Step by Step Instructions to Building Your First Page

OVERVIEW

This tutorial provides step-by-step instructions to show you how to build your own web page starting from a basic template page that we have supplied called ***basic_template.htm***. In order to follow this tutorial you will need to use a Native HTML editor that lets you modify the code of the page. This could be as simple a Notepad, but we would recommend an editor designed for HTML scripting. We can recommend an editor called Homesite 5 from Macromedia as an excellent tool that is easy to learn and use. You can download a 30-day free trial from **www.macromedia.com** if you wish. Do not use WYSIWYG editors like Frontpage because they have a tendency to insert extraneous code that can effect the operation of your website. The completed page is called ***multiple_basic.htm.***

To get started load the ***basic_template.htm*** page into your editor. The page is located in the ***<your server>/IWSDoc/Tutorial*** folder.

Let's have a quick look at the template page.

Your basic template page will look like this:

```
<html>
<head>
<title>My First Image Web Server Web Page</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<LINK REL="stylesheet" HREF="css/SampleStyle.css" TYPE="text/css">

<SCRIPT LANGUAGE=javascript src="scripts/Util.js"></SCRIPT>
<SCRIPT LANGUAGE=javascript src="/ecwplugins/lib/scripts/NCSConstants.js"></SCRIPT>
<SCRIPT LANGUAGE=vbscript   src="/ecwplugins/lib/scripts/NCSCheckActiveX.vbs"></SCRIPT>
<SCRIPT LANGUAGE=javascript src="/ecwplugins/lib/scripts/NCSCheck.js"></SCRIPT>

<SCRIPT LANGUAGE=javascript src="/ecwplugins/lib/scripts/NCSCreateView.js"></SCRIPT>
<SCRIPT LANGUAGE=javascript src="/ecwplugins/lib/scripts/NCSControl.js"></SCRIPT>
<SCRIPT LANGUAGE=javascript src="/ecwplugins/lib/scripts/NCSToolbar.js"></SCRIPT>
<SCRIPT LANGUAGE=javascript src="/ecwplugins/lib/scripts/NCSProgressbar.js"></SCRIPT>

</head>

<SCRIPT LANGUAGE=javascript>
<!--

// You will add your Javascript functions here.

//-->
</SCRIPT>
</head>

<body>
<table width ="100%" align = "center" border = 1 bordercolor = gray>
   <tr>
        <th>
               <font color=gray face=verdana size=2>Enter Your Page Heading Here</font>
        </th>
   </tr>
   <tr>
        <td>
            <font face = arial size = 2 color = black>
                Describe your page here.
             </font>
        </td>
   </tr>
</table>

</body>
</html>
```

Explanation of some of the Included Code

```
<LINK REL="stylesheet" HREF="css/SampleStyle.css" TYPE="text/css">
```

There is a style sheet included in the tutorial folder that is added into your code with the above line. If you are familiar with using Styles you can make use of the style elements defined therein and add your own if you wish.

```
<SCRIPT LANGUAGE=javascript src="scripts/Util.js"></SCRIPT>
```

The above line adds in a JavaScript Include file to your page which is stored in the Tutorial /tutorial/scripts folder. This file has utilities which will be referenced in your web pages.

```
<SCRIPT LANGUAGE=javascript src="/ecwplugins/lib/scripts/NCSConstants.js"></SCRIPT>
```

The above line of code is a JavaScript Include file that is added to your web page and defines a set of constants that are used in the classes that ERM supplies for simplifying your coding.

```
<SCRIPT LANGUAGE=vbscript   src="/ecwplugins/lib/scripts/NCSCheckActiveX.vbs"></SCRIPT>
<SCRIPT LANGUAGE=javascript src="/ecwplugins/lib/scripts/NCSCheck.js"></SCRIPT>
```

The above 2 lines of code are JavaScript Include files which are added to your web page and are used to check that the browser version is supported and that the plug-in version is up to the version level defined in the IMAGE_WEB_SERVER_VERSION constant stored in the ***NCSConstants.js*** include file.

You should always start your page with these included JavaScript files and then use the supplied Classes for adding in the View, Toolbar and ProgressBar controls as it makes your coding much easier and more robust for cross-browser applications.

Save the Page to a new file name first

Before we modify the basic template page you should save it to a different file name so you do not overwrite the template. Use File > SaveAs to save the template code to a new file called ***myfirstpage.htm***.

Viewing your page in a browser

As you modify the code you will want to check the operation of your page to test what you have done. In order to operate this page you need to run it through your http:// web server since the include files are referenced in relation to the root of this server to pick up the ***/ecwplugins*** folder of the Image Web Server installation. Therefore set up your browser to run the following URL:

http://<yourservername>/IWSDoc/tutorial/myfirstpage.htm

Substitute your server name where shown and run the page in your browser (Internet Explorer 5.5+ or Netscape 7+). You should bookmark the page so you can easily get back to it between sessions.

Let's Make Some Simple Changes

To get a feel for modifying your page, saving it and viewing it in your browser, let's make a couple of simple changes to the page that is just straight HTML. You will notice that we have included a basic table structure in the template page. The first row of the table has a table heading cell defined. Modify the heading in this cell to be as follows:

```
<th>
     <font color=gray face=verdana size=2>Multiple ECW Layers Example</font>
</th>
```

Now save the file and click the refresh button on your browser to confirm that the change has taken effect.

Now let's change the brief description at the top of the page to reflect what we are planning to build in this exercise. Modify the description text to the following, save the file and confirm the changes in the browser.

```
<td>
      <font face = arial size = 2 color = black>
            This example shows how to view multiple ECW images from an Image Web
            Server, using the <b>native ECW plug-in</b>, and connect the view to
            a toolbar and progressbar using JavaScript. A slider is implemented
            to blend the top ECW image with the one underneath.
      </font>
</td>
```

If you have successfully completed these changes then you are ready to do the more complex tasks of adding JavaScript functions and deploying the view, toolbar, progressbar and slider objects that we will need for this page.

If you had difficulty with the exercise above then you should see assistance or training in Basic HTML scripting before proceeding.

Adding in the check for browser and plug-in version

It is a good idea to check the browser version when the page first loads and if too old branch the user to a ***'sorry.htm'*** page located in the ***/ecwplugins*** folder of your Image Web Server installation.

