

# Creating your first Virtual Earth Web Page Part 2

By Dr. Neil Roodyn

In the first part of this article you learned how to add and use the version 3 Virtual Earth map control in your own web page. In this article you will discover how to use some of the other controls that make up the Virtual Earth product.

By the end of this article you will have extended the page we build in part 1 to use new images for the compass and zoom bar controls and include a custom panel. The end result should appear as seen in Figure 1 and can be found here

<http://www.viavirtualearth.com/MyVirtualEarth/v3/gettingstartedpt2.htm>

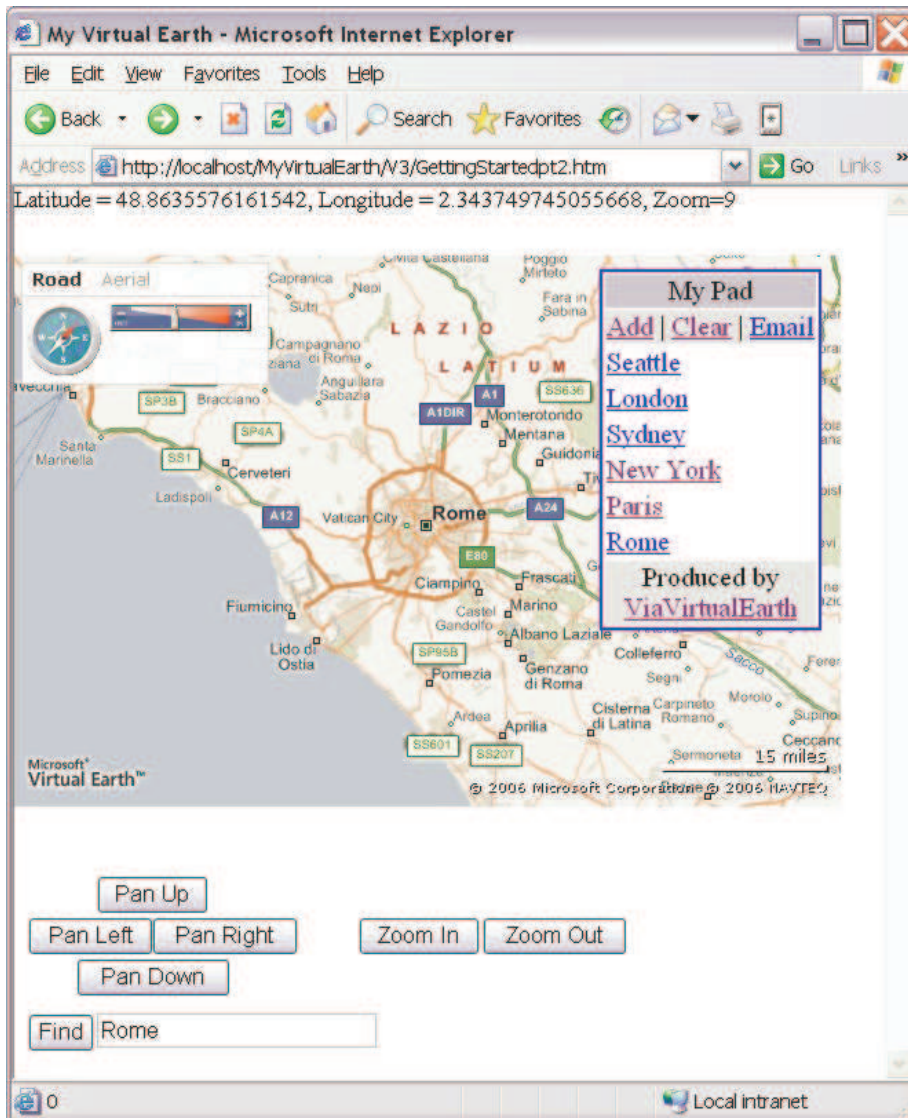


Figure 1

## Setting up the script for the other controls

In part 1 you discovered that the script for the Virtual Earth Map Control can be found at <http://dev.virtualearth.net/mapcontrol/v3/mapcontrol.js>. In the first version of the Virtual Earth control you would have had to include another VE.JS script file to use the basic controls, such as the compass or the zoom bar. These controls are now included by default in the new dashboard on the VirtualEarth Map Control. Version 3 has also negated the need to include the reference to the CSS file in your web page. The CSS is however still there and in this article you will explore how to change the style on the map.

