

SuperMap Objects Getting Started

SuperMap GIS Technologies, Inc.

February 2008 · China

© 2001-2008 SuperMap GIS Technologies, Inc. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form, or by any means, **including, but not limited to**, electronic, mechanical, photocopying, recording, or other wise, without written permission of the copyright holder.

Only SuperMap GIS Technologies, Inc. has the right to modify this publication, which does not include the other suppliers or agencies.

SuperMap and logo  are registered trademarks of SuperMap GIS Technologies, Inc. in China and other countries.

All the rights about SuperMap GIS belong to SuperMap GIS Technologies Inc.

SuperMap GIS Technologies Inc. has sole charge of selling and upgrading the entire software illustrated in this publication.

The proprietary materials of all the other companies and logos mentioned or which appear in this publication belong to the respective owners.

SuperMap GIS Technologies, Inc.

7th Floor, Tower B, Technology Fortune Center, No. 8, Xueqing Road, Haidian District, Beijing, China, 100085

Tel: (0086)-10- 82736655

Fax: (0086)-10-82734630

Website: <http://www.supermap.com>

Sales E-Mail: sales@supermap.com

Online Help: <http://smdn.supermap.com>

Technical Support Center: Support@supermap.com

We greatly appreciate any and all advices and suggestions regarding our products and services.

Content

1	Introduction	1
1.1	About the book	1
1.2	How to read this book.....	1
1.3	Personal motivation	2
1.4	Related controls and interface	2
1.5	Sample Data.....	3
2	Using SuperMap Objects in Visual Basic.....	5
2.1	Creating a new project.....	5
2.2	Loading SuperMap Objects controls	5
2.3	Opening a map and adding layers.....	7
2.4	Browsing map.....	8
2.5	Querying Properties	10
2.6	Querying Map.....	12
3	Using SuperMap Objects in Delphi.....	15
3.1	Creating a new project.....	15
3.2	Loading SuperMap Objects controls	15
3.3	Opening a map and adding layers.....	17
3.4	Browsing map.....	18
3.5	Querying Properties	20
3.6	Querying map	22
4	Using SuperMap Objects in Visual C++	25
4.1	Creating a new project: MySuperMap.....	25
4.2	Loading SuperMap Objects control.....	26
4.3	Opening a map and adding layers.....	30

4.4	Browsing map.....	34
4.5	Querying Properties.....	36
4.6	Querying map	37
5	Using SuperMap Objects in VB .NET	41
5.1	Creating a new project:MySuperMap.....	41
5.2	Loading SuperMap Objects control.....	41
5.3	Opening a map and adding layers.....	43
5.4	Browsing map.....	45
5.5	Querying Properties.....	47
5.6	Querying map	49
6	Using SuperMap Objects in C# .NET	53
6.1	Creating a new project: MySuperMap.....	53
6.2	Loading SuperMap Objects controls	54
6.3	Opening a map and adding layers.....	56
6.4	Browsing map.....	59
6.5	Querying Properties.....	61
6.6	Querying map	63
7	Appendix	67
7.1	How to register SuperMap Objects?.....	67
7.2	How to distribute your application with SuperMap Objects?.....	68



Introduction

1.1 About the book

Welcome to Getting Started with SuperMap Objects. This book is intended to help the first learner to understand and grasp the development approaches and procedures of SuperMap Objects, the powerful and easy-to-use components-based GIS platform. This book has 7 chapters. The first one will present a brief introduction about this book. Chapters 2 to 6 discuss how to build a simple application about map browsing and querying, and as well as we will show you how to load SuperMap Objects' controls under five different development languages, the most widely-used ones at present. The user can read and focus on the familiar or interested parts in the tutorial.

1.2 How to read this book

In this introduction to SuperMap Objects, you will learn to use SuperMap Objects and other program development languages to build an application that how to browse a map with SuperMap Objects. Along the way you will learn how to:

- Load SuperMap Objects controls into a project

- Use SuperWorkspace controls to open a SuperMap data source
- Add and display data on a map with multiple layers
- Control panning and zooming
- Create a control toolbar
- Perform spatial and logical queries

Through this book you will also understand the relationship between:

- Workspace and datasource
- Datasource and dataset
- Dataset and layers

1.3 Personal motivation

This book is mainly for the developers who have some basic development experience, but is a novice of SuperMap Objects. Users can select different development languages: Visual Basic 6.0, Delphi, Visual C++, Visual Studio C# .NET, or Visual Basic.NET, to implement some basic GIS functions in his own application by following the sample code presented in this book.

1.4 Related controls and interfaces

Table 1-1 *Related controls and interfaces*

Objects	Property	Method	Event
SuperMap Control	Action, Selection, Layers	ViewEntire, Connect, Refresh, Disconnect	GeometrySelected
SuperWorkspace Control	Object/ Handle	OpenDataSource	
soDataSource			
soDatasets	Count, Item		
soDatasetVector		Query	
soLayer		AddDataset	

soRecordset	FieldCount	GetFieldInfo, GetFieldValue, MoveFirst
soSelection		ToRecordset, FromRecordset
soStyle		SetPenColor SetPenStyle SetPenWidth SetBrushStyle SetBrushColor SetBrushBackColor SetBrushOpaqueRate

1.5 Sample Data

The sample data in this demo program, used to illustrate some basic operations and reversible queries, are world maps (world.sdb), which includes graticule (Grid) and world map (World). You need to add the two layers on SuperMap to display, and then you can query on the world map which is a SDBPlus file World.sdb and the attribute table World.sdd.

Chapter

2

Using SuperMap Objects in Visual Basic

2.1 Creating a new project

1. Create a working directory such as *C:\MyProject*.
2. Download the data compression package World.zip (Include World.sdb and World.sdd) to the working directory (C:\MyProject)
3. Start Microsoft Visual Basic.
4. Create a new project under the working directory (C:\MyProject).

2.2 Loading SuperMap Objects controls

1. Add SuperMap Objects control to ToolBox:
 - Use shortcut key Ctrl+T or right-click on ToolBox, then press “Components...”, the Components dialog box is displayed, as *Figure 2-1*

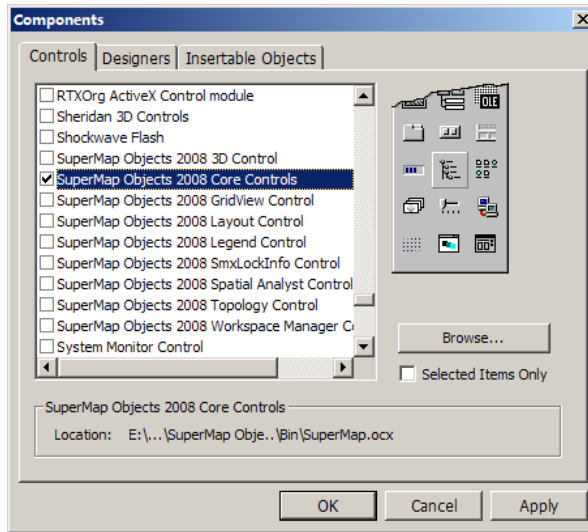


Figure 2-1 Components dialog box

- In the Components dialog box, Select “SuperMap Objects 2008 Core Controls” (by selecting the checkbox before it) and press OK. Then two controls will appear in the Visual Basic toolbox, **Figure 2-2**

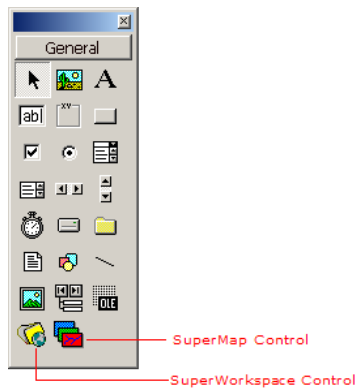


Figure 2-2 Toolbox

2. Click and draw the two Controls into Form1.

2.3 Opening a map and adding layers

1. Add the following code to the code page as the response when the program running:

```
Private Sub Form_Load()
    'Connect SuperWorkspace and SuperMap
    SuperMap1.Connect (SuperWorkspace1.Object)

    Dim strAlias As String           'Alias of datasource
    Dim nEngineType As seEngineType 'Engine type of data being used in this sample
    Dim strDataSourceName As String  'Absolute path of datasource
    Dim objDataSource As soDataSource 'Datasource object
    Dim bReadOnly As Boolean         'Whether the datasource to be opened is read only
    Dim objLayer As soLayer         'Layer object of SuperMap window
    Dim bAddToHead As Boolean       'Whether a new added layer will be brought to front
    Dim i As Integer                'A cycle variable

    'The Alias can be named arbitrarily, but you'd better use the same one as that of the
    'datasource file
    strAlias = "MyDataSource"
    'We will use a SDBPlus data in this sample
    nEngineType = sceSDBPlus
    'You can change this path name according to the particular place of your data, or you can use
    'a relative path.
    strDataSourceName = "c:\MyProject\world.sdb"
    ' Indicates the datasource to be opened can be modified
    bReadOnly = False

    'Open datasource
    Set objDataSource = SuperWorkspace1.OpenDataSource(strDataSourceName, strAlias,
nEngineType, bReadOnly)
    If objDataSource Is Nothing Then
        MsgBox "Failed to open the datasource!", vbInformation
    Else
        For i = 1 To objDataSource.Datasets.Count
            'Add all layers of the datasource to SuperMap window
            bAddToHead = True
            Set objLayer = SuperMap1.Layers.AddDataset(objDataSource.Datasets.Item(i),
bAddToHead)
        Next
    End If
    'Refresh the map window
    SuperMap1.Refresh
End Sub
```

```
'Modify the styles of the selection object
SuperMap1.selection.Style.PenColor = RGB(231, 77, 0)
SuperMap1.selection.Style.PenWidth = 1
SuperMap1.selection.Style.PenStyle = 1
SuperMap1.selection.Style.BrushStyle = 5
SuperMap1.selection.Style.BrushColor = RGB(115, 69, 140)
SuperMap1.selection.Style.BrushBackColor = RGB(239, 150, 255)
SuperMap1.selection.Style.BrushOpaqueRate = 50
'Release memory
Set objDataSource = Nothing
Set objLayer = Nothing
End Sub
```

2. Run the code, the result will be displayed as below: *Figure 2-3*

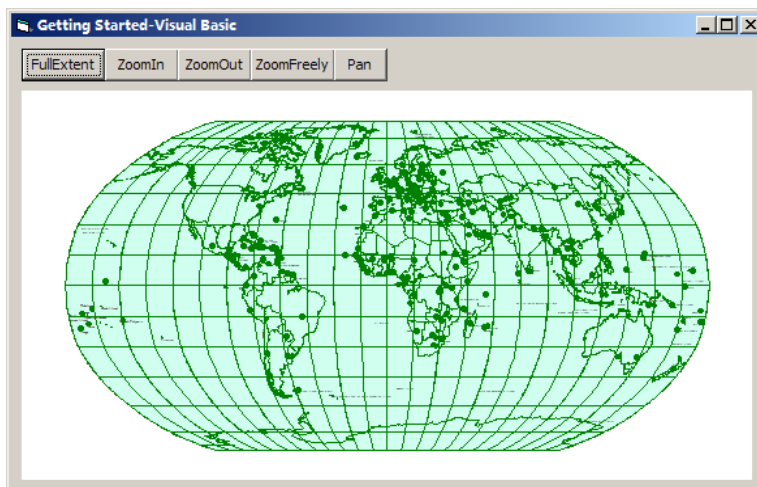


Figure 2-3 *View full extent of the map*

2.4 Browsing map

It is very convenient to perform map operations with SuperMap Objects, such as zooming in, zooming out, zooming freely, panning map, showing the full extent of map, drawing points or lines, etc. Here, we list some map operations as the example in this program.