Copy the code in the box below to your clipboard.

```
// check for older browser versions no longer supported
if (parseFloat(navigator.appVersion) < 4.0) {
      document.location = "/ecwplugins/sorry.htm";
}
```

Paste the code into your web page at the location show below. (**Note: the technique we are using to show you where to place the code is that the surrounding code is shown non-bold, the new code is bold).**

```
<SCRIPT LANGUAGE=javascript>
<!--

// You will add your Javascript functions here.

// check for older browser versions no longer supported
if (parseFloat(navigator.appVersion) < 4.0) {
document.location = "/ecwplugins/sorry.htm";
}
```

Adding in the check plug-in versioning

It is important in your page to check that the user has an ECW Plug-in installed on their PC and that it is up to the version level that you require. When you installed Image Web Server there was put on your PC under the ***/ecwplugins*** folder all of the files required to upgrade a users system via Intranet/Internet to either install a plug-in from scratch if they have never had one before or upgrade to the newest version if they have an older one installed. The JavaScript function call that we are about to add will reference code in the JavaScript include file ***NCSCheck.js*** to automatically perform the checks and branch the user to a ***downloads.htm*** page to upgrade/install as required.

Copy the code in the box below to your clipboard.

```
// check for correct ECW Plug-in Version
ECWCheckPlugin();
```

Paste the code into your web page at the location show below.

```
// check for older browser versions no longer supported
if (parseFloat(navigator.appVersion) < 4.0) {
document.location = "/ecwplugins/sorry.htm";
}

// check for correct ECW Plug-in Version
ECWCheckPlugin();
```

Save your file and confirm it still operates correctly in your browser. If you had not previously installed the plug-in provided with your Image Web Server installation this code should have detected that and led you through the process of installing or upgrading the plug-in.

Set a Cookie to Define we are using the Native Control

The JavaScript Classes that were included in your page when you added the JavaScript Include files, can support two modes of operation; the **native** mode which uses the ECW Plug-in described in the preceding section, or the **java** applet which does not require the user to install the plug-in but instead downloads a **java applet** each time the user runs the page. (**Note:** although the java applet does not require a plug-in to be installed it does require the Sun Java Runtime Environment 1.4.2+ be installed). Because we are structuring our page for the native plug-in we will set the cookie to inform the JavaScript classes that create the view that this is our desired mode of operation.

Copy the code in the box below to your clipboard.

```
// Force the native plugin (ActiveX/NetscapePlugin)
setCookie("NCSPluginInstallMethod", "NATIVE");
```

Paste the code into your web page at the location show below.

```
// check for correct ECW Plug-in Version
ECWCheckPlugin();

// Force the native plugin (ActiveX/NetscapePlugin)
setCookie("NCSPluginInstallMethod", "NATIVE");
```

This single line of code demonstrates the power of the ERM supplied classes for automatically embedding the LayeredView object into your page. By simply changing this line to the following the Java Applet would be used:

```
setCookie("NCSPluginInstallMethod", "JAVA");
```

This is something that you can experiment with on your own if you have a need to deploy your pages using the Java Applet instead of the ECW Plug-in.

Modify the Body tag to handle the page Onload & Unload

It is important in building your pages that you make them cross-browser operational so they work in both Microsoft Internet Explorer and the Mozilla-based browser Netscape 7+ and Mozilla 1.4+. Netscape has a problem in that it detects the page loaded event prior to our control being fully initialized. This can cause your JavaScript to fail since it tries to call methods of the control before it is ready. To solve this we use the onload event of the body tag to introduce a delay which gives the objects in the page time to get fully loaded. While we are at it we will also add an unload event to do some housekeeping when the page is shut down.

Copy the code in the box below to your clipboard.

```
<body onLoad="onLoadCB();" onUnload="onUnloadCB();" border=20>
```

Paste the code into your web page at the location show below.

```
</head>
<body onLoad="onLoadCB();" onUnload="onUnloadCB();" border=20>
<table width ="100%" align = "center" border = 1 bordercolor = gray>
```

When the page loads it will call the function defined ‘onLoadCB()’ which we will now add to your JavaScript code. When the page unloads it will call the ‘onUnloadCB()’ function which we will also add now to your JavaScript code.

Adding the Onload and Unload Event Handler Functions

You will soon realize that much of what we do in these web pages is driven by **events** that are programmed to call JavaScript functions to perform certain tasks. If you are not familiar with JavaScript functions and events now would be a good time to do some study on the subject either with a good JavaScript manual or using the many good free web sites on the Internet.

Copy the code in the box below to your clipboard.

```
var tOnLoadTimer = null;

function onLoadCB() {
      tOnLoadTimer = setTimeout("LoadHandler()", 500);
}

function LoadHandler() {
      clearTimeout(tOnLoadTimer);
      alert(“The Page has Loaded”);
}

function onUnloadCB() {
      deleteCookie("NCSPluginInstallMethod");
}
```

Paste the code into your web page at the location show below.

```
// Force the native plugin (ActiveX/NetscapePlugin)
setCookie("NCSPluginInstallMethod", "NATIVE");

var tOnLoadTimer = null;
function onLoadCB() {
      tOnLoadTimer = setTimeout("LoadHandler()", 500);
}
function LoadHandler() {
      clearTimeout(tOnLoadTimer);
      alert("The Page has Loaded");
}
function onUnloadCB() {
      deleteCookie("NCSPluginInstallMethod");
}

//-->
</SCRIPT>
```

When the page loads it will call the function defined ‘onLoadCB()’ which we will now add to your JavaScript code. When the page unloads it will call the ‘onUnloadCB()’ function which we will also add now to your JavaScript code.

Save your file and confirm that when you click refresh on your browser the page does not generate any errors and brings up the alert with the message “The page has loaded”.

If your page does not do this correctly it is imperative that you debug the code to get any problems resolved since each step must be done correctly to

achieve an operational final result. Seek assistance from a DHTML (Dynamic HTML) developer if you are having problems.

In a later step we will modify the **LoadHandler()** function to remove the alert and insert code to automatically load in an ECW image. However, we need to insert the view control before we can make this happen.

Adding the LayeredView Control object to your page

Now we will get into the exciting part of this exercise where we add in a view control to your web page and then start displaying ECW images in the view. Adding the View Control has been made very simple for you because ERM has supplied a JavaScript Class that is referenced in the **NCSCreateView.js** file to automatically embed the required code into your page. This Class definition will detect whether you are running an IE or Mozilla style browser (and if you are in Windows or Macintosh OS) and handle it for you behind the scenes.