### The Compass control

The first control you are going to modify is the compass control. This provides a great way for navigating around the map. The compass is represented by an image, it is best to use a gif with transparency so that the compass does not cover any more of the map than is needed. You can create your own image or copy the image used with this example. The compass is defined in the Mapcontrol.css cascading style sheet as

```
.Compass
{
    width:54px;
    height:54px;
    background:url(../i/compass.gif);
    margin:0;
    cursor:pointer
}
```

In order to use a different image for the compass you will need to add your own definition for the .Compass style.

```
<link href="http://local.live.com/css/MapControl.css"
      type="text/css" rel="stylesheet" />
<script src="http://local.live.com/MapControl.ashx">
</script>
<STYLE TYPE="text/css" MEDIA=screen>
<!--
.Compass{ width:54px; height:54px; background:url(i/compass.gif);
margin:0px; cursor:pointer }
-->
</STYLE>
```

This is the image we use for this exercise



The page should now display your compass image in the dashboard on the map.


## ***The Zoom control***

The zoom control provides a slick user interface for zooming in and out of the map. It also provides visual feedback as to the amount a map can be zoomed and where the current map is on that scale.

In order to modify the zoom control to the page you must add some style classes that describe how the control will look.

The control consists of 2 graphical components, the bar and the slider. For this exercise we are using these images

Bar: 

Slider: 

In the same way you defined a different image for the compass you will need to add your own definition for the `.ZoomBar` and `.ZoomBar_slider` styles.

```
<link href="http://local.live.com/css/MapControl.css"
      type="text/css" rel="stylesheet" />
<script src="http://local.live.com/MapControl.ashx">
</script>
<STYLE TYPE="text/css" MEDIA=screen>
<!--
.Compass{ width:54px; height:54px; background:url(i/compass.gif);
margin:0px; cursor:pointer }
.ZoomBar{ position:relative;background:url(i/zoom/bar.gif);
width:103px; height:20px; margin:2px; overflow:hidden; }
.ZoomBar_slider { position:absolute; background:url(i/zoom/slider.gif);
width:7px; height:20px; overflow:hidden; display:block; }
-->
</STYLE>
```

At this stage you have a web page with a map and great looking controls for interacting with the map. The page should look something like that shown in figure 2.