First, add five command buttons to form1, and then set their properties as follows (Set

the properties of others by default.), **Table 2-1**

Table 2-1 *Button properties*

Name	Caption
cmdPan	Pan
cmdZoomIn	ZoomIn
cmdZoomOut	ZoomOut
cmdZoomFree	ZoomFreely
cmdViewEntire	FullExtent

Then, add below code to each Click event to implement corresponding function.

```
Private Sub cmdPan_Click()  
    SuperMap1.Action = scaPan  
End Sub  
  
Private Sub cmdViewEntire_Click()  
    SuperMap1.ViewEntire  
End Sub  
  
Private Sub cmdZoomFree_Click()  
    SuperMap1.Action = scaZoomFree  
End Sub  
  
Private Sub cmdZoomIn_Click()  
    SuperMap1.Action = scaZoomIn  
End Sub  
  
Private Sub cmdZoomOut_Click()  
    SuperMap1.Action = scaZoomOut  
End Sub
```

The following picture shows the map after zoomed in, *Figure 2-4*

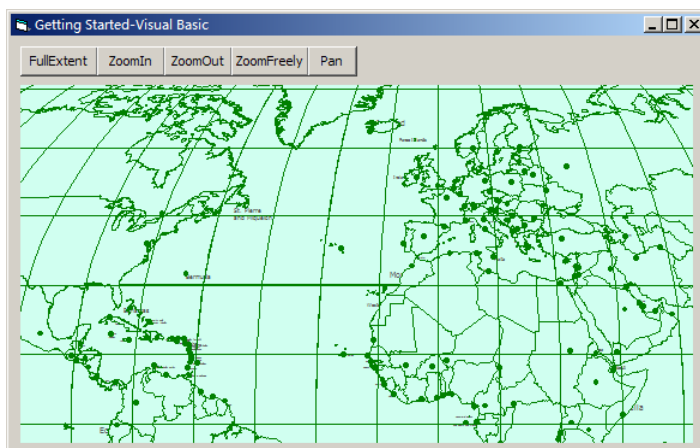


Figure 2-4 The result of map zoom in

2.5 Querying Properties

First, add a command button to form1 and then set its property as described in the following table, **Table 2-2**

Table 2-2 Button property

Name	Caption
cmdSelect	Identify

Second, add below code in Click event to implement related functions:

```
Private Sub cmdSelect_Click ()
    SuperMap1.Action = scaSelect
End Sub
```

Finally, add the following code to the GeometrySelected event of SuperMap1:

```
Private Sub SuperMap1_GeometrySelected(ByVal nSelectedGeometryCount As Long)
    Dim objRecordSet As soRecordset      'A recordset object
    Dim i As Integer                    'A cycle variable
    Dim strName(40) As String           'For storing field names
    Dim strValue(40) As String         'For storing field values
    Dim strMessage As String           'Showing all field informations
```

```
'Get the properties of the selected object
Set objRecordSet = SuperMap1.selection.ToRecordset(False)
objRecordSet.MoveFirst ' Move to the first record
For i = 1 To objRecordSet.FieldCount
    'Get the field name
    strName(i - 1) = objRecordSet.GetFieldInfo(i).Name
    'Get the field value
    strValue(i - 1) = objRecordSet.GetFieldValue(i)
Next

strMessage = ""
For i = 1 To objRecordSet.FieldCount
    strMessage = strMessage & strName(i - 1) & ": " & strValue(i - 1) & Space(5) & vbCrLf
Next
MsgBox strMessage

'Release memory
Set objRecordSet = Nothing
End Sub
```

The following picture shows the attributes from the selected object in the map, *Figure 2-5*

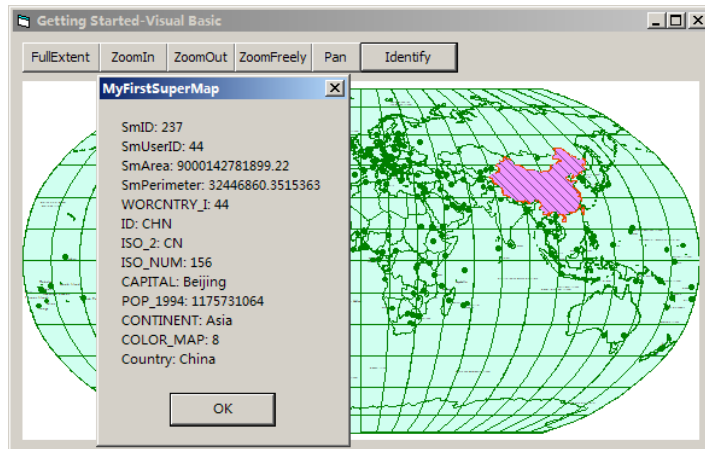


Figure 2-5 The result of identifying map

2.6 Querying Map

1. Add a command button and a textbox to form1, and then set their properties as follows, **Table 2-3**

Table 2-3 Controls' properties

Control	Name	Caption
Button	cmdQueryMap	QueryMap
TextBox	txtExpression	---

2. Add the following code to Click event of button:

```

Private Sub cmdQueryMap_Click()
    Dim objDtVector As soDatasetVector           'Define a vector dataset object
    Dim objRecordSet As soRecordset             'Define a recordset object
    Dim objSelection As soSelection             'Define a selection object

    'Get the vector dataset "World_countries" on which an attribute query would be performed
    Set                                     objDtVector                                     =
    SuperWorkspace1.Datasources.Item("MyDataSource").Datasets("World_countries")

    If objDtVector Is Nothing Then
        MsgBox "Failed to open the specified dataset!", vbInformation
    Exit Sub

```

```
End If

'Query properties from the dataset (The 'Query' method can only be applied to the object of
soDatasetVector.)
Set objRecordSet = objDtVector.Query(txtExpression.Text, True)
If objRecordSet Is Nothing Then
    Exit Sub
Else
    'Add the query result to current selection
    Set objSelection = SuperMap1.selection
    objSelection.FromRecordset objRecordSet
    'Refresh the map
    SuperMap1.Refresh
End If

Set objDtVector = Nothing
Set objRecordSet = Nothing
Set objSelection = Nothing
End Sub
```

3. Type a SQL expression, e.g. `smid>50`, in the Query text box, and Click 'QueryMap', then the results will be displayed as follows, **Figure 2-6**

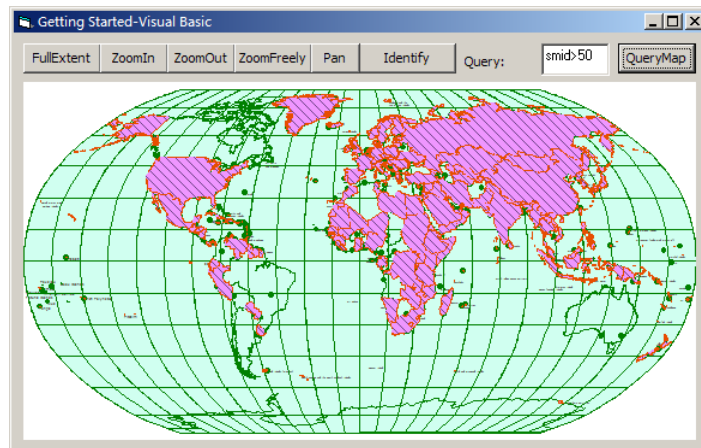


Figure 2-6 The result of map querying

Chapter

3

Using SuperMap Objects in Delphi

3.1 Creating a new project

1. Creating a working directory such as C:\MyProject.
2. Download the data compression package World.zip (Include World.sdb and World.sdd) to the working directory (C:\MyProject).
3. Start Delphi.
4. Creating a new project MySuperMap at working directory (C:\MyProject) and name the form1 as frmmap.

3.2 Loading SuperMap Objects controls

1. Add the ActiveX controls to the ActiveX tab of the Component Palette.

In Delphi, open the package file SuperMap5.dpk the special file provided for developer

to load controls more convenient and usually located at the SuperMap installation directory (\Bin\InterfaceClass\), then the Package dialog box will be displayed as follows, **Figure 3-1**

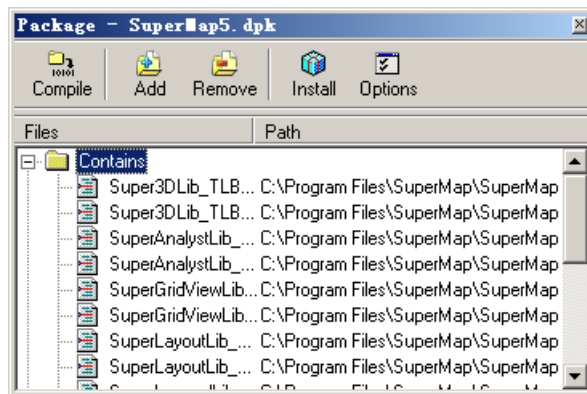


Figure 3-1 Package

If the earlier version's controls have already existing in Delphi package, you can remove them and reload the new package file, then click 'Install'.

After the controls are installed successfully, the icons of SuperMap controls will be listed on the ActiveX tab of Component Palette as follows, **Figure 3-2**

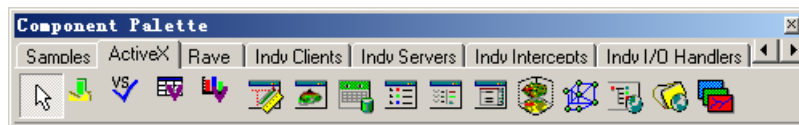


Figure 3-2 Component Palette

If this is the first time for you to develop programs in Delphi environment using SuperMap Objects, you should set the search path of the type library by opening Project->Options. In detail, select the Directories/conditionals tab of the Options page, and navigate the path to SuperMap Objects installation directory (/Bin/InterfaceClass) in the Search Path box. Now SuperMap Objects has been completely installed.

2. Add SuperWorkspace control and SuperMap control to form1.

3.3 Opening a map and adding layers

Add the following code to the unit:

```

procedure Tfrmmap.FormCreate(Sender: TObject);
var
  strAlias:String ; // Alias of datasource
  nEngineType:seEngineType; // Type of data engine
  strDataSourceName:String; // Absolute path of datasource
  objDataSource:soDataSource; //Datasource object point to the open datasource
  bReadOnly:Boolean; // Whether the data of datasource can read only
  objLayer:soLayer; // The variable of layer object refer to open layer
  bAddToHead:Boolean; // Whether adding to head or not
  i:Integer; // Cycle variable
begin
  // Establish the relationg between supermap and SuperWorkspace
  supermap1.Connect(superworkspace1.handle);
  strAlias:='MyDatasource'; // Alias can be named arbitrarily in principle but you 'd better advise
  that the main name same to datasource file
  nEngineType:=sceSDBPlus; //SuperMap can support various type , here is SDB+
  strDataSourceName:='c:\myproject\world.sdb';
  bReadOnly:=false; // Not set read only
  // Open datasource

  objDataSource:=SuperWorkspace1.OpenDataSource(strDataSourceName,strAlias,nEngineType,bRe
  adOnly);
  if objDataSource=nil then
    begin
      MessageBox(frmmmap.handle, 'Failed to open the datasource! ', 'SuperMap Objects 5.2 Getting
      Started', MB_OK);
      MessageBox(frmmmap.handle,'If you did not download the datasource,Please download sample
      data(world.sdb,world.sdd) to current directory,Thanks!', 'SuperMap Objects 5.2 Getting Started',
      MB_OK);
    end
  else
    begin
      // Add all layers of datasource to Supermap
      for i:=1 to objDataSource.Datasets.Count do
        begin
          bAddToHead:=true; // Adding to head
          objLayer:=SuperMap1.Layers.AddDataset(objDataSource.Datasets.Item[i], bAddToHead);
        end;
      // Refresh map
      Supermap1.Refresh;
      // Modify selection style
      supermap1.selection.Style.PenColor := RGB(231, 77, 0);
    end;
  end;
end;

```

```
supermap1.selection.Style.PenWidth := 1;
supermap1.selection.Style.PenStyle := 1;
supermap1.selection.Style.BrushStyle := 5;
supermap1.selection.Style.BrushColor := RGB(115, 69, 140);
supermap1.selection.Style.BrushBackColor := RGB(239, 150, 255);
supermap1.selection.Style.BrushOpaqueRate := 50;
end
end;
procedure Tfrmmap.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  // Close workspace
  SuperMap1.Disconnect;
  SuperMap1.Close;
  SuperWorkspace1.Close;
  SuperMap1.Free;
  SuperWorkspace1.Free;
end ;
```

Run the program, then the result will be displayed as follows, *Figure 3-3*

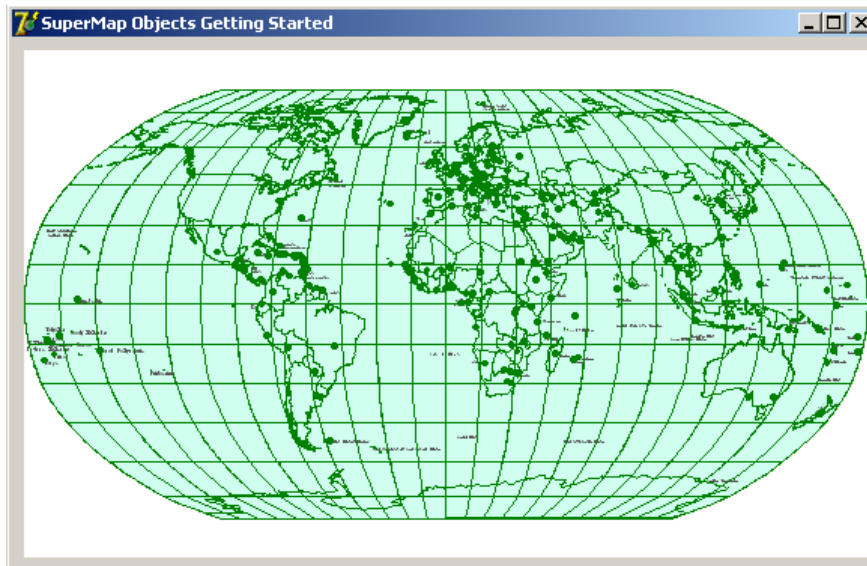


Figure 3-3 *Showing map in its full extent*

3.4 Browsing map

It is very convenient to perform map operations with SuperMap Objects, such as

zooming in, zooming out, zooming freely, panning map, showing the full extent of map, drawing points or lines, etc. Here, we list some map operations as the example in this program.

First, add five common buttons into form1, and then set their caption properties as follows, **Table 3-1**

Table 3-1 *Button properties*

Name	Caption
btPan	Pan
btZoomIn	ZoomIn
btZoomOut	ZoomOut
btZoomFree	ZoomFreely
btViewEntire	FullExtent

Then, add the code below to each Click event to implement the corresponding function:

```
procedure Tfrmmap.btViewEntireClick(Sender: TObject);
begin
    // View Entire
    supermap1.ViewEntire;
end;
application Tfrmmap.btZoomInClick(Sender: TObject);
begin
    // Zoom in
    Supermap1.Action:=scaZoomIn;
end ;
procedure Tfrmmap.btZoomOutClick(Sender: TObject);
begin
    // Zoom out
    Supermap1.Action:=scaZoomOut;
end ;
procedure Tfrmmap.btZoomFreeClick(Sender: TObject);
begin
    // Zoom free
    Supermap1.Action:=scaZoomFree;
end ;
procedure Tfrmmap.btPanClick(Sender: TObject);
begin
    // Pan
```

```
Supermap1.Action:=scaPan;
end ;
```

The following picture shows the result of the Zoom In operation, *Figure 3-4*.