Copy the code in the box below to your clipboard.

```
<tr>
   <!-- This code adds the NCS Layered View object to the page. This
object is used to display the map images and is controlled by Javascript
calls to load the image layers, set tranparency, handle callback events
and so on. -->
   <TD>
      <SCRIPT LANGUAGE=javascript>
      NCSCreateView("ECWView1", "100%", "400",
PARAM_VIEW_ONPERCENTCOMPLETE, "ECWProgressBar.setProgress(value);");
      </SCRIPT>
   </TD>
</tr>
```

Paste the code into your web page at the location show below.

```
  </tr>
  <tr>
   <!-- This code adds the NCS Layered View object to the page. This
object is used to display the map images and is controlled by Javascript
calls to load the image layers, set tranparency, handle callback events
and so on. -->
   <TD>
      <SCRIPT LANGUAGE=javascript>
      NCSCreateView("ECWView1", "100%", "400",
PARAM_VIEW_ONPERCENTCOMPLETE, "ECWProgressBar.setProgress(value);");
      </SCRIPT>
   </TD>
</tr>
</table>
</body>
</html>
```

Now save the page and click refresh on your web browser. Click ok on the alert message and you should see a View window with the following text displayed

in the center of the view…."ECW Image View vs2.0". If you see this then congratulations! you have successfully added your first ECW Control. So that you fully understand what you have done we will now take you through the parameters that are specified in this call to the NCSCreateView code.

```
NCSCreateView("ECWView1", "100%", "400", PARAM_VIEW_ONPERCENTCOMPLETE,
"ECWProgressBar.setProgress(value);");
```

This code calls a function defined in the JavaScript include file **NCSCreateView.js** to instantiate a Layered View Control Object. This object is used to display images in a "stack" of images. These are the parameters passed to the NCSCreateView function:

"ECWView1"	The name of this Control. This name will be used in your JavaScript to refer to the View object
"100%"	The width of the View Control (can be in percent as shown or pixels)
"400"	The height of the object (can be in pixels as shown or in percent)
PARAM_VIEW_ONPERCENTCOMPLETE	This sets up a callback (event) which is fired whenever the views '% loaded' changes
"ECWProgressBar.setProgress(value);"	This defines what code to execute when the callback event is fired. In this case the Percent Complete value is passed to the. setProgress method of the ECWProgressBar object

Callbacks:

Callbacks are very important in scripting these web pages so let's take a moment and delve into what they are all about. Callbacks are what are commonly known in programming as **events.** ERM has defined a very useful set of these events or callbacks that are built into our Controls to allow you to enhance your web page in many ways. Examples of this are:

- Tracking the image loaded status with the onPercentProgress callback as shown in the code above.
- Detecting when the user has panned or zoomed the image using the onExtentsChange Callback so you can update a readout of the x/y corner values of the map, or request a new GIS overlay image as examples.
- Detecting mouse events for mouse up, mouse down and mouse move an using these events to call functions that display the coordinates at the mouse cursor, read the RGB value under the cursor and so on.

For a complete list of the callbacks currently supported refer to the Image Web Server manual.

Adding the Progress Bar Control Object

Since the View object is already set up to update a ProgressBar we need to add in this object and the supporting code before we can start adding images to the View otherwise the callback will try to execute and will fail.

Copy the code in the box below to your clipboard.

```
<tr>
   <TD class="NCSTableHeading" bgcolor='#dddddd'>
      <SCRIPT LANGUAGE=javascript>
            ECWProgressBar = new NCSProgressbar("NCS_UID_PROGRESS",
null, 60, true);
            ECWProgressBar.label = "Progress : ";
            document.writeln(ECWProgressBar.build());
      </SCRIPT>
  </TD>
</tr>
```

Paste the code into your web page at the location show below.

```
</tr>
<tr>
   <TD class="NCSTableHeading" bgcolor='#dddddd'>
      <SCRIPT LANGUAGE=javascript>
            ECWProgressBar = new NCSProgressbar("NCS_UID_PROGRESS",
null, 60, true);
            ECWProgressBar.label = "Progress : ";
            document.writeln(ECWProgressBar.build());
      </SCRIPT>
  </TD>
</tr>
</table>
</body>
</html>
```

Now save the page and click refresh on your web browser. Click ok on the alert message and you should see a View window and below that a progress bar that looks similar to the screenshot below.

In order that you will fully understand the code just added the following is a detailed description of what the elements are:

ECWProgressBar = new NCSProgressbar("NCS_UID_PROGRESS", null, 60, true); Here we are creating (instantiating) a new ProgressBar object and calling it **ECWProgressBar** the name by which it will be referred to from your JavaScript code. The parameters passed to the code that creates the object are as follows:	
" NCS_UID_PROGRESS"	This defines the type of object we wish to create.
"60"	Defines the width of the progress bar object in characters
true	Boolean that controls whether to display a text readout of percentage value to the right of the progress bar
ADDITIONAL LINES OF CODE	
ECWProgressBar.label = "Progress : ";	This defines what label to place to

	the left of the Progressbar object
document.writeln(ECWProgressBar.build());	This places the control into the HTML of the page.

Loading the initial ECW Image on Startup

We are now ready to modify the LoadHandler() function to automatically load an ECW image from your Image Web Server on page startup.

C
```
function LoadHandler() {
      clearTimeout(tOnLoadTimer);
      if (document.ECWView1.AddLayer("ECW",
                                                    "ecwp://" +
document.location.host + "/sampleiws/images/usa/1metercalif.ecw",
                                                    "RasterLayer1", "") <
0 ) {
            alert(document.ECWView1.GetLastErrorText());
      }
      document.ECWView1.SetBackGroundColor("#DDDDDD"); // sets the
background color of the View control
    ECWToolbar1.setCurrentItem("UID_VIEW_PAN"); // moved from
ECWToolbar1 setup code to handle a timing issue
}
```

Paste the code into your web page at the location show below(**replacing the current LoadHandler() function**).

```
function onLoadCB() {
      tOnLoadTimer = setTimeout("LoadHandler()", 500);
}
function LoadHandler() {
      clearTimeout(tOnLoadTimer);
      if (document.ECWView1.AddLayer("ECW",
                                                    "ecwp://" +
document.location.host + "/sampleiws/images/usa/1metercalif.ecw",
                                                    "RasterLayer1", "") <
0 ) {
            alert(document.ECWView1.GetLastErrorText());
      }
      document.ECWView1.SetBackGroundColor("#DDDDDD"); // sets the
background color of the View control
    ECWToolbar1.setCurrentItem("UID_VIEW_PAN"); // moved from
ECWToolbar1 setup code to handle a timing issue
}
function onUnloadCB() {
```

Now save the page and click refresh on your web browser. You should see a monochrome image loaded to the View Window from your image web server. If you do not then you should confirm that your image web server is functioning properly by trying the sample pages supplied with the installation.