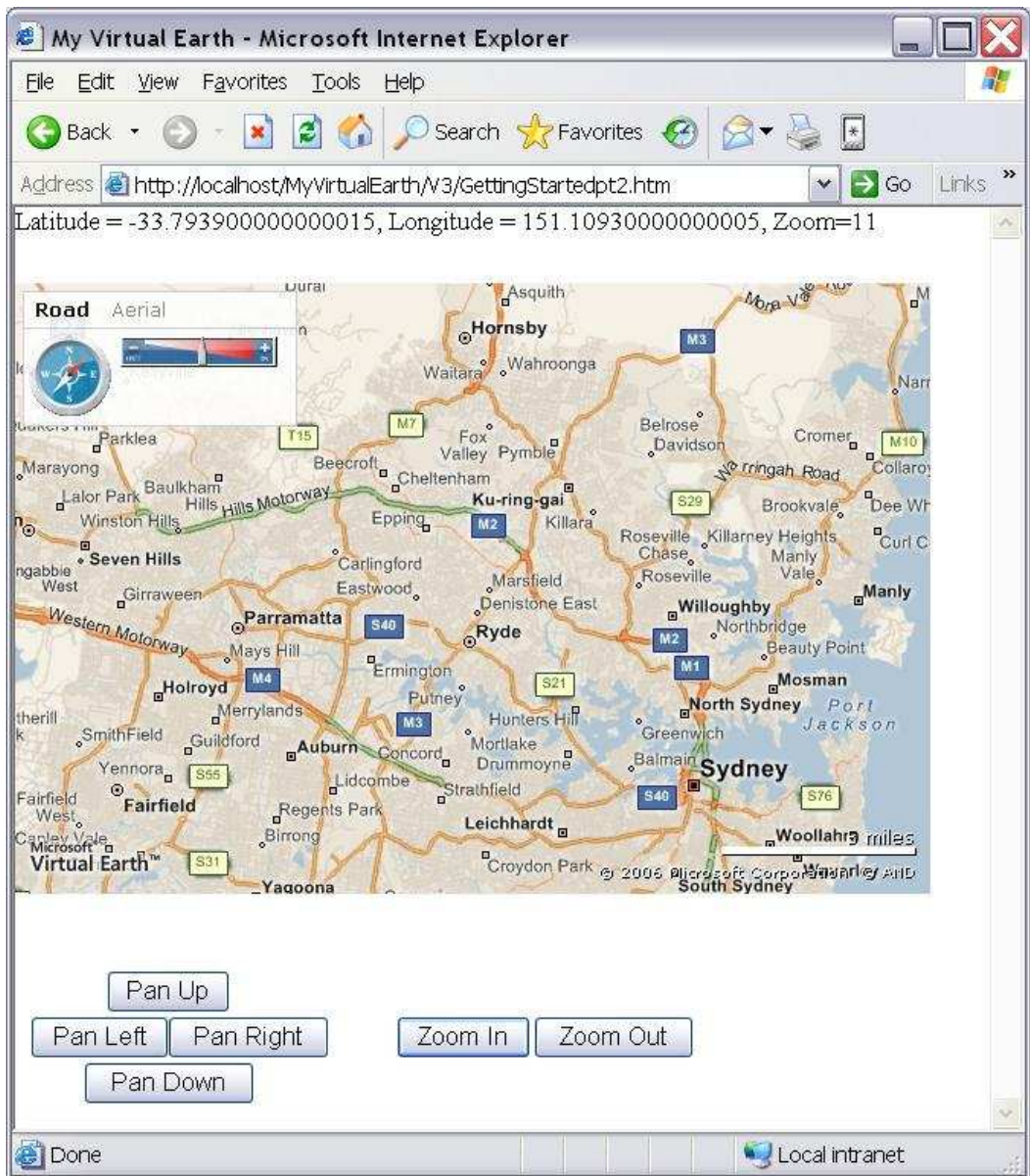


Figure 2

The next control you are going to explore is the most complicated and the one that has the most compelling features; the panel. You will discover how to create a panel that allows you to display information in a 'window' that displays on the map.

### **Creating a Panel**

The ability to create a panel and add information to the map is something that has been unsupported and hard to achieve in previous versions of Virtual Earth. With the release of beta version 3 the VEMap control supports a method called AddControl. AddControl allows you to define an HTML control and place it on the map.

As an example try to create a Panel element and add to the map in the your OnPageLoad method.

```
var panel = document.createElement("div");
panel.style.top = "10px";
panel.style.left = "450px";
panel.style.width = "100px";
panel.style.border = "2px solid blue";
panel.style.background = "White";
panel.innerHTML = "<table border=0 width='100%' >" +
"<tr> <td bgcolor='#C0C0C0'> <p align='center'>My Pad</td></tr>" +
"<tr><td>Some information you want to display</td></tr>" +
"<tr><td>&nbsp;</td></tr>" +
"<tr><td>&nbsp;</td></tr>" +
"<tr><td>&nbsp;</td></tr>" +
"<tr><td>Produced by " +
"<a href=http://viavirtualearth.com/>ViaVirtualEarth</a>" +
"</td></tr>" +
"</table>";
map.AddControl(panel);
```

If you view the page in your browser you will now see a Panel, figure 3.

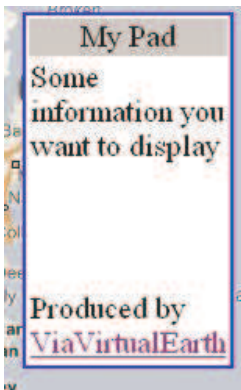


Figure 3

As the control is standard HTML you can add links, customise the appearance and make it behave as you see fit.