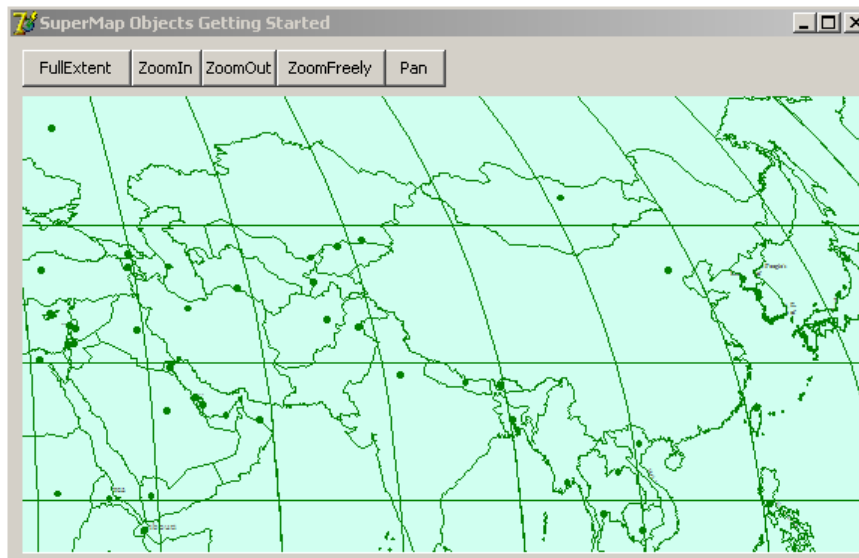


Figure 3-4 *The result of map zoom in*

3.5 Querying Properties

First, add a command button to form and then set relative properties as below, **Table 3-2**

Table 3-2 *Button properties*

Name	Caption
btProperty	Identify

Second, add the following code to the Click event to implement the associated function:

```
procedure Tfrmmap.btPropertyClick(Sender: TObject);
begin
  // Query property
  Supermap1.action:=scaSelect;
end ;
```

Finally, add the following code to the GeometrySelected event of SuperMap1:

```
procedure Tfrmmap.SuperMap1GeometrySelected(ASender: TObject;
nSelectedGeometryCount: Integer);
var
  objRecordSet:soRecordset; // Property object
  i,j:Integer; // Cycle variable
  strName:array[1..4] of string; // Storage property name
  strValue:array[1..4] of String; // Storage property value
  strMessage:String; // Show all message
begin
  // Convert the selection object to dataset object
  objRecordSet:=Supermap1.Selection.ToRecordset(false);
  objRecordSet.MoveFirst;
  // Read feild count
  j:=objRecordSet.FieldCount;
  // If the dataset is more than four fields , only read front four feilds
  if j>4 then
    j:=4;
  // Only display the front four fields
  for i:=1 to j do
    begin
      strName[i]:=objRecordSet.GetFieldInfo(i).Name;
      strValue[i]:=objRecordSet.GetFieldValue(i);
    end;
  strMessage:="";
  for i:=1 to j do
    strMessage:=strMessage+strName[i]+ ':'+strValue[i]+' '+ chr(VK_return);
  showMessage(strMessage);
end ;
```

The following picture shows the result of querying properties from an object in the map,
Figure 3-5

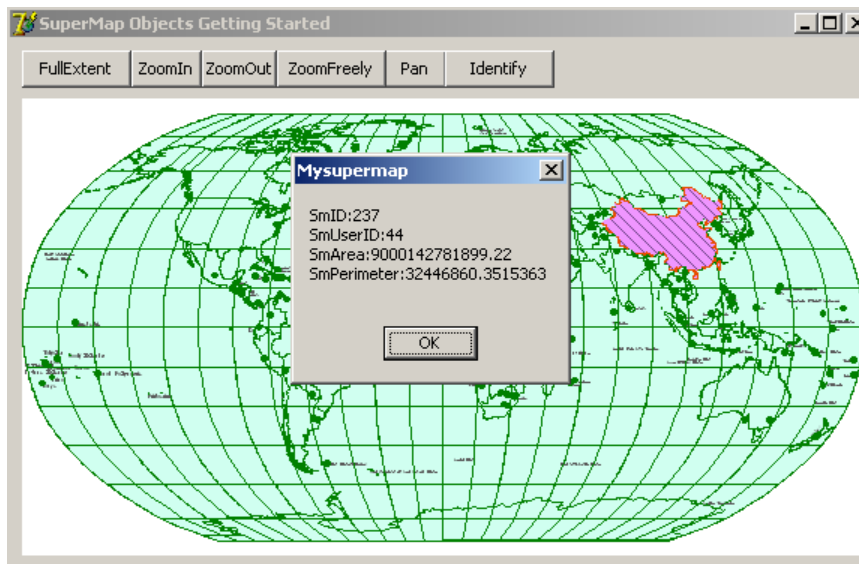


Figure 3-5 The result of identifying map

3.6 Querying map

1. Add a command button and a textbox to form, and then set their properties as follows, **Table 3-3**

Table 3-3 The controls properties

Control	Name	Caption
Button	btQueryMap	QueryMap
Textbox	editExpression	---

2. Add the following code to the Click event of the button to perform the associated function:

```

procedure Tfrmmap.btQueryMapClick(Sender: TObject);
var
  objDataSource:soDatasource; //Datasource
  objDtVector:soDatasetVector; //Vector dataset variable
  objRecordset:soRecordset; //Attribute dataset variable
  objSelection:soSelection; //Selection variable
  
```

```

strOptions:widestring;
objerrors:soerror;
objGeo:soGeometry;
begin
  //Set Action of Supermap1 as null
  Supermap1.Action:=scaNull;
  if SuperWorkspace1.Datasources.count=0 then
    exit;
  //Get the vector dataset
  objDataSource:=SuperWorkspace1.Datasources.Item[1];
  if objDataSource= nil then
    begin
      showMessage('Error to datasource'+objerrors.LastErrorMsg );
      Exit;
    end;
  objDtVector:=objDataSource.Datasets.Item['World_countries'] as soDataSetVector;
  if objDtVector=nil then
    begin
      showMessage('Error to open dataset'+objerrors.LastErrorMsg );
      Exit
    end;
  try
    //Query attribute data from dataset
    objRecordset:=objDtVector.Query(editExpression.Text, true, nil, '') as soRecordset;
    if objRecordset=nil then
      exit
    else
      begin
        //Add the data which have been queried to selection
        objSelection:=SuperMap1.Selection;
        objSelection.FromRecordset(objRecordset);
        //Refresh the map
        SuperMap1.Refresh ;
        exit;
      end;
    except
      MessageBox(frmmmap.Handle,'Failed to query!','Error',MB_RETRYCANCEL+MB_ICONERROR)
    end;
  end;
end;

```

3. Type a SQL expression, e.g. `smid>50`, in the Query text box, and Click 'QueryMap', then the result will be displayed as follows, **Figure 3-6**



Figure 3-6 *The result of map querying*

Chapter

4

Using SuperMap Objects in Visual C++

4.1 Creating a new project: MySuperMap

1. Creating a working directory (C:\Myproject)
2. Download the data compression package World.zip (Include World.sdb and World.sdd) to the working directory (C:\MyProject)
3. Start Visual C++ 6.0 and create a new project. Choose menu File -> New, Click the Projects tab in the dialog box and select MFC AppWizard(exe); locate the file path to (C:\Myproject) to save the project, and type MySuperMap as the project name, finally Click OK to continue, *Figure 4-1*

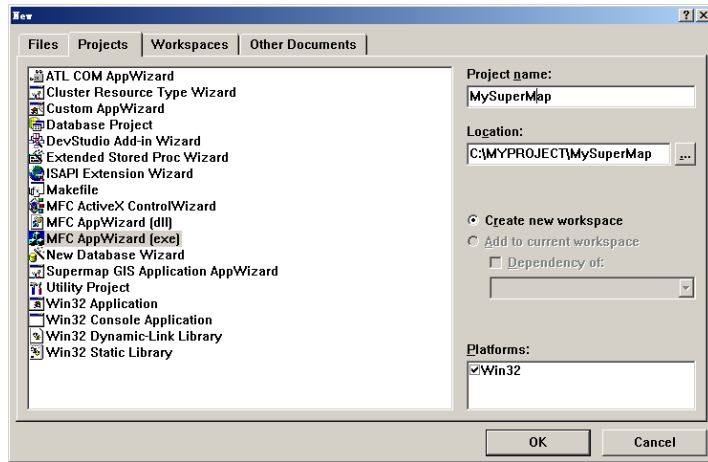


Figure 4-1 Create a new project

4. Select the type of application. Just do as what is shown in the following dialog box, select the third option 'Dialog based', then click the Finish button to complete the creation of a dialog based project, *Figure 4-2*

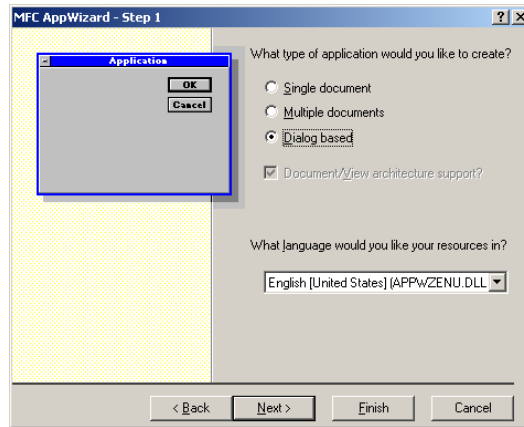


Figure 4-2 MFC AppWizard

4.2 Loading SuperMap Objects control

Loading SuperMap Objects control to project:

1. Click Project-> “Add To Project” -> “Components and Controls ...”, **Figure 4-3**

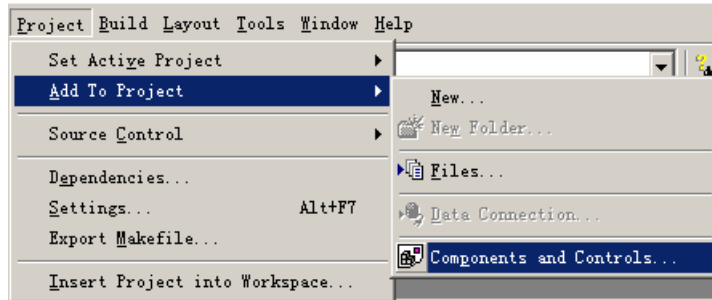


Figure 4-3 Add to project

2. Double click “Registered ActiveX Controls” directory, select “SuperMap Control”. Dialog box is displayed as follows, **Figure 4-4**

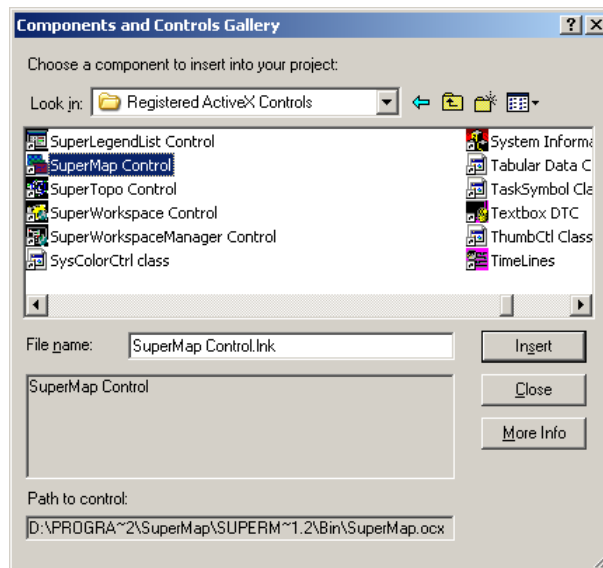


Figure 4-4 Components and Controls Gallery

3. Click the "Insert" button, and click OK in the message box that will pop up to ask whether to insert this component, then the Confirm Classes dialog box will be displayed as follows. Click OK in the dialog box, **Figure 4-5**

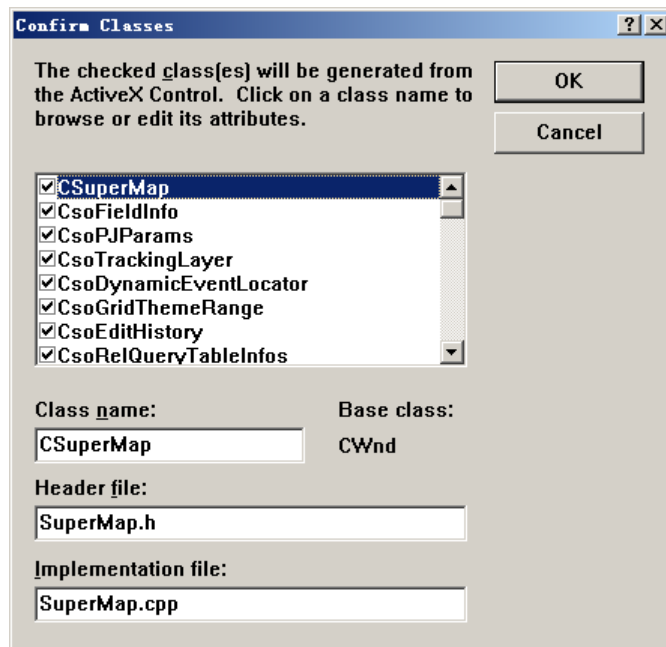


Figure 4-5 *Confirm Classes Dialog box*

Now SuperMap Control has been loaded into the project, you can load “SuperWorkspace Control” in the same way. When close control option dialog box, there are two loaded controls in the ToolBox, **Figure 4-6**