The elements of the code that we have loaded are described below:

clearTimeout(tOnLoadTimer);	The LoadHandler() function was called from the Body onLoad event via a timer in onLoadCB(). This clears that

	timer.
document.ECWView1.AddLayer("ECW", "ecwp://" + d**ocument.location.host + "/sampleiws/images/usa/1metercalif.ecw", "RasterLayer1", "")**	
document.ECWView1.AddLayer	You should always precede calling an object on your page with 'document' otherwise it will not work in Netscape. ECWView1 is the name of the view control that we assigned when we created this object. .AddLayer is a very important **method** of the View Control and is used to create a new layer in the view.
"ECW"	The View Control supports different types of layers so we need to define the type which in this case is ECW indicating an image from an Image Web Server via the **ecwp** protocol.
"ecwp://" + d**ocument.location.host + "/sampleiws/images/usa/1metercalif.ecw**	This is the url for an image installed with your Image Web Server. Document.location.host picks up the name of your web server automatically.
"RasterLayer1"	Each layer added must have a unique **LayerName** so you can reference them for operations via JavaScript
""	The .AddLayer method supports setting Parameters when you add an image layer. Since we are not setting any we still need to put the double quotes as a blank for this field
alert(document.ECWView1.GetLastErrorText());	When you invoke .AddLayer() to add an image the call returns the layer count assigned to the layer (starting at 0). If the call fails then a –1 is returned. The if statement does a test for –1 and raises this alert with a call to the **GetLastErrorText()** method of the control to retrieve the error message.
document.ECWView1.SetBackGroundColor("#DDDDDD");	Sets the background color of the control defined as a hex RGB code.
ECWToolbar1.setCurrentItem("UID_VIEW_PAN");	This line of code sets the initial toolbar button which is selected on startup. We need to edit the loadhandler function for Netscape browsers to allow time for the view control to be fully loaded.

Pan and Zoom using the default & Chord Click Modes

You will notice that when you start the page now and the image loads, the default cursor mode is a hand, which indicates, the **pan** tool is active. Move your mouse over the image and left-click and drag to move the image around. You can also do a **chord click** which means to hold down the left mouse button and then click the right mouse button to switch the cursor into a small magnifying glass which indicates the **zoom mode** is active. Now drag up on the image to zoom out and down to zoom in.

Adding the Toolbar Support Function to Your Page

There is another object called the **ToolBar** object which allows the user to switch the cursor modes by clicking on the appropriate tool button which latches and calls a function in your JavaScript to change the mode. We will now add in this toolbar object to your page.

This object makes calls to a function in your JavaScript code to invoke the modes so we will first add in this function so we are ready when we later add the ToolBar object.

Copy the code in the box below to your clipboard.

```
function ECWToolbarCB(uid, value)
{
      if (uid == "UID_VIEW_PAN")
      {
            document.ECWView1.SetPointerMode(0);
      }
      else if (uid == "UID_VIEW_ZOOM")
      {
            document.ECWView1.SetPointerMode(1);
      }
      else if (uid == "UID_VIEW_ZOOMBOX")
      {
            document.ECWView1.SetPointerMode(3);
      }
      else if (uid == "UID_VIEW_POINTER")
      {
            document.ECWView1.SetPointerMode(2);
      }
      else if (uid == "UID_VIEW_RESET")
      {
            document.ECWView1.SetExtentsAll();
      }
      // This is used for switching images if an image array is defined
for the object
      else if (uid == "UID_IMAGE_COMBO")
      {
            document.ECWView1.DeleteAllLayers();
            if (document.ECWView1.AddLayer("ECW", value, "RasterLayer",
"") < 0 ) {
                  alert(document.ECWView1.GetLastErrorText());
            }
      }
}
```

Paste the code into your web page at the location show below.

```
function onUnloadCB() {
      deleteCookie("NCSPluginInstallMethod");
}
function ECWToolbarCB(uid, value)
{
      if (uid == "UID_VIEW_PAN")
      {
            document.ECWView1.SetPointerMode(0);
      }
      else if (uid == "UID_VIEW_ZOOM")
      {
            document.ECWView1.SetPointerMode(1);
      }
      else if (uid == "UID_VIEW_ZOOMBOX")
      {
            document.ECWView1.SetPointerMode(3);
      }
      else if (uid == "UID_VIEW_POINTER")
      {
            document.ECWView1.SetPointerMode(2);
      }
      else if (uid == "UID_VIEW_RESET")
      {
            document.ECWView1.SetExtentsAll();
      }
      // This is used for switching images if an image array is defined
for the object
      else if (uid == "UID_IMAGE_COMBO")
      {
            document.ECWView1.DeleteAllLayers();
            if (document.ECWView1.AddLayer("ECW", value, "RasterLayer",
"") < 0 ) {
                  alert(document.ECWView1.GetLastErrorText());
            }
      }
}
//-->
</SCRIPT>
```

Same your page and click refresh on your browser to confirm that no errors are generated. If all is ok then proceed to the next step which is adding the ToolBar Control into your page.

Adding the Toolbar Object to your Page with one Button

There is an object called the **ToolBar** object which allows the user to switch the cursor modes by clicking on the appropriate tool button which is set up to make a call to the function you just added to your JavaScript code. This function sorts out which button initiated the call and invokes the methods to

switch the cursor modes and change the toolbar button image to indicate a "latched" condition.

Copy the code in the box below to your clipboard.

```
<tr bgcolor='#dddddd'>
   <form name = "form1">
      <!-- This code adds the Toolbar object to the page. This object is
used to control the pointer mode for pan, zoom, zoombox and pointer -->
      <td>
         <SCRIPT LANGUAGE=javascript>
            var ECWToolbar1 = new NCSToolbar("UID_VIEW_TOOLBAR", null,
false, -1);
            // add in the single button for PAN
            ECWToolbar1.addButton("Pan around the image by click-
dragging the hand on the image",
                          "/ecwplugins/lib/bitmaps/buttons/roam.png",
"/ecwplugins/lib/bitmaps/buttons/roamSelect.png", true, ECWToolbarCB,
"UID_VIEW_PAN");

            document.writeln(ECWToolbar1.rebuild());
            // ECWToolbar1.setCurrentItem("UID_VIEW_PAN"); // moved to
loadhandler function to solve timing issue
            ECWToolbar1.getTable().border=0;
            ECWToolbar1.getTable().cellSpacing=2;
            ECWToolbar1.getTable().cellPadding=0;
         </SCRIPT>
      </td>
   </form>
</tr>
```

