 2, strTemp1, "]")
682|      strNameIn = Mid$(strTemp1, 1, i_RtBrackLoc)
683|      jNextChar = i_RtBrackLoc + 1
684|     Else
685|      'Started with a [ ]. style field, and ends with a single word
686|      ' This would get "[List of Tables].FieldName" from " [List of Tables].FieldName AS
     FieldName"
687|      i_RtBrackLoc = InStr(i_RtBrackLoc + 2, strTemp1, " ")
688|      strNameIn = Mid$(strTemp1, 1, i_RtBrackLoc - 1)
689|      jNextChar = i_RtBrackLoc
690|    End If
691|  End If
692| Else
693|  'The string does not start with a [, so look for the next space
694|
695|    i_SpaceLoc = InStr(strTemp1, " ")
696|
697|    i_PeriodLoc = InStr(strTemp1, ".")
698|    If (i_PeriodLoc = 0) Or (i_PeriodLoc > i_SpaceLoc) Then
699|      'Just a string of characters with no brackets or periods
700|      strNameIn = Mid$(strTemp1, 1, i_SpaceLoc - 1)
701|      jNextChar = i_SpaceLoc
702|     Else
703|      'we have a simple string to start, followed by a period.
704|      'So see if the next character after the period is a bracket.
705|      If (Mid$(strTemp1, i_PeriodLoc + 1, 1) = "[") Then
706|   'We've got a bracketed second string like "TableName.[Field Name] as qwerty"
707|   strNameIn = Mid$(strTemp1, 1, InStr(strTemp1, "]"))
708|   jNextChar = InStr(strTemp1, "]") + 1
709|      Else
710|   'A word followed by a period followed by a word
711|   strNameIn = Mid$(strTemp1, 1, i_SpaceLoc - 1)
712|   jNextChar = i_SpaceLoc
713|     End If
714|    End If
715| End If
716|
717| If strNameIn = "[PointIDNumber]" Then Stop
718|
719| j_OldNextChar = jNextChar
720|
721| TheNextWord = UCase$(NextWord(strTemp1, jNextChar))
722| 'dDebug.Print lPassNumber, TheNextWord
723|
724| If TheNextWord = "AS" Then
725|   ' The next word in sequence is AS, so pull the word after that and use it
726|   ' for the strNameOut
727|
728|   strTemp3 = Mid$(strTemp1, jNextChar)
729|   strNameOut = ExtractFieldName(strTemp3)
730|   strTemp3 = Right$(strNameOut, 5)
```

```
731|    jNextChar = InStr(j_OldNextChar, strTemp1, strTemp3) + Len(strTemp3)
732|   Else
733|    jNextChar = j_OldNextChar
734|    strNameOut = strNameIn
735|  End If
736|
737|
738|  If Len(strNameIn) > 0 Then
739|   nFields = nFields + 1
740|   ReDim Preserve FieldsOut(1 To nFields), FieldsIn(1 To nFields)
741|   If Right$(strNameOut, 1) = "," Then strNameOut = Left$(strNameOut, Len(strNameOut)
      - 1)
742|   FieldsOut(nFields) = Trim$(strNameOut)
743|        'strNameIn = Strip2Brackets(strNameIn)
744|   FieldsIn(nFields) = Trim$(strNameIn)
745|   jCurrChar = jCurrChar + jNextChar
746|
747|   'dDebug.Print nFields, "_" & FieldsIn(nFields) & "_", "_" & FieldsOut(nFields) & "_"
748|
749|  End If
750|   ' Stop
751|  Loop
752|
753|  End Sub
754|
```