Figure 4-6 *Controls box*

4. Through above steps, some classes still can not be loaded, we continue choose the

menu View -> ClassWizard... as *Figure 4-7*

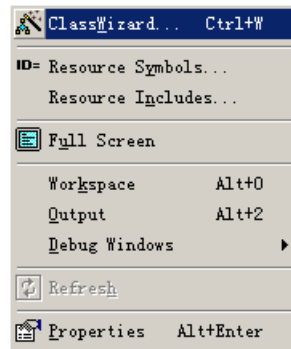


Figure 4-7 *ClassWizard menu*

In MFC ClassWizard dialog box, click Add Class -> From a Type Library, as *Figure 4-8*

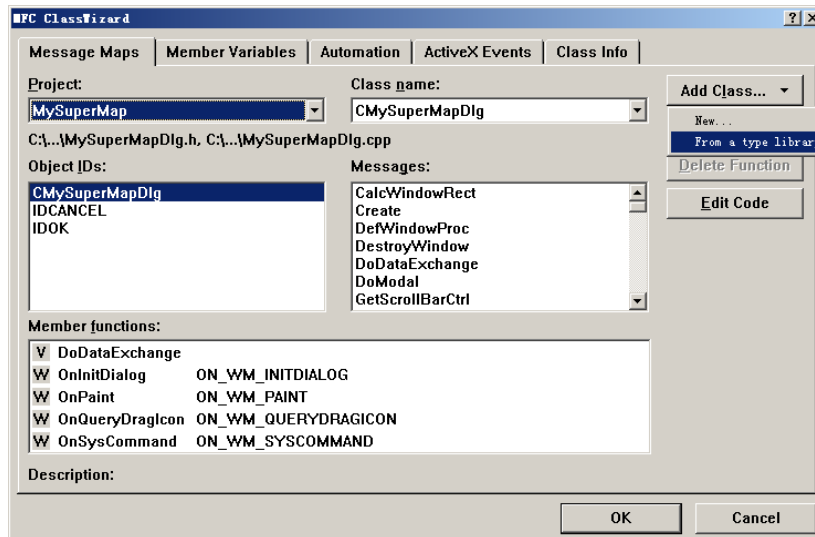


Figure 4-8 *MFC ClassWizard dialog box*

Then select SuperMap.tlb in the installation directory (Bin\TypeLibrary\SuperMap.tlb), *Figure 4-9*

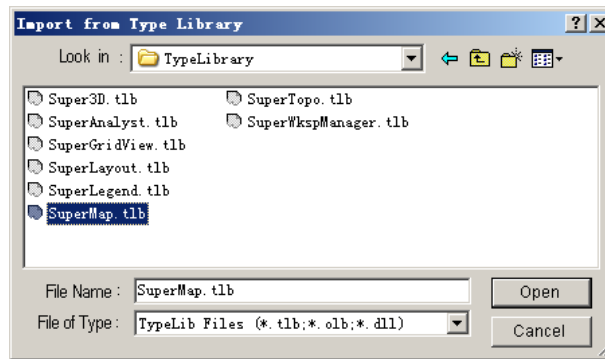


Figure 4-9 *Import from Type library*

When the required .tlb file was opened, all classes will be listed as below. Now, we can select all of them and click OK. **Figure 4-10**

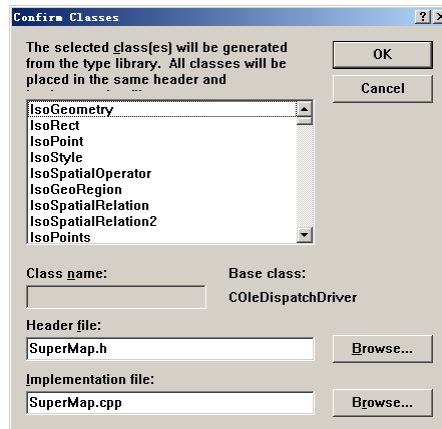


Figure 4-10 *Confirm Classes*

4.3 Opening a map and adding layers

1. Add a SuperWorkspace control to the current dialog (If the dialog is not open, you can find it by its ID that is IDD_MYSUPERMAP_DIALOG in the ResourceView tab of the VC workspace manager. And open it after found), name its ID as IDC_SuperWorkspace; add a SuperMap control to the dialog box and name its ID as

IDC_SuperMap.

2. Right-click "SuperMap" control on the dialog box, and choose "ClassWizard ..." as follows, **Figure 4-11**

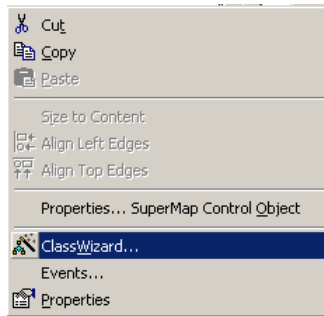


Figure 4-11 Right key menu

Choose Member Variables on "MFC ClassWizard" dialog box and double click IDC_SuperMap in the Control IDs list box; or click IDC_SuperMap, and then click "Add Variable..." to connect a variable to the SuperMap control. Do just as the following, **Figure 4-12**

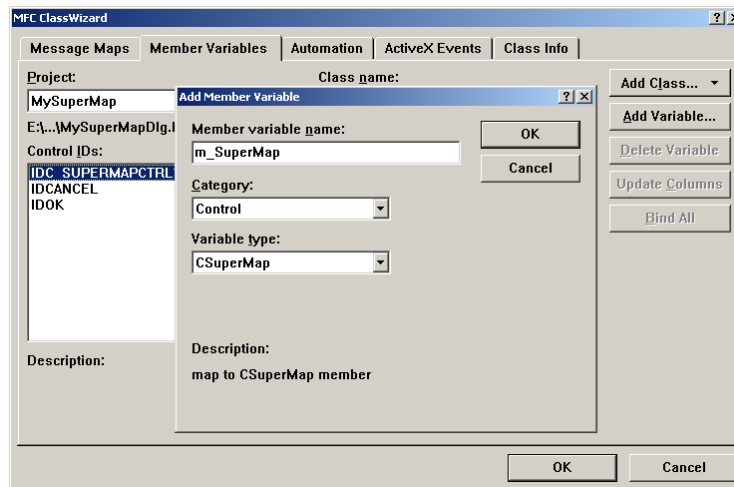


Figure 4-12 Add member variable

Click OK; the added variable will be listed in the "MFC ClassWizard" dialog box showing as follows. Then you can connect another variable m_SuperWorkspace to the SuperWorkspace control IDC_SuperWorkspace in the same way. At last click OK to close "MFC ClassWizard" dialog box, **Figure 4-13**

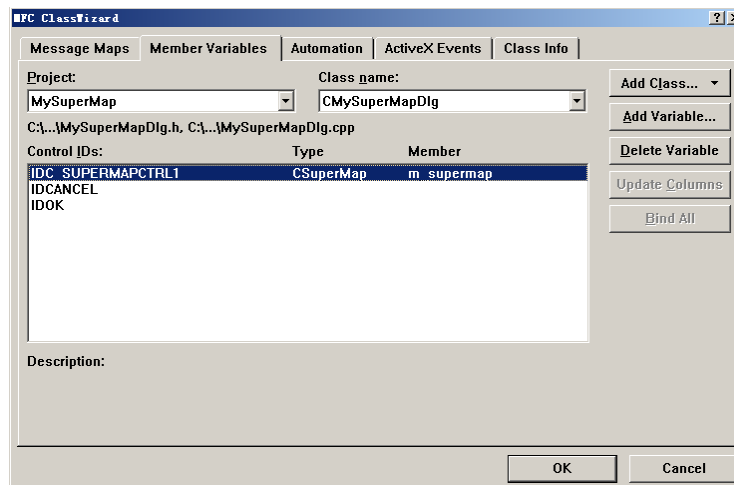


Figure 4-13 Relating "IDC_SuperWorkspace" into "m_SuperWorkspace"

3. Add required head files on the top of "MySuperMapDlg.cpp"

```
#include "sodatasource.h"
#include "sodatasources.h"
#include "sorecordset.h"
#include "sofieldinfo.h"
#include "sodatasesets.h"
#include "sodataset.h"
#include "solayers.h"
#include "solayer.h"
#include "soselection.h"
#include "sodatasetvector.h"
#include "sostyle.h"
```

4. Add the following code before the "return TRUE;" sentence which is in the CMySuperMapDlg::OnInitDialog function:

```
// Open map and display
CsoDataSource objDataSource;
CsoDatasesets objDataSets;
long Index;
//Establish the relation between workspace and datasource
LPDISPATCH handle = m_SuperWorkspace.GetHandle ();
m_SuperMap.Connect(handle);
handle->Release();
// Open datasource,14 express SDB+ engine
objDataSource=m_SuperWorkspace.OpenDataSource("C:\\MyProject\\world.sdb","world",14,false);
if (! objDataSource)
{
    MessageBox("Error to open datasource!");
    m_SuperMap.Close();
    m_SuperMap.Disconnect();
    m_SuperWorkspace.Close();
    return FALSE;
}
objDataSets=objDataSource.GetDataSets();
//Add layers
for(Index=1;Index<=objDataSets.GetCount();Index++)
{
    m_SuperMap.GetLayers().AddDataset(objDataSets.GetItem(COLEVariant(Index)),false);
}
//Refresh,display
m_SuperMap.Refresh();
//Modify the style of selection object
CsoStyle objStyle = m_SuperMap.GetSelection().GetStyle();
objStyle.SetPenColor(RGB(231,77,0));
objStyle.SetPenStyle(1);
objStyle.SetPenWidth(1);
```

```
objStyle.SetBrushStyle(5);
objStyle.SetBrushColor(RGB(115,69,140));
objStyle.SetBrushBackColor(RGB(239,150,255));
objStyle.SetBrushOpaqueRate(50);
m_SuperMap.GetSelection().SetStyle(objStyle);
Add "WM_CLOSE" message handling function "OnClose" to dialog box, and then add code before
"CDialog::OnClose() ":
//Close windows and workspace, noted the right order
m_SuperMap.Disconnect();
m_SuperMap.Close();
m_SuperWorkspace.Close();
```

Add "WM_CLOSE" message handling function "OnClose" to dialog box, and then add code before "CDialog::OnClose() ":

```
//Close the SuperMap and the SuperWorkspace. Please notice the operation order
m_SuperMap.Close ();
m_SuperMap.Disconnect ();
m_SuperWorkspace.Close();
```

Compile the code, the result will be displayed as follows, **Figure 4-14**

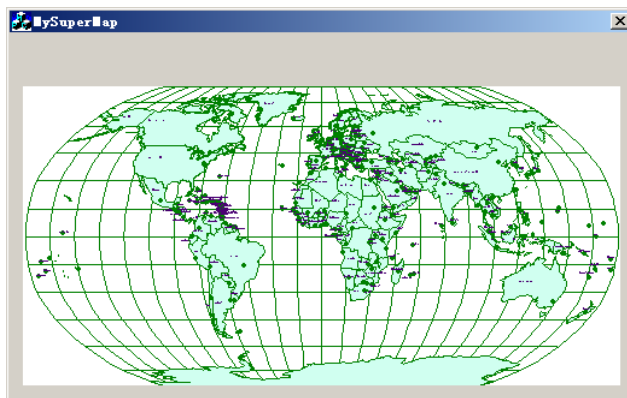


Figure 4-14 *The result interface*

4.4 Browsing map

It is very convenient to perform map operations with SuperMap Objects, such as zooming in, zooming out, zooming freely, panning map, showing the full extent of map,

drawing points or lines, etc. Here, we list some map operations as the example in this program.

First, add five Command Buttons in Form1 and modify their Captions as below, **Table 4-1**

Table 4-1 *Buttons properties*

ID	Caption
IDC_ButtonPan	Pan
IDC_ButtonZoomIn	ZoomIn
IDC_ButtonZoomOut	ZoomOut
IDC_ButtonZoomFree	ZoomFreely
IDC_ButtonViewEntire	FullExtent

5. Then, add code to each Click event to implement corresponding functions:

```
void CMySuperMapDlg::OnButtonPan()
{
    //Pan
    m_SuperMap.SetAction(1);
}
void CMySuperMapDlg::OnButtonZoomIn()
{
    //Zoom in
    m_SuperMap.SetAction (2);
}
void CMySuperMapDlg::OnButtonZoomOut()
{
    //Zoom out
    m_SuperMap.SetAction (3);
}
void CMySuperMapDlg::OnButtonZoomFree()
{
    //Zoom free
    m_SuperMap.SetAction(4);
}
void CMySuperMapDlg::OnButtonViewEntire()
{
    //View entire
    m_SuperMap.ViewEntire();
}
```

The following picture shows the result of map Zoom In operation, **Figure 4-15**

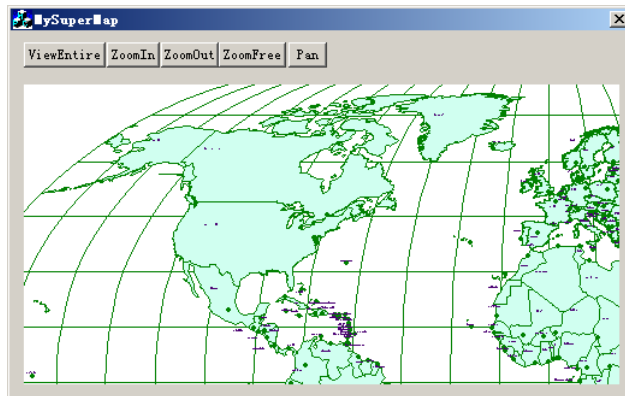


Figure 4-15 The result of map zooming in

4.5 Querying Properties

1. Add a button to Form and set its Caption as follow, **Table 4-2**

Table 4-2 Button property

ID	Caption
IDC_ButtonQueryProperties	Identify

2. Add the code to respond the Identity command:

```
//Identity
m_SuperMap.SetAction(5);
```

3. Add the function "OnGeometrySelectedSuperMap(long nSelectedGeometryCount)" to respond the "GeometrySelected" control by using the "MFC ClassWizard" dialog box. Then add the following code to the "OnGeometrySelectedSuperMap(long nSelectedGeometryCount)" function:

```
CsoRecordset Record;
CsoFieldInfo Info;
long Index;
COleVariant var;
```

```

CString strName;
CString strValue;
CString strMsg;
//Convert to property recordset
Record=m_SuperMap.GetSelection().ToRecordset(false);
strMsg="";
//Get Property name and value
for(Index=1;Index<=Record.GetFieldCount();Index++)
{
Info=Record.GetFieldInfo(COLEVariant(Index));
strName=Info.GetName();
var=Record.GetFieldValue(COLEVariant(Index));
var.ChangeType(VT_BSTR);
strValue=var.bstrVal;
strMsg =strMsg + strName + ":" + strValue + "\n";
}
//Display property
MessageBox(strMsg);