Paste the code into your web page at the location show below.

```
</tr>
  <tr bgcolor='#dddddd'>
   <form name = "form1">
      <!-- This code adds the Toolbar object to the page. This object is
used to control the pointer mode for pan, zoom, zoombox and pointer -->
      <td>
         <SCRIPT LANGUAGE=javascript>
            var ECWToolbar1 = new NCSToolbar("UID_VIEW_TOOLBAR", null,
false, -1);
            // add in the single button for PAN
            ECWToolbar1.addButton("Pan around the image by click-
dragging the hand on the image",
                          "/ecwplugins/lib/bitmaps/buttons/roam.png",
"/ecwplugins/lib/bitmaps/buttons/roamSelect.png", true, ECWToolbarCB,
"UID_VIEW_PAN");

            document.writeln(ECWToolbar1.rebuild());
            // ECWToolbar1.setCurrentItem("UID_VIEW_PAN"); // moved to
loadhandler function to solve timing issue
            ECWToolbar1.getTable().border=0;
            ECWToolbar1.getTable().cellSpacing=2;
            ECWToolbar1.getTable().cellPadding=0;
         </SCRIPT>
      </td>
   </form>
</tr>

   <!-- This code adds the NCS Layered View object to the page. This
object is used to display the map images and is controlled by Javascript
calls to load the image layers, set tranparency, handle callback events
and so on. -->
```

Save your page and click refresh on your browser and confirm that you now see a toolbar above the View window with a single PAN button shown.

In order that you will fully understand the code just added to define the toolbar object, the following is a detailed description of what the elements are:

var ECWToolbar1 = new NCSToolbar("UID_VIEW_TOOLBAR", null, false, -1); Here we are creating (instantiating) a new **NCSToolbar** object and calling it **ECWToolbar1,** the name by which it will be referred to from your JavaScript code. The parameters passed to the code that creates the object are as follows:	
"**UID_VIEW_TOOLBAR”**	This defines the type of object we wish to create. This is a unique id (uid) for this html element.
"**null** "	ojectID - write code to page directly when null specify HTML object if not null
false	bShowStatusBar shows a text status bar under the toolbar if true
-1	nCols specifies the number of colums before the toolbar wraps, -1 means no wrap

In order that you will fully understand the code just added to define the PAN toolbar button the following is a detailed description:

ECWToolbar1.addButton("Pan around the image by click-dragging the hand on the image", "/ecwplugins/lib/bitmaps/buttons/roam.png", "/ecwplugins/lib/bitmaps/buttons/roamSelect.png", true, ECWToolbarCB, "UID_VIEW_PAN"); Here we are using the .**addButton method** of the **NCSToolbar** object to add a tool button to the toolbar. The parameters passed to the code that creates the object are as follows:	
"Pan around the image by click-dragging the hand on the image”	This is a tool tip shown at the bottom of the browser and as an ALT message.
"/ecwplugins/lib/bitmaps/buttons/roam.png"	The path to load the buttons image file when the image is **not** selected
"/ecwplugins/lib/bitmaps/buttons/roamSelect.png"	The path to load the buttons image when the image **is** selected
true	Boolean value (true = exclusive & false = non exclusive). Exclusive means only that button can be on.
“ECWToolbarCB”	Function to call when user clicks the button. This is the function that you just added to your JavaScript code.
"UID_VIEW_PAN"	The Constant value passed to called function as 'uid'. This is assigned in the NCSConstants.js include file.

The following is a detailed description of the supporting lines of code:

document.writeln(ECWToolbar1.rebuild());	This code builds the toolbar object and writes it to your HTML page.
ECWToolbar1.setCurrentItem("UID_VIEW_PAN");	Sets the initial button selection to the PAN button.
ECWToolbar1.getTable().border=0;	Finds the table in which the toolbar is placed and sets the table border to 0.
ECWToolbar1.getTable().cellSpacing=2;	Finds the table in which the toolbar is placed and sets the table cellspacing to 2.
ECWToolbar1.getTable().cellPadding=0;	Finds the table in which the toolbar is placed and sets the table cellpadding to 0.

Adding the Remaining Buttons to your Toolbar Object

Now lets add in the rest of the buttons.

Copy the code in the box below to your clipboard.

```
ECWToolbar1.addButton("Zoom in or out of the image interactively by
click-dragging the magnifier on the image",

"/ecwplugins/lib/bitmaps/buttons/zoom.png",
"/ecwplugins/lib/bitmaps/buttons/zoomSelect.png", true, ECWToolbarCB,
"UID_VIEW_ZOOM");
                        ECWToolbar1.addButton("Zoom into the image by
click-dragging a zoom box over the image",

"/ecwplugins/lib/bitmaps/buttons/zoombox.png",
"/ecwplugins/lib/bitmaps/buttons/zoomBoxSelect.png", true, ECWToolbarCB,
"UID_VIEW_ZOOMBOX");
                        ECWToolbar1.addButton("Select pointer mode to
profile or select image features",

"/ecwplugins/lib/bitmaps/buttons/pointer.png",
"/ecwplugins/lib/bitmaps/buttons/pointerSelect.png", true, ECWToolbarCB,
"UID_VIEW_POINTER");
                        ECWToolbar1.addSpace ("SPACE",
"UID_VIEW_SPACE1");
                        ECWToolbar1.addButton("Reset the image to
maxiumum extents",

"/ecwplugins/lib/bitmaps/buttons/reset.png",
"/ecwplugins/lib/bitmaps/buttons/resetSelect.png", false, ECWToolbarCB,
"UID_VIEW_RESET");
                        //ECWToolbar1.addSpace ("SPACE",
"UID_VIEW_SPACE1");
                        //ECWToolbar1.addPopdownCombo ("Image : ",
imageArray, ECWToolbarCB, "UID_IMAGE_COMBO");

                        //ECWToolbar1.addPopdownCombo ("Image : ",
imageArray, ECWToolbarCB, "UID_IMAGE_COMBO");
```