A good approach to make the management of the panel contents easier is to define the fixed areas of the panel as variables that can be reused.

```
var panelHead = "<table border=0 width='100%' >" +
"<tr> <td bgcolor='#C0C0CF'> <p align='center'>My Pad</td></tr>" +
"<tr><td bgcolor='#E0E0E0'> <p align='center'>" +
"<a href='\";javascript:AddPoint();\" >Add</a>&nbsp;&nbsp;&nbsp;";
```

```

"<a href=\"javascript:ClearPanel();\" >Clear</a>&nbsp;&nbsp;&nbsp;|&nbsp;&nbsp;&nbsp;\" +
"<a href=\"javascript:EmailPanel();\" >Email</a>\" +
\"</td></tr>\";
var panelEmptyContent = \"<tr><td>Empty panel</td></tr>\" +
\"<tr><td>&nbsp;&nbsp;&nbsp;</td></tr>\" +
\"<tr><td>&nbsp;&nbsp;&nbsp;</td></tr>\";
var panelFoot = \"<tr><td bgcolor=#E0E0E0><p align='center'>Produced by \" +
\"<a href=http://viavirtualearth.com/>ViaVirtualEarth</a>\" +
\"</td></tr>\" +
\"</table>\";

```

Then when you create the panel you can set the HTML content using these variables:

```

panel.innerHTML = panelHead +
panelEmptyContent +
panelFoot;

```

In this example you have Clear and Email and also an Add menu item. You now need to write the code that implements the functions called by the menu. Start with the AddPoint function. This will add an entry to the Panel with a name and a location (LatLong). The name in this example will be “custom”, you will extend this later on in this article.

Define a type called a PanelPoint that contains a name and a LatLong variable. The panel will contain a collection of these points so also define an array.

```

PanelPoint = function(n, pt)
{
    this.Name = n;
    this.LatLong = pt;
}
//array of PanelPoint objects
var panelPoints = new Array();

```

The AddPoint function can then be written to use this array. This function takes the center point of the map and adds that location to the panel array with the name supplied.

```

function AddPoint(name)
{
    var mapCenter = map.GetCenter();
    var panelContent = \"\";
    var newPoint = new PanelPoint(name, mapCenter);

    panelPoints.push(newPoint);
    var pts = panelPoints;
    for(var i=0;i<pts.length;i++)
    {
        var pt = panelPoints[i];
        panelContent += \"<tr><td>\";
        panelContent +=
            \"<a href='javascript:map.PanToLatLong(new VELatLong(\";

```

```

        panelContent += pt.LatLong.Latitude + "," + pt.LatLong.Longitude;
        panelContent += "));'>" + pt.Name;
        panelContent += "</a>";
        panelContent += "</td></tr>";
    }

    panel.innerHTML = panelHead +
        panelContent +
        panelFoot;
}

```

You should now be able to add points to the Panel, move around the map and add points. Clicking on the points in the panel should take you back to those points.

Now you can write the EmailPanel function to send an email with the points in the Panel. This function uses the Live Local site to display any points you send in the email.

```

function EmailPanel()
{
    var body = "http://ViaVirtualEarth.com/ \n\n";
    var pts = panelPoints;
    for(var i=0;i<pts.length;i++)
    {
        var pt = panelPoints[i];
        body += pt.Name + "\n";
        body += "http://local.live.com/default.aspx?cp=";
        body += pt.LatLong.Latitude + "~" + pt.LatLong.Longitude;
        body += "&lvl=12";
        body += "\n\n";
    }
    var url = 'mailto:?subject=My%20Virtual%20Earth%20Panel&body='
        + escape(body);
    window.open(url);
}

```

Finally you can write the function to clear the items from the panel.

```

function ClearPanel()
{
    while (panelPoints.length)
    {
        panelPoints.pop();
    }

    panel.innerHTML = panelHead +
        panelEmptyContent +
        panelFoot;
}

```

It would be good to add points to the Panel based on a search, that is what you will learn in the next section.

## Finding

In previous versions of Virtual Earth there have been a number of issues with performing a find operation on the map. Microsoft has done a fantastic job of resolving these and adding find functionality to your map is now simple.

At the end of the body section of the HTML page add an input and button control to allow the user to enter some text to search for and click the button to start the search.

```
<input type="button" value="Find" onclick="DoFind()"
ID="FindButton" NAME="FindButton"
style="position:absolute;left:10px;top:600"/>
<input type="text" name="WhereText" size="20"
ID="WhereText"
style="position:absolute;left:60px;top:600"/>
```

To the script section of our page add a DoFind function.

```
function DoFind()
{
    var where = document.getElementById("WhereText").value;
    map.FindLocation(where);
}
```

This page will perform a search and center the map at location found. It would now be great to add the found location to the Panel.

In order to achieve this you will need to set a variable to indicate you have requested a find operation. Then when the map is changed you can add the new center point to the panel. This is somewhat of a hack. At the moment there is no supported method to get the results from a FindLocation request.