```

Run the code, click the Identity button, then select an object on the map, finally the properties of the selected object will be displayed as follows, **Figure 4-16**

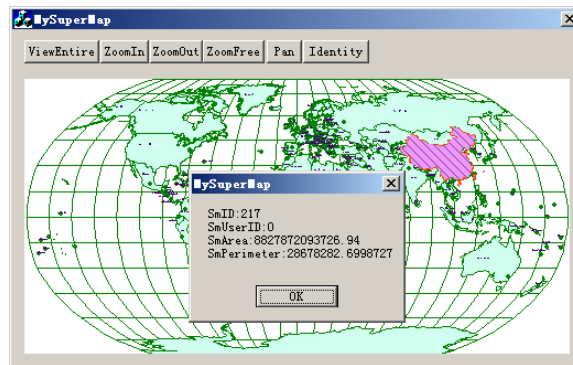


Figure 4-16 The result of Identity map

4.6 Querying map

1. Add a button and a TextBox to Form1 and set their properties as follow, **Table 4-3**

Table 4-3 Controls properties

Controls	ID	Caption
----------	----	---------

Button	IDC_ButtonQueryMaps	QueryMap
Edit Box	IDC_Edit	---

2. Add the responded function of the IDC_ButtonQueryMaps button to the dialog box, then add the following code to the function, finally connect a string variable named "m_QueryCondition" to "IDC_Edit" using the "MFC ClassWizard":

```
//Query map
UpdateData(true);
CsoDatasetVector objDtVector; //Vector dataset
CsoDataset objDt;
CsoRecordset objRecordset; //Attribute dataset variable
CsoDatasets objDtSets;
CsoDataSource objDataSource;
IDispatch *ar = NULL;
char *dd = NULL;
//Get the vector dataset
objDataSource = m_SuperWorkspace.GetDatasources().GetItem(COLEVariant(1L));
objDtSets = objDataSource.GetDatasets();
objDt = objDtSets.GetItem(COLEVariant(3L));
//The second parameter is false, avoiding objDt to be release
objDtVector.AttachDispatch(objDt,false);
//Query attribute data from dataset
objRecordset = objDtVector.Query(m_QueryCondition ,true,ar,dd);
//Add the data which have been queried to selection
m_SuperMap.GetSelection ().FromRecordset(objRecordset);
// Refresh map and close recordset
objRecordset.Close();
m_SuperMap.Refresh();
```

3. Run the program and type a SQL expression, e.g. `smid > 50` , in the Query text box, then click the QueryMap button, finally the result will be displayed as follows,

Figure 4-17

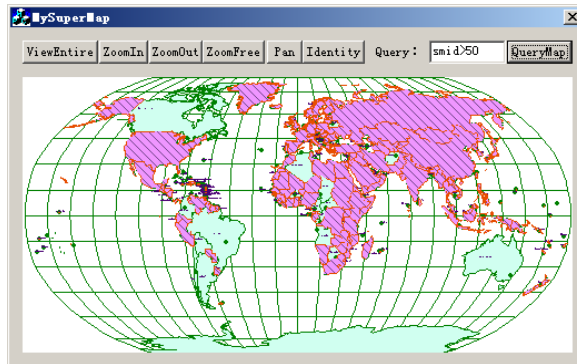


Figure 4-17 *The result of map querying*

Chapter

5

Using SuperMap Objects in VB .NET

5.1 Creating a new project: MySuperMap

1. Create a working directory (C:\MyProject)
2. Download the data compression package World.zip (Include World.sdb and World.sdd) to the working directory (C:\MyProject)
3. Start Visual Studio.NET
4. Create a new Visual Basic project (MySuperMap) at the working directory (C:\MyProject)

5.2 Loading SuperMap Objects controls

1. Add SuperMap Objects controls to ToolBox:
 - Right-click the Toolbox and click 'Add Tab', and name the new tab as SuperMap.

- Right-click the Toolbox and click 'Choose Items...', then the Customize Toolbox will be displayed as follows, *Figure 5-1*

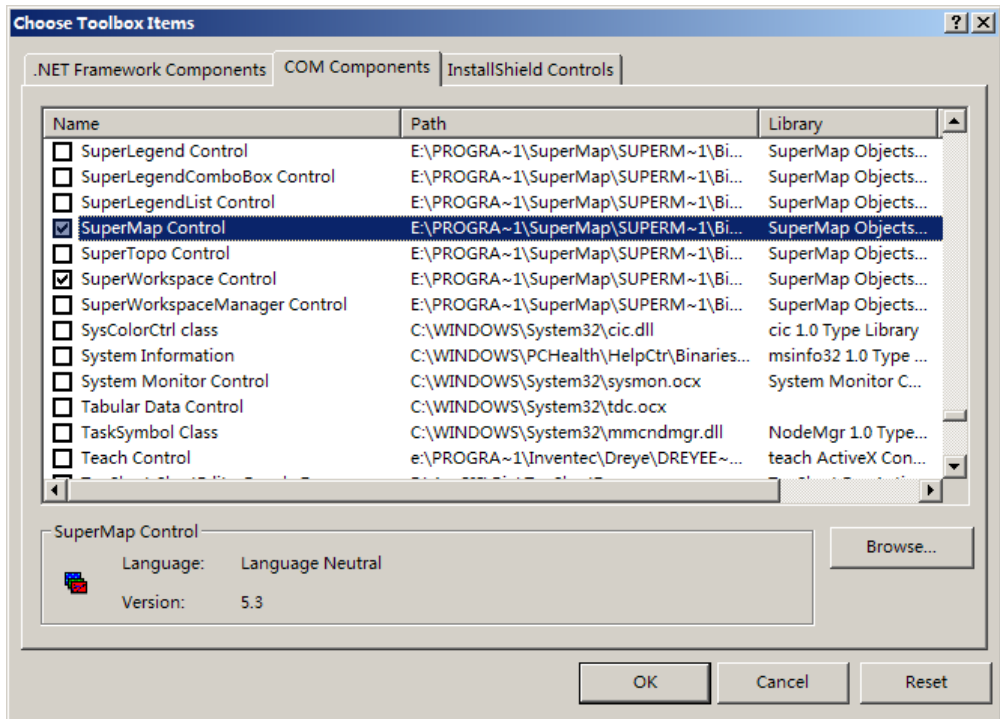
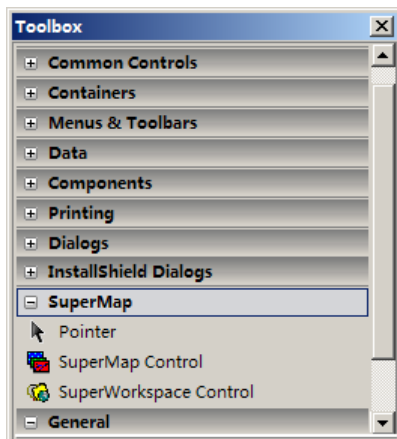


Figure 5-1 Load SuperMap control and SuperWorkspace control

- Select 'SuperMap Control and SuperWorkspace Control' in the COM Components tab (by selecting the checkbox before them) and click OK. Then two controls will be added to the Visual Basic toolbox showing as follows:

2. Rename the window form as FrmMain, and add the two controls to the form and name them as AxSuperMap1 and AxSuperWorkspace1 respectively, *Figure 5-2*

Figure 5-2 *Toolbox*

5.3 Opening a map and adding layers

Add the following code into window form:

```
Private Sub frmMap_Load(ByVal eventSender As System.Object, ByVal eventArgs As
System.EventArgs) Handles MyBase.Load

    ' Connect SuperWorkspace and SuperMap
    ' Handle is name of the system property in Visual Basic.NET, so CtlHandle will be used for
    SuperWorkspace handle

    AxSuperMap1.Connect(AxSuperWorkspace1.CtlHandle)

    Dim strAlias As String ' Alias of datasource
    Dim nEngineType As SuperMapLib.seEngineType 'Engine type of data being used in this
sample
    Dim strDataSourceName As String 'Absolute path of datasource
    Dim objDataSource As SuperMapLib.soDataSource 'Datasource object

    Dim bReadOnly As Boolean 'Whether the datasource to be opened is read only
    Dim objLayer As SuperMapLib.soLayer 'Layer object of SuperMap window
    Dim bAddToHead As Boolean 'Whether a new added layer will be brought to front
    Dim i As Short ' A cycle variable

    strAlias = "MyDataSource" 'The Alias can be named arbitrarily, but you'd better use the same
one as that of the datasource file
    nEngineType = SuperMapLib.seEngineType.sceSDBPlus 'We will use a SDBPlus data in this
```

```
sample
    strDataSourceName = "c:\MyProject\World.sdb" 'You can change this path name according to
the particular place of your data, or you can use a relative path.
    bReadOnly = False ' Indicates the datasource to be opened can be modified

    ' Open datasource
    objDataSource = AxSuperWorkspace1.OpenDataSource(strDataSourceName, strAlias,
nEngineType, bReadOnly)
    If objDataSource Is Nothing Then
        MsgBox("Failed to open datasource!", MsgBoxStyle.Information)
        MsgBox("Please download the datasource files (world.sdb, world.sdd) to the directory
C:\MyProject\World.sdb, then run the program!")
    Else
        For i = 1 To objDataSource.Datasets.Count
            ' Add all layers of the datasource to SuperMap
            bAddToHead = True
            objLayer = AxSuperMap1.Layers.AddDataset(objDataSource.Datasets.Item(i),
bAddToHead)
        Next
    End If

    ' Refresh the map window
    ' Handle is the name of system property in Visual Basic.NET, so CtlHandle will be used for
SuperWorkspace handle
    AxSuperMap1.Refresh()

    'Modify styles of the selected object
    AxSuperMap1.selection.Style.PenColor = System.Convert.ToUInt32(RGB(231, 77, 0))
    AxSuperMap1.selection.Style.PenWidth = 1
    AxSuperMap1.selection.Style.PenStyle = 1
    AxSuperMap1.selection.Style.BrushStyle = 5
    AxSuperMap1.selection.Style.BrushColor = System.Convert.ToUInt32(RGB(115, 69, 140))
    AxSuperMap1.selection.Style.BrushBackColor = System.Convert.ToUInt32(RGB(239, 150,
255))
    AxSuperMap1.selection.Style.BrushOpaqueRate = 50

    ' Release memory
    objDataSource = Nothing
    objLayer = Nothing

End Sub
```

Run above code, the result will be displayed as follows, **Figure 5-3**

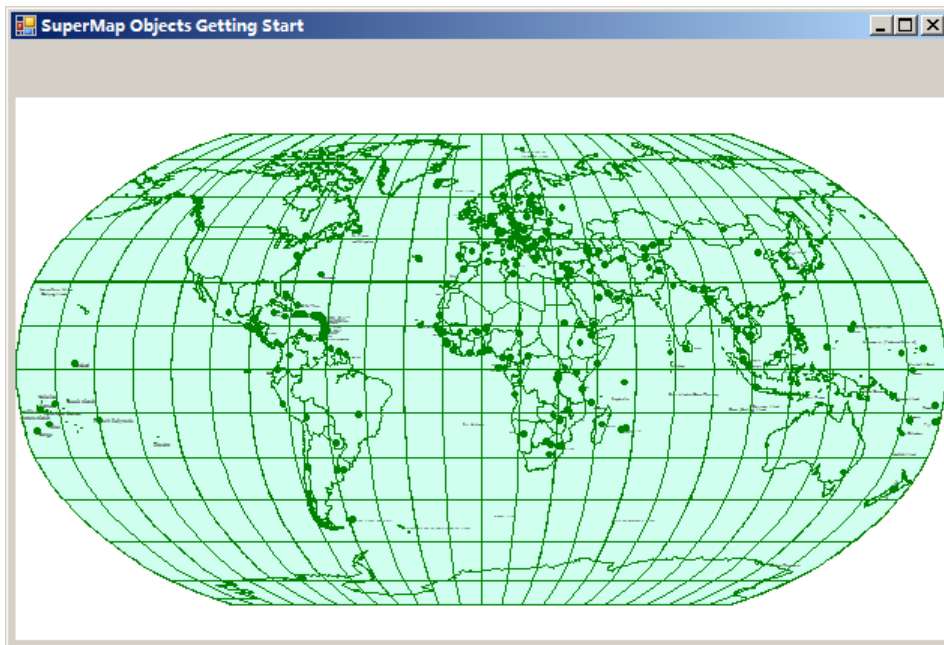


Figure 5-3 *The result interface*

5.4 Browsing map

It is very convenient to perform map operations with SuperMap Objects, such as zooming in, zooming out, zooming freely, panning map, showing the full extent of map, drawing points or lines, etc. Here, we list some map operations as the example in this program.