Paste the code into your web page at the location show below.

```
   <SCRIPT LANGUAGE=javascript>
            var ECWToolbar1 = new NCSToolbar("UID_VIEW_TOOLBAR", null,
false, -1);
            // add in the single button for PAN
    ECWToolbar1.addButton("Pan around the image by click-dragging the
hand on the image",
                          "/ecwplugins/lib/bitmaps/buttons/roam.png",
"/ecwplugins/lib/bitmaps/buttons/roamSelect.png", true,
                                                    ECWToolbarCB,
"UID_VIEW_PAN");

      ECWToolbar1.addButton("Zoom in or out of the image interactively
by click-dragging the magnifier on the image",

"/ecwplugins/lib/bitmaps/buttons/zoom.png",
"/ecwplugins/lib/bitmaps/buttons/zoomSelect.png", true,
                                                    ECWToolbarCB,
"UID_VIEW_ZOOM");

      ECWToolbar1.addButton("Zoom into the image by click-dragging a
zoom box over the image",

"/ecwplugins/lib/bitmaps/buttons/zoombox.png",
"/ecwplugins/lib/bitmaps/buttons/zoomBoxSelect.png", true,
                                                    ECWToolbarCB,
"UID_VIEW_ZOOMBOX");

      ECWToolbar1.addButton("Select pointer mode to profile or select
image features",

"/ecwplugins/lib/bitmaps/buttons/pointer.png",
"/ecwplugins/lib/bitmaps/buttons/pointerSelect.png", true,
                                                    ECWToolbarCB,
"UID_VIEW_POINTER");

      ECWToolbar1.addSpace ("SPACE", "UID_VIEW_SPACE1");

      ECWToolbar1.addButton("Reset the image to maxiumum extents",

"/ecwplugins/lib/bitmaps/buttons/reset.png",
"/ecwplugins/lib/bitmaps/buttons/resetSelect.png", false,
                                                    ECWToolbarCB,
"UID_VIEW_RESET");

      //ECWToolbar1.addSpace ("SPACE", "UID_VIEW_SPACE1");
      //ECWToolbar1.addPopdownCombo ("Image : ", imageArray,
ECWToolbarCB, "UID_IMAGE_COMBO");

            document.writeln(ECWToolbar1.rebuild());
            ECWToolbar1.setCurrentItem("UID_VIEW_PAN");
            ECWToolbar1.getTable().border=0;
            ECWToolbar1.getTable().cellSpacing=2;
            ECWToolbar1.getTable().cellPadding=0;
         </SCRIPT>
```

Save your page and click refresh on your browser and confirm that you now see a toolbar that has the following buttons from left to right:

- **pan** – used to pan the map (default selection on startup)
- **zoom** – click mouse and drag up to zoom out and down to zoom in
- **zoombox** – click mouse and drag to draw zoom in rectangle
- **pointer** – combined with a callback used to show coordinates
- **reset** – this is non-exclusive and takes map back to start extents

Adding in a Second ECW Layer

By now you have built a fairly useful web page. Now let's enhance it further by adding a checkbox which when checked will add in a second ECW layer on top of the default image layer loaded on startup.

Copy the code in the box below to your clipboard.

```
<tr>
   <td>
      <font face = verdana size = 2 color =  red>Add SanDiego3i.ecw
            <input type = "checkbox" name = "chkLayer2" id = "chkLayer2"
onclick = "LoadECWLayer2(this.value)">
      </font>
   </td>
</tr>
```

Paste the code into your web page at the location show below.

```
</tr>
  <tr>
   <td>
      <font face = verdana size = 2 color =  red>Add SanDiego3i.ecw
            <input type = "checkbox" name = "chkLayer2" id = "chkLayer2"
onclick = "LoadECWLayer2(this.value)">
      </font>
   </td>
 </tr>
  <tr bgcolor='#dddddd'>
   <form name = "form1">
      <!-- This code adds the Toolbar object to the page. This object is
used to control the pointer mode for pan, zoom, zoombox and pointer -->
      <td>
```

Now save the page and click refresh on your web browser. Confirm that you have a new row above the toolbar which shows **Add SanDiego3i.ecw** in red with a checkbox to the right of it. Try clicking the checkbox … you should receive an error alert "Object Expected at line _____. This is because we are calling a non-existent JavaScript function which we will now add to the page.

Adding in the JavaScript Function for the Second ECW Layer

Now we will add the function that gets called when the checkbox is clicked. In the HTML code for the checkbox we have defined that this should be called **`LoadECWLayer2().`** Remember that JavaScript is case sensitive so you must be precise in putting in your capitals.

Copy the code in the box below to your clipboard.

```
// function to handle adding the second ECW layer
function LoadECWLayer2(value){
      if(value){
            if (document.ECWView1.AddLayer("ECW",
                                                "ecwp://" +
document.location.host + "/sampleiws/images/usa/SanDiego3i.ecw",
                                                "RasterLayer2", "") <
0 ) {
                  alert(document.ECWView1.GetLastErrorText());
            }
      }else{
            document.ECWView1.DeleteLayer("RasterLayer2");
      }

}
```

Paste the code into your web page at the location show below.

```
// function to handle adding the second ECW layer
function LoadECWLayer2(value){
   if(value){
      if (document.ECWView1.AddLayer("ECW", "ecwp://" +
document.location.host + "/sampleiws/images/usa/SanDiego3i.ecw",
"RasterLayer2", "") < 0 ) {
      alert(document.ECWView1.GetLastErrorText());
      }
  }else{
      document.ECWView1.DeleteLayer("RasterLayer2");
  }

}
//-->
</SCRIPT>
</head>
```

Note that the call to the function from the checkbox carries in the value **this.checked** that will be true if the checkbox is checked and false if unchecked. This is used to either create the layer and load the image if true, or delete the layer if false.

Save your page and refresh your browser. Now try clicking the checkbox on and you should see the small SanDiego3i.ecw image from your Image Web Server overlaid on top of the monochrome 1metercalif.ecw image. Unchecking the checkbox should remove the SanDiego3i.ecw image from the View.

Adding Transparency Slider Class to Your Page

When you have multiple images in the layered view control it is very handy to be able to adjust the transparency on the top image so you can blend it with the image below or temporarily make it fully transparent for unobstructed viewing of the lower image.

ERM supplies a slider class that makes it easy to add a slider to your page. First we need to add this class to your page as a JavaScript Include file.

Copy the code in the box below to your clipboard.

```
<SCRIPT LANGUAGE=javascript src="/ecwplugins/lib/scripts/NCSSlider.js"
></SCRIPT>
```

Paste the code into your web page at the location show below.

```
<SCRIPT LANGUAGE=javascript
src="/ecwplugins/lib/scripts/NCSToolbar.js"></SCRIPT>
<SCRIPT LANGUAGE=javascript
src="/ecwplugins/lib/scripts/NCSProgressbar.js"></SCRIPT>

<SCRIPT LANGUAGE=javascript src="/ecwplugins/lib/scripts/NCSSlider.js"
></SCRIPT>

</head>
```

Now we can begin to add in the slider object itself and the JavaScript code that it calls.