If you look in the documentation you will find that if you are searching for something (a “what” search) at a location, using the VEMap.Find method, you can specify a callback. This callback doesn’t get called if the ‘what’ field is empty, i.e. you are only doing a location search.

```
var where;
var finding = false;
function DoFind()
{
    where = document.getElementById("WhereText").value;
    map.FindLocation(where);
    finding = true;
}

function FindCompleted()
{
    finding = false;
```

```
AddPoint(where);  
}
```

Now in your OnPageLoad method you can hook the onchangeview event to a method called ViewChange.

```
map.AttachEvent("onchangeview",  
    ViewChange);
```

Then in the ViewChange method call FindCompleted if you requested a find operation.

```
function ViewChange(e)  
{  
    if (finding)  
    {  
        FindCompleted();  
    }  
}
```

## **Conclusion**

You have now created a web page that allows a user to search for a location and add that location to a panel.

The full listing for the page is shown in the listing below.

```
<html>  
  <head>  
    <title>My Virtual Earth</title>  
  
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">  
    <script  
src="http://dev.virtualearth.net/mapcontrol/v3/mapcontrol.js"></script>  
    <STYLE TYPE="text/css" MEDIA=screen>  
    <!--  
    .Compass{ width:54px; height:54px; background:url(i/compass.gif);  
margin:0px; cursor:pointer }  
    .ZoomBar{ position:relative;background:url(i/zoom/bar.gif);  
width:103px; height:20px; margin:2px; overflow:hidden; }  
    .ZoomBar_slider { position:absolute;  
background:url(i/zoom/slider.gif);  
width:7px; height:20px; overflow:hidden; display:block; }  
    -->  
    </STYLE>  
  
    <script>  
  
    var map = null;  
    var panel = null;
```

```

var panelHead = "<table border=0 width='100%' >" +
  "<tr> <td bgcolor='#C0C0CF'> <p align='center'>My  

Pad</td></tr>" +
  "<tr><td bgcolor='#E0E0E0'> <p align='center'>" +
  "<a href=\"javascript:AddPoint('Custom');\"  
>Add</a>&nbsp;  |&nbsp;  " +
  "<a href=\"javascript:ClearPanel();\"  
>Clear</a>&nbsp;  |&nbsp;  " +
  "<a href=\"javascript:EmailPanel();\" >Email</a>" +
  "</td></tr>";
var panelEmptyContent = "<tr><td>Empty panel</td></tr>" +
  "<tr><td>&nbsp;  </td></tr>" +
  "<tr><td>&nbsp;  </td></tr>";
var panelFoot = "<tr><td bgcolor='#E0E0E0'><p  
align='center'>Produced by " +
  "<a  
href=http://viavirtualearth.com/>ViaVirtualEarth</a>" +
  "</td></tr>" +
  "</table>";

PanelPoint = function(n, pt)
{
  this.Name = n;
  this.LatLong = pt;
}

var panelPoints = new Array();

function UpdateInfo(e)
{
  document.getElementById("info").innerHTML =
    'Latitude = ' +
    e.view.latlong.latitude +
    ', Longitude = '
    + e.view.latlong.longitude +
    ', Zoom=' +
    e.view.zoomLevel;

  }

  function ViewChange(e)
  {
    if (finding)
    {
      FindCompleted();
    }
  }

var pushpinOnMap = false;
function MouseClick(e)
{
  if (pushpinOnMap)
  {
    map.DeletePushpin(1);
  }
}

```

```

    }

    var pin = new VEPushpin(1,
        new VELatLong(e.view.LatLong.Latitude,
e.view.LatLong.Longitude),
        'pin.jpg',
        'My Pin',
        'Text that describes my pin'
    );

    map.AddPushpin(pin);
    pushpinOnMap = true;
}

function AddPoint(name)
{
    var mapCenter = map.GetCenter();
    var panelContent = "";
    var newPoint = new PanelPoint(name, mapCenter);

    panelPoints.push(newPoint);
    var pts = panelPoints;
    for(var i=0;i<pts.length;i++)
    {
        var pt = panelPoints[i];
        panelContent += "<tr><td>";
        panelContent += "<a
href='javascript:map.PanToLatLong(new VELatLong(";
        panelContent += pt.LatLong.Latitude + "," +
pt.LatLong.Longitude;
        panelContent += ")');>" + pt.Name;
        panelContent += "</a>";
        panelContent += "</td></tr>";
    }

    panel.innerHTML = panelHead +
        panelContent +
        panelFoot;
}

function ClearPanel()
{
    while (panelPoints.length)
    {
        panelPoints.pop();
    }

    panel.innerHTML = panelHead +
        panelEmptyContent +
        panelFoot;
}

function EmailPanel()
{
    var body = "http://ViaVirtualEarth.com/ \n\n";
    var pts = panelPoints;
    for(var i=0;i<pts.length;i++)