First, add five command buttons to form1, and then set their properties as follows, **Table 5-1**

Table 5-1 *Button properties*

Name	Caption
cmdPan	Pan
cmdZoomIn	ZoomIn

cmdZoomOut	ZoomOut
cmdZoomFree	ZoomFreely
cmdViewEntire	FullExtent

Then, add the code to each Click event to implement corresponding functions:

```
Private Sub cmdFullExtent_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles cmdFullExtent.Click
    AxSuperMap1.ViewEntire() 'Show full extent of map
End Sub

Private Sub cmdZoomIn_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles cmdZoomIn.Click
    AxSuperMap1.Action = SuperMapLib.seAction.scaZoomIn 'Zoom map in
End Sub

Private Sub cmdZoomOut_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles cmdZoomOut.Click
    AxSuperMap1.Action = SuperMapLib.seAction.scaZoomOut 'Zoom map out
End Sub

Private Sub cmdZoom_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles cmdZoom.Click
    AxSuperMap1.Action = SuperMapLib.seAction.scaZoomFree 'Zoom map in/out
End Sub

Private Sub cmdPan_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles cmdPan.Click
    AxSuperMap1.Action = SuperMapLib.seAction.scaPan 'Pan map to browse
End Sub
```

The following picture shows the result of Zoom In operation, **Figure 5-4**

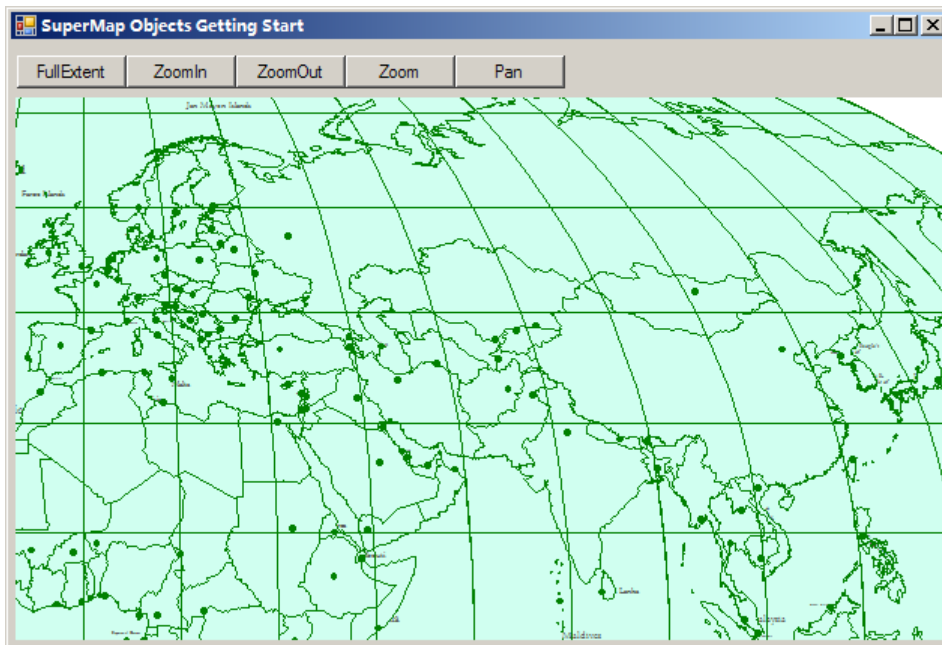


Figure 5-4 The result of map zoom in

5.5 Querying Properties

1. Add a command button to form and then set it's property, **Table 5-2**

Table 5-2 Button property

Name	Caption
cmdSelect	Identify

2. Then, add the following code to each Click event to implement corresponding functions:

```
SuperMap1.Action = SuperMapLib.seAction.scaSelect 'Select
```

3. Finally, add the following code to the GeometrySelected event of SuperMap1:

```
Private Sub AxSuperMap1_GeometrySelected(ByVal sender As System.Object, ByVal e As  
AxSuperMapLib._DSuperMapEvents_GeometrySelectedEvent) Handles  
AxSuperMap1.GeometrySelected
```

```
Dim objRecordset As SuperMapLib.soRecordset 'A recordset object
Dim i As Short 'A cycle variable
Dim strName(12) As String 'For storing property names
Dim strValue(12) As String 'For storing property values
Dim strMessage As String 'Showing all informations

objRecordset = AxSuperMap1.selection.ToRecordset(False) 'Get properties of the selected
object
objRecordset.MoveFirst() 'Move to the first

For i = 1 To objRecordset.FieldCount
    strName(i - 1) = objRecordset.GetFieldInfo(i).Name 'Get the property name
    strValue(i - 1) = objRecordset.GetFieldValue(i) 'Get the property value
Next

strMessage = ""
For i = 0 To objRecordset.FieldCount - 1
    strMessage = strMessage & strName(i) & ": " & strValue(i) & Space(5) & vbCrLf
Next

MsgBox(strMessage)
objRecordset = Nothing 'Release memory
End Sub
```

The following picture shows the result of querying properties from the selected object in the map, *Figure 5-5*

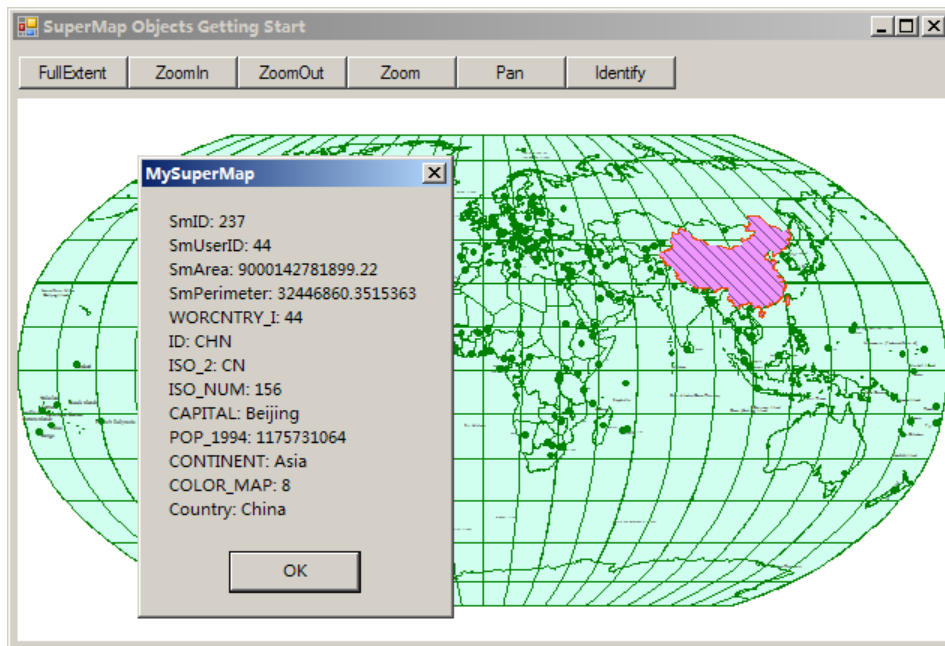


Figure 5-5 The result of identifying map

5.6 Querying map

1. Add a command button and a textbox to form, and then set their properties as follows, **Table 5-3**

Table 5-3 The controls properties

Control	Name	Caption
Button	cmdQueryMap	QueryMap
Textbox	txtExpression	---

2. Add the code to the Click event:

```
Private Sub cmdQueryMap_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Handles cmdQueryMap.Click
    Dim objDtVector As SuperMapLib.soDatasetVector 'A vector dataset variable
    Dim objRecordset As SuperMapLib.soRecordset 'An attribute dataset variable
    Dim objSelection As SuperMapLib.soSelection 'A selection variable
```

```
'Get the vector dataset World_countries on which the query would be performed
If AxSuperWorkspace1.Datasources.Count = 0 Then
    Exit Sub
End If

objDtVector = AxSuperWorkspace1.Datasources.Item("MyDataSource").Datasets("World_countries")
If objDtVector Is Nothing Then
    MsgBox("Failed to open dataset ", MsgBoxStyle.Information)
    Exit Sub
End If

'Query properties from the dataset (The "Query" method can only be applied to the object of
soDatasetVector.)
objRecordset = objDtVector.Query(txtExpression.Text, True)
If objRecordset Is Nothing Then
    Exit Sub
Else
    'Add the data which have been queried to selection
    objSelection = AxSuperMap1.selection
    objSelection.FromRecordset(objRecordset)
    'Refresh the map
    AxSuperMap1.Refresh()
End If

objDtVector = Nothing
objRecordset = Nothing
objSelection = Nothing
End Sub
```

3. Type a SQL expression, e.g. `smid>50`, in query box, and then Click the "QueryMap" button, the results will be displayed as follows, **Figure 5-6**

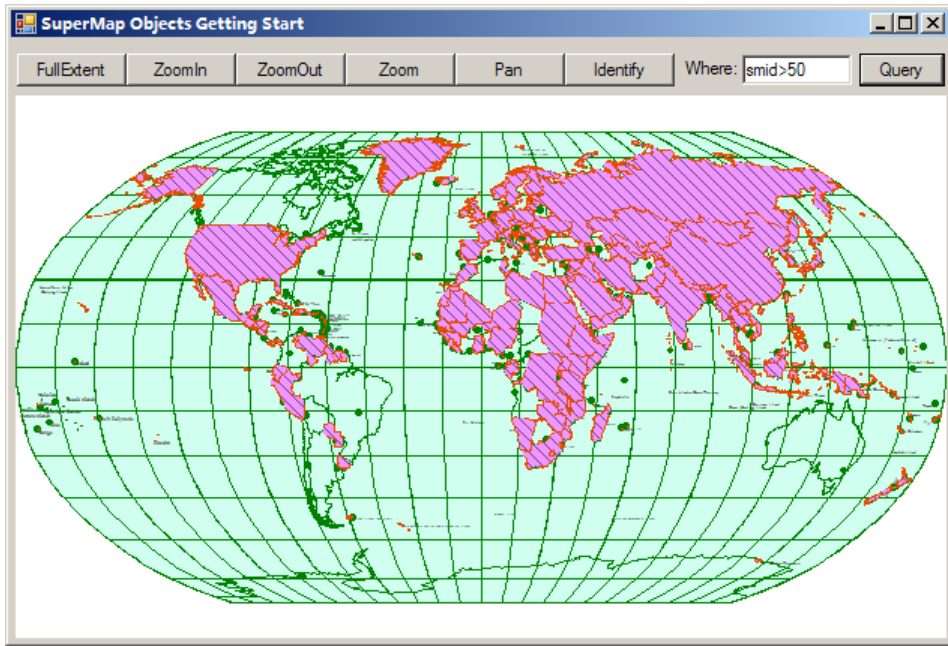


Figure 5-6 *The result of map querying*

Chapter

6

Using SuperMap Objects in C# .NET

6.1 Creating a new project: MySuperMap

1. Create a working directory (C:\Myproject)
2. Download the data compression package World.zip (Include World.sdb and World.sdd) to the working directory (C:\MyProject)
3. Start Visual Studio.NET
4. Create a new windows application named MyFirstSuperMap in working directory (C:\Myproject) , *Figure 6-1*

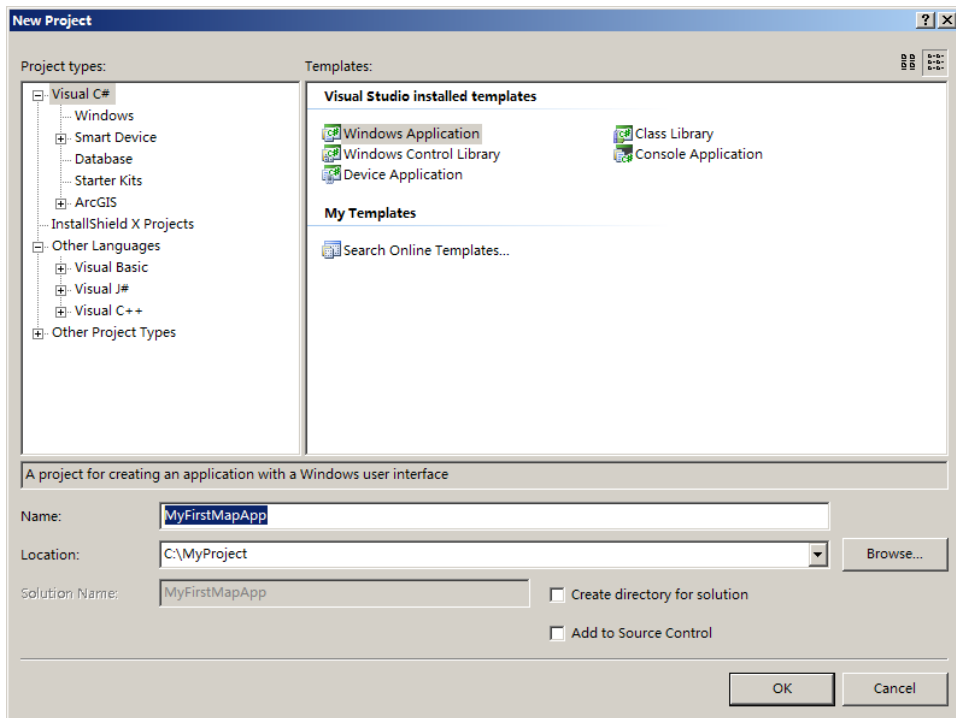


Figure 6-1 Create a new project

6.2 Loading SuperMap Objects controls

1. Load SuperMap Objects control to ToolBox:

- Right-click on the ToolBox and click "Add Tab", and name the new tab as SuperMap
- Right-click on the ToolBox, and click "Choose Items...", the **Customize Toolbox** will be displayed, *Figure 6-2*