Adding the Transparency Slider Object to Your Page

We will now add some code that embeds a slider object into your page.

Copy the code in the box below to your clipboard.

```
<tr>
<td align = center>
<form name = "sliderform">
<SCRIPT LANGUAGE=javascript>
document.writeln("Drag the slider to alter the SanDiego layer
transparency.<br>");
var transpSlider = NCSSlider(650, 2, 'ECWUpdateTransparency', null);
transpSlider.setPosition(1.0);
bHaveNetscape4 = false;
var style = ((bHaveNetscape4) ? "SIZE=\"5\"" :
"style='width:120px;border:0;readonly=true;'");
document.writeln("<INPUT type='text' id='Transparency'
name='Transparency' " + style + " value='100 %'>");
</SCRIPT>
</form>
</td>
</tr>
```

Paste the code into your web page at the location show below.

```
<tr>
  <td align = center>
      <form name = "sliderform">
         <SCRIPT LANGUAGE=javascript>
            bHaveNetscape4 = false;
            document.writeln("Drag the slider to alter the SanDiego
layer transparency:");
            var style = ((bHaveNetscape4) ? "SIZE=\"5\"" :
"style='width:120px;border:0;readonly=true;'");
            document.writeln("<INPUT type='text' id='Transparency'
name='Transparency' " + style + " value='100 %'>");
            //   Slider Parameters:  NCSSlider(width, height,
moveCallback, stopCallback)
            var transpSlider = NCSSlider(650, 2,'ECWUpdateTransparency',
null);
            transpSlider.setPosition(1.0);
         </SCRIPT>
      </form>
   </td>
</tr>
</table>
</body>
</html>
```

A description of key elements of the code you just added is included below.

var transpSlider = NCSSlider(650, 2,'ECWUpdateTransparency', null); This code calls a function which creates a new instance of the NCSSlider class of objects and calls it by the name '**transpSlider'.** The parameters of the call to create the object are as follows	
650	**width** - This is the width of the slider object in pixels
2	**height** - This is the height of the slider object in pixels
ECWUpdateTransparency	**MoveCallback** - The name of the function to call when the slider is moved.
null	**StopCallback** - The name of the function to call when the slider is stopped. In this case none.
Supporting Code Explanations	
transpSlider.setPosition(1.0);	This calls the **.setPosition()** method of the **transpSlider** object to set the starting transparency.
bHaveNetscape4 = false; **var style = ((bHaveNetscape4) ? "SIZE=\"5\"" :** **"style='width:120px;border:0;readonly=true;'");** **document.writeln("<INPUT type='text'** **id='Transparency' name='Transparency' " + style** **+ " value='100 %'>");**	This code sets up the text box used for display of the percentage value as the slider is moved. It is updated each time the **ECWUpdateTransparency** function is called.

Adding the Transparency Slider JavaScript Function

Now we need to add the **`ECWUpdateTransparency()`** function that is called when the slider is moved.

Copy the code in the box below to your clipboard.

```
// function to set transparency based on slider position
var tTraspTimer = null;
function ECWUpdateTransparency(value) {
      var percent = parseFloat(value);
      if (percent > 1.0) percent = 1.0;
      if (percent < 0.0) percent = 0.0;
      bUseJavaPlugin = false;
      if (!bUseJavaPlugin) {
         document.ECWView1.SetLayerTransparency( "rasterlayer2", "#",
percent );
      }
      else { // need to set a delay for Java Applet or too many calls
will cause stack overflow
        clearTimeout(tTraspTimer);
        tTraspTimer =
setTimeout('document.ECWView1.SetLayerTransparency("RasterLayer2", "#",
' + percent + ');', 500);
      }
      document.sliderform.Transparency.value = Math.round(percent*100) +
" %";
}
```

Paste the code into your web page at the location show below.

```
// function to set transparency based on slider position
var tTraspTimer = null;
function ECWUpdateTransparency(value) {
   var percent = parseFloat(value);
   if (percent > 1.0) percent = 1.0;
   if (percent < 0.0) percent = 0.0;
   bUseJavaPlugin = false;
   if (!bUseJavaPlugin) {
      document.ECWView1.SetLayerTransparency( "rasterlayer2", "#",
percent );
   }
   else { // need to set a delay for Java Applet or too many calls will
cause stack overflow
      clearTimeout(tTraspTimer);
      tTraspTimer =
setTimeout('document.ECWView1.SetLayerTransparency("RasterLayer2", "#",
' + percent + ');', 500);
   }
   document.sliderform.Transparency.value = Math.round(percent*100) + "
%";
}

//-->
</SCRIPT>
</head>
<body onLoad="onLoadCB();" onUnload="onUnloadCB();" border=20>
```

A description of key elements of the code you just added is included below.

This slider has called the function with a value of the slider position which is converted to a float, checked to ensure it is between 0 and 1.0, then used to set the transparency for the rasterlayer2 layer with the .SetLayerTransparency method of the View object.

```
document.ECWView1.SetLayerTransparency( "rasterlayer2", "#", percent );
```

This code displays the percentage between 0 and 100 back to the text box on your page

```
document.sliderform.Transparency.value = Math.round(percent*100) + " %";
```

Adding Mouse Down and Mouse Move Callbacks

Now we will add in two new callbacks to the View that raise events when the left mouse button is clicked and when the mouse is moved. These callbacks will be used to display the Latitude / Longitude of the image under the mouse cursor. We will also add in a new on extents change callback that raises an event when the extents of the View are changed.

Copy the code in the box below to your clipboard.

```
NCSCreateView("ECWView1", "100%", "400", PARAM_VIEW_ONPERCENTCOMPLETE,
"ECWProgressBar.setProgress(value)",
            PARAM_VIEW_ONMOUSEDOWN,"onMouseDownCb();",
PARAM_VIEW_ONMOUSEMOVE,"onMouseDownCb();",
            PARAM_VIEW_ONEXTENTCHANGE,"onExtentsChangeCB();");
```

Paste the code into your web page at the location show below (replacing the existing NCSCreateView call.

```
<!-- This code adds the NCS Layered View object to the page. This object
is used to display the map images and is controlled by Javascript calls
to load the image layers, set tranparency, handle callback events and so
on. -->
   <TD>
      <SCRIPT LANGUAGE=javascript>
      NCSCreateView("ECWView1", "100%", "400",
PARAM_VIEW_ONPERCENTCOMPLETE, "ECWProgressBar.setProgress(value)",
            PARAM_VIEW_ONMOUSEDOWN,"onMouseDownCb();",
PARAM_VIEW_ONMOUSEMOVE,"onMouseDownCb();",
            PARAM_VIEW_ONEXTENTCHANGE,"onExtentsChangeCB();");
      </SCRIPT>
   </TD>
```

The callbacks are set up in sequences of callback name and the function that gets called when the event is triggered as shown in the table below.