```

```

    {
        var pt = panelPoints[i];
        body += pt.Name + "\n";
        body += "http://local.live.com/default.aspx?cp=";
        body += pt.LatLong.Latitude + "~" + pt.LatLong.Longitude;
        body += "&lvl=12";
        body += "\n\n";
    }
    var url =
        'mailto:?subject=My%20Virtual%20Earth%20Panel&body='
        + escape (body);
    window.open (url);

}

function OnPageLoad()
{
    map = new VEMap ('myMap');
    map.LoadMap (new VELatLong (-33.7939, 151.1093),
        10, 'r', false);

    map.AttachEvent ("onendcontinuouspan",
        UpdateInfo);

    map.AttachEvent ("onendzoom",
        UpdateInfo);

    map.AttachEvent ("onclick",
        MouseClick);

    map.AttachEvent ("onchangeview",
        ViewChange);

    panel = document.createElement ("div");
    panel.style.top = "10px";
    panel.style.left = "425px";
    panel.style.width = "150px";
    panel.style.border = "2px solid blue";
    panel.style.background = "White";
    panel.innerHTML = panelHead +
        panelEmptyContent +
        panelFoot;

    map.AddControl (panel);
}

function DoPanUp ()
{
    map.vemapcontrol.ContinuousPan (0, -10, 20, false);
}
function DoPanDown ()
{
    map.vemapcontrol.ContinuousPan (0, 10, 20, false);
}
function DoPanLeft ()
{
    map.vemapcontrol.ContinuousPan (-10, 0, 20, false);
}

```

```

    }
    function DoPanRight()
    {
        map.vemapcontrol.ContinuousPan(10, 0, 20, false);
    }

    function DoZoomIn()
    {
        map.ZoomIn();
    }

    function DoZoomOut()
    {
        map.ZoomOut();
    }

    var where;
    var finding = false;
    function DoFind()
    {
        where = document.getElementById("WhereText").value;
        map.FindLocation(where);
        finding = true;
    }

    function FindCompleted()
    {
        finding = false;
        AddPoint(where);
    }

</script>
</head>
<body onload="OnPageLoad();" >
    <div id="info"
        style="position:relative;HEIGHT: 50px; font-size:10pt">
    </div>
    <div id='myMap'
        style="position:relative;width:600px; height:400px;">
    </div>
    <input type="button" value="Pan Up"
        onclick="DoPanUp()"
        ID="PanUpButton" NAME="PanUpButton"
        style="position:absolute;left:60px;top:500"/>

    <input type="button" value="Pan Left"
        onclick="DoPanLeft()"
        ID="PanLeftButton" NAME="PanLeftButton"
        style="position:absolute;left:10px;top:530"/>

    <input type="button" value="Pan Right"
        onclick="DoPanRight()"
        ID="PanRightButton" NAME="PanRightButton"
        style="position:absolute;left:100px;top:530"/>

    <input type="button" value="Pan Down"
        onclick="DoPanDown()"

```

```
ID="PanDownButton" NAME="PanDownButton"
style="position:absolute;left:45px;top:560"/>

<input type="button" value="Zoom In" onclick="DoZoomIn()"
ID="ZoomInButton" NAME="ZoomInButton"
style="position:absolute;left:250px;top:530"/>

<input type="button" value="Zoom Out" onclick="DoZoomOut()"
ID="ZoomOutButton" NAME="ZoomOutButton"
style="position:absolute;left:340px;top:530"/>

<input type="button" value="Find" onclick="DoFind()"
ID="FindButton" NAME="FindButton"
style="position:absolute;left:10px;top:600"/>
<input type="text" name="WhereText" size="20"
ID="WhereText" value="Seattle"
style="position:absolute;left:60px;top:600"/>

</body>
</html>
```