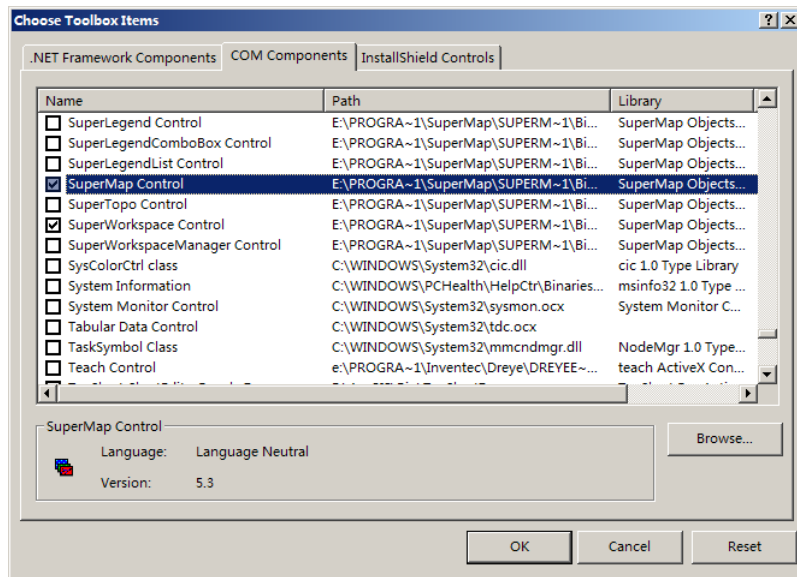


Figure 6-2 Load the SuperMap controls

- In the COM Components tab of the above dialog box, check“SuperMap Control and SuperWorkspace Control”, press OK button. There are two controls in the ToolBox as follow, **Figure 6-3**

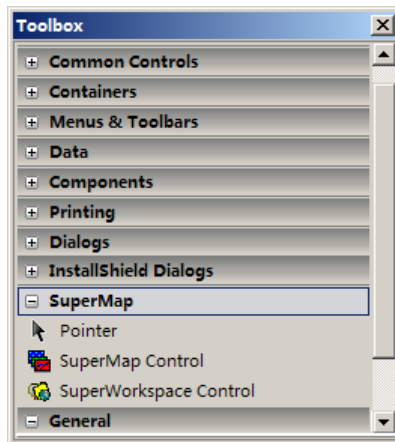


Figure 6-3 *The controls in Toolbox*

6.3 Opening a map and adding layers

1. Rename the window form as FrmMain. Add SuperMap and SuperWorkspace controls to the form and name them as SuperMap1 and SuperWorkspace1 respectively, *Figure 6-4*

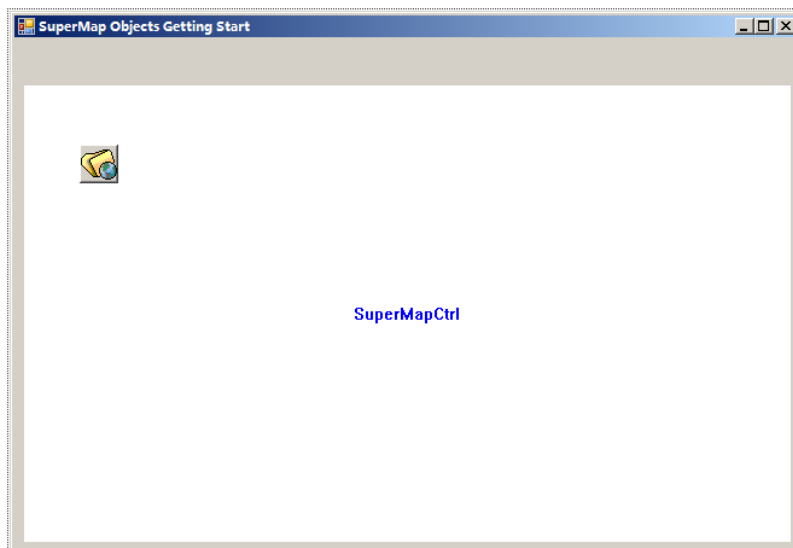


Figure 6-4 Add SuperMap and SuperWorkspace controls to the form

2. Add the following code to FrmMain_Load in FrmMain:

```
private void FrmMain_Load(object sender, System.EventArgs e)
{
    // Connect SuperWorkspace and SuperMap
    SuperMap1.Connect(SuperWorkspace1.CtlHandle);
    // Alias of datasource
    String strAlias;
    // Engine type of data being used in this sample
    SuperMapLib.seEngineType nEngineType;
    // Absolute path of datasource
    String strDataSourceName;
    // Datasource object
    SuperMapLib.soDataSource objDataSource;
    // Whether the datasource to be opened is read only
    bool bReadOnly;
    // Whether a new added layer will be brought to front
    bool bAddToHead;
    // A cycle variable
    int i;
    // Alias can be named arbitrarily, but you'd better use the same one as that of the
    // datasource file
    strAlias = "MyDataSource";
}
```

```

// We will use a SDBPlus data in this sample
nEngineType = SuperMapLib.seEngineType.sceSDBPlus;

strDataSourceName = "C:\\MyProject\\World.sdb";
// Indicates the datasource to be opened can be modified
bReadOnly = false ;
// Open the datasource
objDataSource = SuperWorkspace1.OpenDataSource(strDataSourceName, strAlias,
nEngineType, bReadOnly);
if (objDataSource == null)
{
    MessageBox.Show( "Please download the datasource files(world.sdb,world.sdd)
to content C:\\MyProject\\World.sdb, then run the program, Thanks","Failed to open the
datasource");
    return ;
}
else
{
    // Add all layers of datasource to SuperMap
    for ( i=1;i<=objDataSource.Datasets.Count;i++)
    {
        bAddToHead = true;
        SuperMap1.Layers.AddDataset( objDataSource.Datasets[i], bAddToHead);
    }
    // Refresh the map window
    SuperMap1.Refresh();
    // Modify the styles of the selection object
    SuperMap1.selection.Style.PenColor =
System.Convert.ToUInt32( System.Drawing.ColorTranslator.ToOle(Color.FromArgb(231, 77, 0)));
    SuperMap1.selection.Style.PenWidth = 1;
    SuperMap1.selection.Style.PenStyle = 1;
    SuperMap1.selection.Style.BrushStyle = 5;
    SuperMap1.selection.Style.BrushColor =
System.Convert.ToUInt32( System.Drawing.ColorTranslator.ToOle(Color.FromArgb(115, 69, 140)));
    SuperMap1.selection.Style.BrushBackColor =
System.Convert.ToUInt32( System.Drawing.ColorTranslator.ToOle(Color.FromArgb(239, 150,
255)));
    SuperMap1.selection.Style.BrushOpaqueRate = 50;
    objDataSource = null;
}
}

```

Run the code, and the result will be displayed as follows, *Figure 6-5*

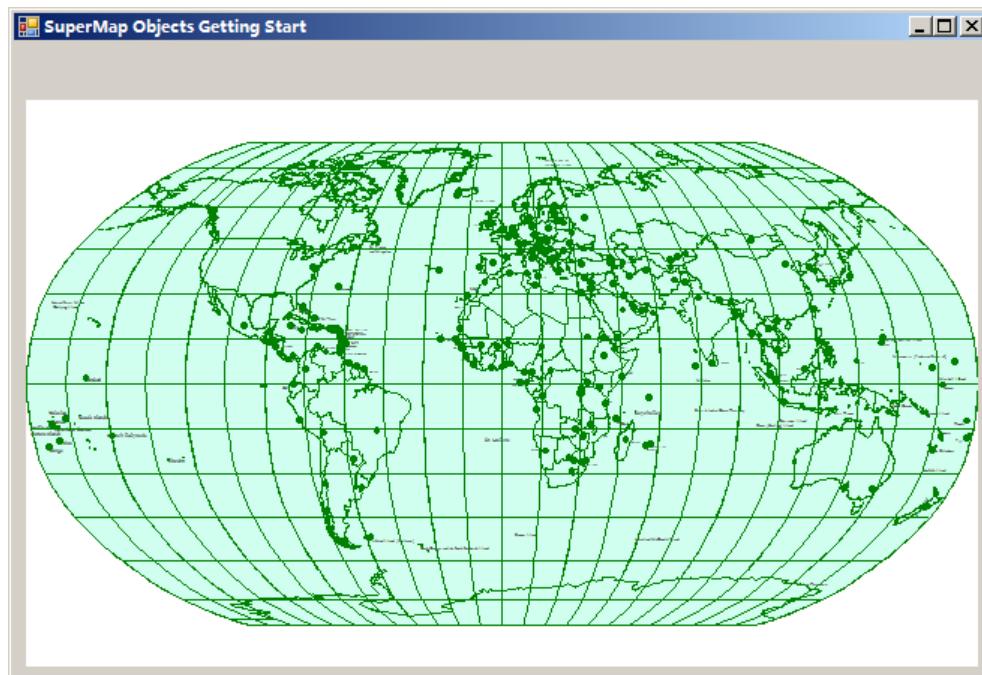


Figure 6-5 *The result interface*

6.4 Browsing map

It is very convenient to perform map operations with SuperMap Objects, such as zooming in, zooming out, zooming freely, panning map, showing the full extent of map, drawing points or lines, etc. Here, we list some map operations as the example in this program.

First, add five command buttons to form1, and then set their properties as follows,
Table 6-1

Table 6-1 *The button properties*

Name	Caption
cmdPan	Pan
cmdZoomIn	ZoomIn

cmdZoomOut	ZoonOut
cmdZoomFree	ZoomFree
cmdViewEntire	ViewEntire

Then, add the code to each Click event to implement corresponding function:

```
private void cmdFullExtent_Click(object sender, System.EventArgs e)
{
    // show full extent of map
    SuperMap1.ViewEntire();
}

private void cmdZoomIn_Click_1(object sender, EventArgs e)
{
    // zoom map in
    SuperMap1.Action = SuperMapLib.seAction.scaZoomIn;
}

private void cmdZoomOut_Click_1(object sender, EventArgs e)
{
    // zoom map out
    SuperMap1.Action = SuperMapLib.seAction.scaZoomOut;
}

private void cmdZoom_Click(object sender, EventArgs e)
{
    // zoom map in or out
    SuperMap1.Action = SuperMapLib.seAction.scaZoomFree;
}

private void cmdPan_Click_1(object sender, EventArgs e)
{
    // pan
    SuperMap1.Action = SuperMapLib.seAction.scaPan;
}
```

The following picture shows the result of map zoom in, **Figure 6-6**

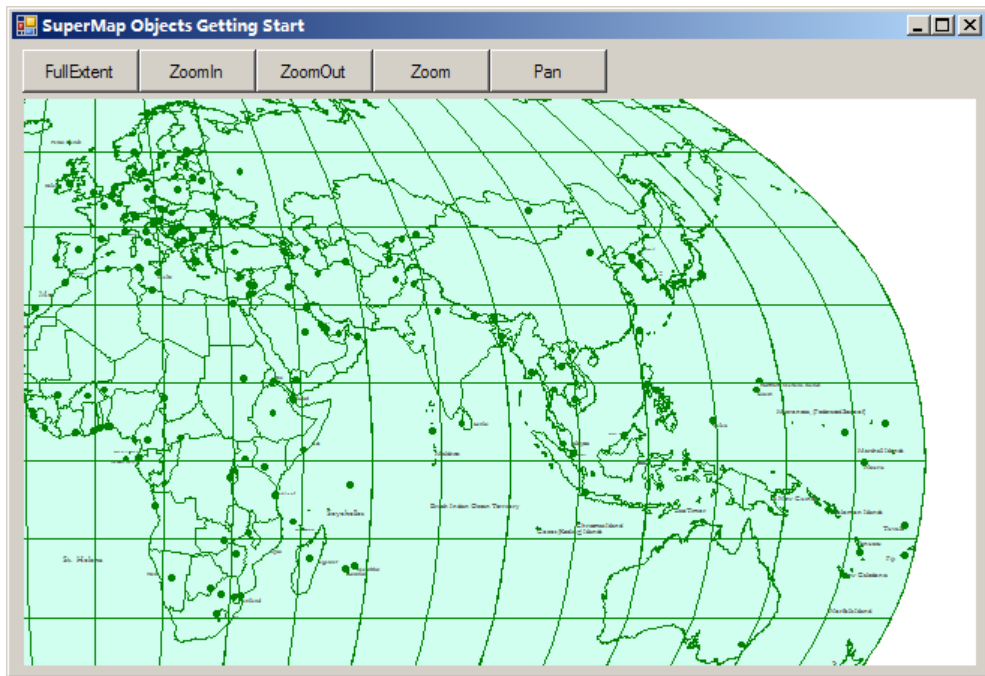


Figure 6-6 *The result of map zoom in*

6.5 Querying Properties

1. Add a command button to form and then set it's property, **Table 6-2**

Table 6-2 *The button property*

Name	Caption
cmdQueryProperties	Identify

2. Add the following code to the responded event of cmdQueryProperties_Click(object sender, System.EventArgs e)

```
private void SuperMap1_GeometrySelected(object sender,
AxSuperMapLib._DSuperMapEvents_GeometrySelectedEvent e)
{
    SuperMapLib.soSelection objSelection;
```

```
SuperMapLib.soRecordset objRd;
objSelection = this.SuperMap1.selection;
// Get the properties of the selected features
objRd = objSelection.ToRecordset(false);
string str = "";
for (int i = 1; i <= objRd.FieldCount; i++)
{
    // Get the property name
    str += objRd.GetFieldInfo(i).Name;
    // Get property value
    str += ":" + objRd.GetFieldValue(i).ToString() + "\n";
}
MessageBox.Show(str, "MyFirstSuperMap ");
objSelection = null;
objRd = null;
}
```

Run the code and click the Identity button, then select an object on the map, finally the properties of the selected object will be displayed as follows, **Figure 6-7**

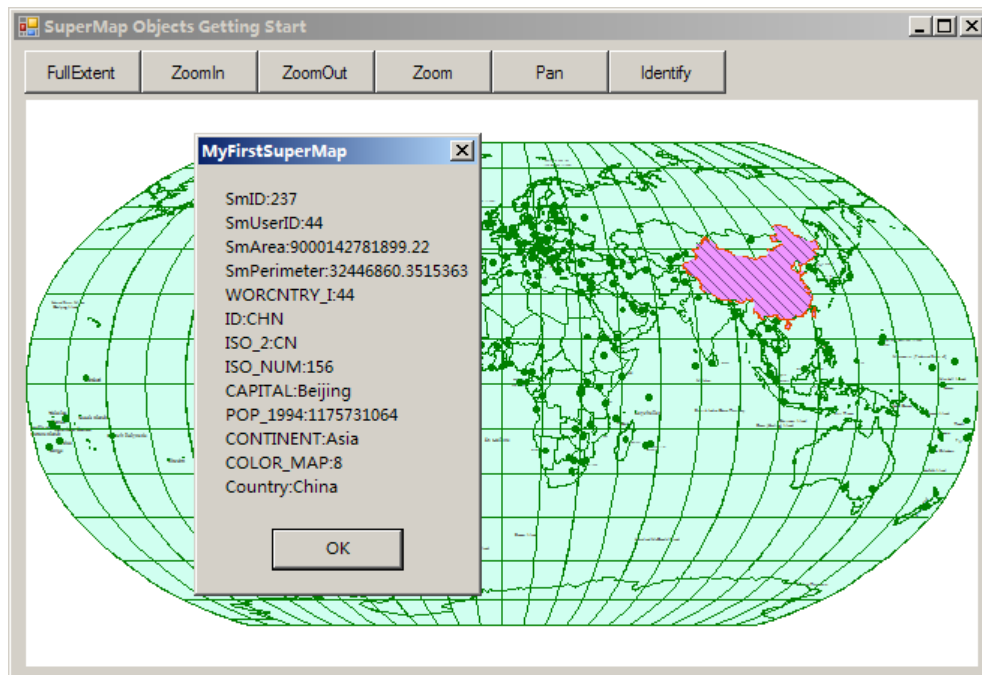


Figure 6-7 The result of identifying map

6.6 Querying map

1. Add a command button and a textbox to form, and then set their properties as follows, **Table 6-3**

Table 6-3 The controls properties

Control	Name	Caption
Button	cmdQueryMaps	QueryMap
Textbox	textBox1	---

2. Add the code to the responded event in cmdQueryMaps_Click(object sender, System.EventArgs e)

```
private void cmdQueryMap_Click(object sender, EventArgs e)
{
```

```
SuperMapLib.soDatasetVector objDtv;           //A vector dataset variable
SuperMapLib.soDataset objDt;
SuperMapLib.soSelection objSelection;         //A selection variable
SuperMapLib.soRecordset objRd;               //A recordset variable

//Get the vector dataset "World_countries" on which the query would be performed
objDt = this.SuperMap1.Layers["World_countries@Mydatasource"].Dataset;
objDtv = (SuperMapLib.soDatasetVector)objDt;

//Query properties from the dataset (The Query method can only be applied to the object
of soDatasetVector.)
objRd = objDtv.Query(this.txtExpression.Text, true, null, "");
//Add the query result to selection
objSelection = this.SuperMap1.selection;
objSelection.FromRecordset(objRd);

// Refresh the map
this.SuperMap1.Refresh();

objRd = null;
objDt = null;
objDtv = null;
objSelection = null;
}
```

3. Type the SQL expression, e.g. `smid>50` in Query text box, then Click the "QueryMap", the results will be displayed as follows, **Figure 6-8**

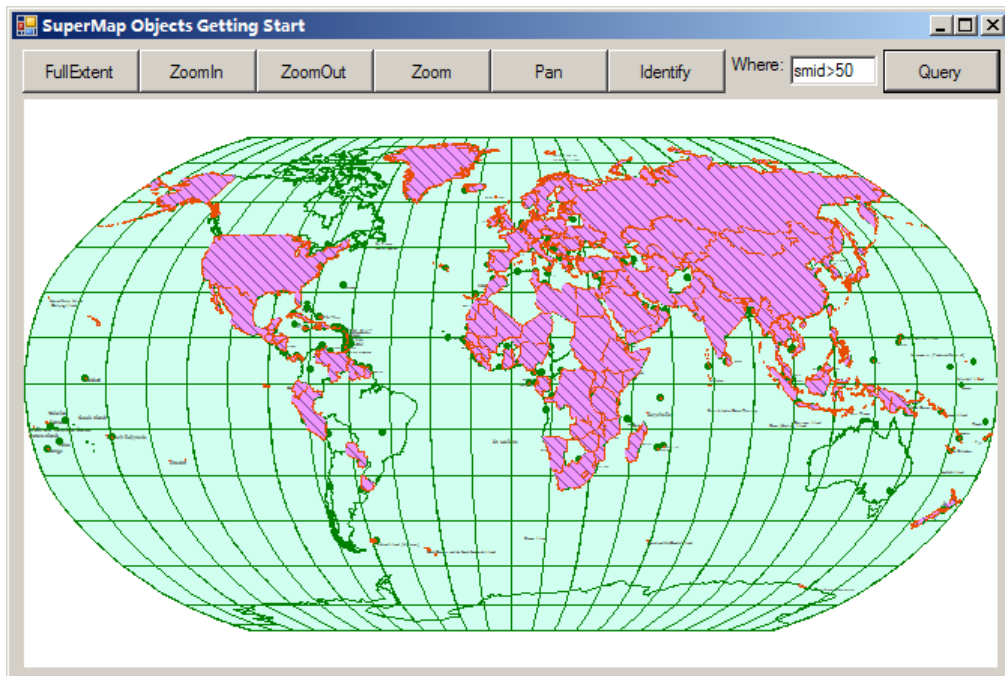


Figure 6-8 *The result of map querying*

Chapter

7

Appendix

7.1 How to register SuperMap Objects?

You will receive a group of serial numbers after you have purchased SuperMap Objects License. You need to register for the SuperWorkspace, SuperAnalyst, Super3D, SuperLayout, SuperTopo and SDX+ moduels through the serial numbers. The remains are not required to be registered. You can call RegisterRuntime method in the related controls to register.

For SuperWorkspace Control, you can use the RegisterForRuntime (Method) and follow the steps below to register when you first open the data, otherwise the About dialog box will pop up when you run the application and the register information will appear on the map window as follows:

```
RegisterForRuntime (User name, Company and Product Serial Number )
```

For example, to register runtime in Form_Load event of a VB6 Project:

```
Private Sub Form_Load()  
    If Not SuperWorkspace1.RegisterForRuntime("", "", "") Then  
        MsgBox "SuperMap Objects failed to register,please type serial number!"  
    End If
```

```
SuperMap1.Connect SuperWorkspace1.Handle
SuperWorkspace1.OpenDataSource App.Path & "..\data\world\world.sdb", "world", sceSDB,
True
SuperMap1.Layers.AddDataset SuperWorkspace1.Datasources(1).Datasets ("World"), True
SuperMap1.ViewEntire
End Sub
```

You also need to call RegisterForRuntime Method to register when using the other controls for the first time.

7.2 How to distribute your application with SuperMap Objects?

1. Distributed Mode

SuperMap Objects is a GIS platform for the developer. You can use SuperMap Objects to develop different, powerful applications in different fields. Usually the installation package is needed to pack the application with its attachment, which is easy to use. Therefore, you may wonder how to distribute your application with SuperMap Objects. We will introduce two modes here, which are the simple and advanced modes.

a. Simple Mode

The simple mode does not attach the SuperMap Objects into the installation package. The user should install SuperMap Objects Runtime and then run the application. To redistribute the application, you only need to supply the others with SuperMap Objects Runtime version, not the development one.

b. Advanced Mode

The advanced mode is produced in your application installation package together with SuperMap Objects. The user needs only to install it once. We recommend you to use this mode if you have integrated rights for your production.

Note: Some files are exclusive to the SuperMap Objects Development version that cannot be used for your user when using the advanced mode to redistribute the application. We list all the redistributed files and unredistributed files below.

2. Distributed files

There are three types of files in SuperMap Objects runtime library: DLL files in the MFC runtime library, DLL files in the SuperMap Objects supported library, and SuperMap Objects OCX controls. Usually we install the first suite of files in Windows operating system folder and the other two files in the same folder. It is highly advised that the user not install these two files in the Windows operating system folder in order to avoid version conflict.

a. DLL files in the MFC Runtime Library:

SuperMap Objects is developed based on MFC of Microsoft. Due to be supplied by MFC runtime library, the user needs to install the files in Windows system folder. For Windows NT4.0 and Windows 2000 users, you need to install the files in System 32 folder. MFC runtime library includes:

msvcrt.dll
msvcp60.dll
mfc42.dll

b. DLL files in SuperMap Objects supported library:

There are more than 30 DLL files in the SuperMap Objects supported library. It is highly recommended that the users don't install the program files under the System directory of the operating system to prevent the possible version conflicts.

DiskSerial.dll
GdiPlus.dll
iconv.dll
hasp_windows_demo.dll
libexpat.dll
lt_appSupport.dll
lt_common.dll
lt_messageText.dll
lt_meta.dll
lt_trans.dll
lt_utils.dll
lt_xtrans.dll

mrsid32.dll
mrsidd.dll
NCScnet.dll
NCSEcw.dll
NCSEcwC.dll
NCSUtil.dll
sx32w.dll
SmAlib50.dll
SmCSF50.dll
SmCtl50.dll
SmDTM50.DLL
SmeDgn50.dll
SmEdit50.dll
SmeImg50.dll
SmeMemory.dll
SmElem50.dll
SmEng50.dll
SmeSDB50.dll
SmeSDBPlus50.dll
SmFPS50.dll
SmFSL50.DLL
SmGrid50.dll
SmImg50.dll
SmLogRes.dll
SmLsl50.dll
Smlyt50.dll
SmOCI.dll
SmOdbc50.dll
SmPmp50.dll
SmPrj50.dll
SmScn50.dll
SmSym50.dll

SmTopo50.dll

SmWks50.dll

c. OCX files

There are several OCX files in SuperMap Objects. We also recommend you not to install them in the Window operating system folder. Once the user has completely installed these files, the user needs to register them either by running the OCX register functions in the installation tools such as InstallShield, or manually select the RegSvr32.exe file in Windows. SuperMap Objects installation package also supplies a special installation file (RegisterSuperMap.exe), which can automatically register all SuperMap Objects controls from the same directory. The OCX files and other matter for attention will be explained in details further on.

SuperMap.ocx

SuperLayout.ocx

SuperLegend.ocx

SuperAnalyst.ocx

Super3D.ocx

SuperTopo.ocx

SuperGridView.ocx

SuperWkspManager.ocx

SmxLockInfo.ocx

The user needs to add the following associated data engines if you are using the data engine in SuperMap Objects.

SmeAcad50.sdx

SmeDB2Plus50.sdx

SmeKingBase50.sdx

SmeODm50.sdx

SmeOrcPlus50.sdx

SmeOSbs50.sdx

SmeOSP50.sdx

SmeOSqlPlus.sdx

SmeSQL50.sdx

SmAcad.spx

SmDgn.spx

[d. Image Plugins files](#)

SmBmpImage.spi

SmJpegImage.spi

SmMapCache.spi

SmRawImage.spi

SmTifImage.spi

SmWrapImg.spi

3. Undistributed files

Some files are exclusive to SuperMap Objects developed version, which cannot be supplied to another user when sharing the application. Your user can contact our local agent if they want to use SuperMap Objects for development. The exclusive files include, but not limit to:

[a. Type library files \(*.tlb used in developing VC application\)](#)

SuperLayout.tlb

Super3D.tlb

SuperLegend.tlb

SuperAnalyst.tlb

SuperMap.tlb

SuperTopo.tlb

SuperGridView.tlb

SuperWkspManager.tlb

and other files with .tlb as extension in the installation package in SuperMap Objects.

[b. Help documents](#)

SmProRef.chm

GettingStarted.chm

SmoTech.chm

Samples.chm

4. Register SuperMap Objects

First, the user needs to pack the RegisterSuperMap.exe file supplied by SuperMap Objects into InstallShield, which is in the same directory with the SuperMap Objects OCX files. The user should set the self-registered property to FALSE in OCX file

groups to solve the registration problem himself. The user also needs to package the MFC runtime library together and set Potentially Locked properties to TRUE; otherwise the system will not work properly.

Second, define and implement the following function in InstallShield (6.22 higher) Script:

(1) Define ExecuteAfterRebootOrAtEnd() function, declare in the head file (distinguish the capital letters)

```
function ExecuteAfterRebootOrAtEnd()
STRING strTempFile;
begin
strTempFile = TARGETDIR ^ "\\Bin\\SmRegister.exe"; //name the target path for automatic register
and modify it based on practical installation directory
LongPathToQuote ( strTempFile, TRUE );
LaunchAppAndWait ( strTempFile , "" , WAIT );
end;
```

(2) Response OnEnd(), add these code:

```
function OnEnd()
begin
if (BATCH_INSTALL) then
//do nothing
else
ExecuteAfterRebootOrAtEnd();
endif;
end;
```

(3) Response OnRebooted() function , add these code:

```
function OnRebooted()
begin
ExecuteAfterRebootOrAtEnd();
end;
```

The End