Callback Name	Function to call on event
PARAM_VIEW_ONMOUSEDOWN	"onMouseDownCb();"
PARAM_VIEW_ONMOUSEMOVE	"onMouseDownCb();"
PARAM_VIEW_ONEXTENTCHANGE	"onExtentsChangeCB();"

Add in Handler Functions for Mouse and Extents Callbacks

Now we need to add JavaScript code that handles these events.

Copy the code in the box below to your clipboard.

```
// this function handles the on mouse up and on mouse move callback
events
function onMouseDownCb(nMask,ScreenX,ScreenY,WorldX,WorldY){
   document.form1.txtLat.value =
      document.ECWView1.GetCoordLatitude(WorldX,WorldY);
   document. form1.txtLong.value =
      document.ECWView1.GetCoordLongitude(WorldX,WorldY);
}

// this function handles the on extents change callback events
function onExtentsChangeCB(dWorldTLX, dWorldTLY, dWorldBRX, dWorldBRY){
 // do nothing for now
}
```

Paste the code into your web page at the location show.

```
// this function handles the on mouse up and on mouse move callback
events
function onMouseDownCb(nMask,ScreenX,ScreenY,WorldX,WorldY){
      document.form1.txtLat.value =
document.ECWView1.GetCoordLatitude(WorldX,WorldY);
      document.form1.txtLong.value =
document.ECWView1.GetCoordLongitude(WorldX,WorldY);
}

// this function handles the on extents change callback events
function onExtentsChangeCB(dWorldTLX, dWorldTLY, dWorldBRX, dWorldBRY){
 // do nothing for now
}

//-->
</SCRIPT>
</head>
```

An analysis of this code is below.

function onMouseDownCb(nMask,ScreenX,ScreenY,WorldX,WorldY){

The callback passes several parameters when it calls the function. These are described below.

nMask	An integer value that holds the mask encoding which mouse buttons and control keys were depressed. See the manual for details.
ScreenX	The cursor X coordinate in **pixels** relative to top left of the view control
ScreenY	The cursor Y coordinate in **pixels** relative to top left of the view control
WorldX	The cursor X coordinate in the world coordinates relative to top left of the view control.
WorldY	The cursor Y coordinate in the **world** coordinates relative to top left of the view control.

document.form1.txtLat.value = document.ECWView1.GetCoordLatitude(WorldX,WorldY);

Here we are deriving the Latitude values by passing in the World coordinates obtained during the event into the **.GetCoordLatitude** method of the View Control which performs a GDT conversion for the particular coordinate system. The result is written to a text box on the page which we will add next.

document.form1.txtLong.value =
document.ECWView1.GetCoordLongitude(WorldX,WorldY);

Same as above but for the Longitude values.

Add in Handler Text to Display boxes for Lat/Long Values

Now need to add the text boxes that were referenced above.

Copy the code in the box below to your clipboard.

```
<tr>
   <td>
      <form name = "form1">
      <font face = verdana size = 2 color =  red>Add SanDiego3i.ecw
      <input type = "checkbox" name = "chkLayer2" id = "chkLayer2"
onclick = "LoadECWLayer2(this.checked)">
      <!-- add in lat/long readout from mouse pointer -->
                
      <font face = verdana size = 2 color = brown>
      latitude <INPUT type="text" id = "txtLat" name = "txtLat"
STYLE="color: #FF0000; font-family: Verdana; font-weight: normal;
       font-size: 12px;       background-color: #99FFFF;" size="24"
maxlength="30"> 
      longitude <INPUT type="text" id = "txtLong" name = "txtLong"
STYLE="color: #FF0000; font-family: Verdana; font-weight: normal;
                  font-size: 12px; background-color: #99FFFF;"
size="24" maxlength="30"> 
       </font>
   </td>
 </tr>
```

Paste the code into your web page at the location (**you will be overwriting some existing code so be careful**)

```
<tr>
   <td>
      <form name = "form1">
      <font face = verdana size = 2 color =  red>Add SanDiego3i.ecw
      <input type = "checkbox" name = "chkLayer2" id = "chkLayer2"
onclick = "LoadECWLayer2(this.checked)">
      <!-- add in lat/long readout from mouse pointer -->
                 
      <font face = verdana size = 2 color = brown>
      latitude <INPUT type="text" id = "txtLat" name = "txtLat"
STYLE="color: #FF0000; font-family: Verdana; font-weight: normal;
       font-size: 12px;       background-color: #99FFFF;" size="24"
maxlength="30"> 
      longitude <INPUT type="text" id = "txtLong" name = "txtLong"
STYLE="color: #FF0000; font-family: Verdana; font-weight: normal;
       font-size: 12px; background-color: #99FFFF;" size="24"
maxlength="30"> 
       </font>
   </td>
 </tr>
 <tr bgcolor='#dddddd'>
      <!-- This code adds the Toolbar object to the page. This object is
used to control the pointer mode for pan, zoom, zoombox and pointer -->
      <td>
```

This code simply adds a form and two text boxes for the display of the Latitude and Longitude values calculated in the callback handlers.

Add in Text Boxes For the Map Extents Display

Now need to add the text boxes that were referenced above.

Copy the code in the box below to your clipboard.

```
      tlx <INPUT type="text" id = "txtTLX" name = "txtTLX" STYLE="color:
#FF0000; font-family: Verdana;font-weight: normal;font-size: 12px;
      background-color: #99FFFF;" size="22" maxlength="30">
      tly <INPUT type="text" id = "txtTLY" name = "txtTLY" STYLE="color:
#FF0000; font-family: Verdana;font-weight: normal;font-size: 12px;
      background-color: #99FFFF;" size="22" maxlength="30">
      brx <INPUT type="text" id = "txtBRX" name = "txtBRX" STYLE="color:
#FF0000; font-family: Verdana;font-weight: normal;font-size: 12px;
      background-color: #99FFFF;" size="22" maxlength="30">
      bry <INPUT type="text" id = "txtBRY" name = "txtBRY" STYLE="color:
#FF0000; font-family: Verdana;font-weight: normal;font-size: 12px;
      background-color: #99FFFF;" size="22" maxlength="30">
```

Paste the code into your web page at the.

```
<tr>
   <TD class="NCSTableHeading" bgcolor='#dddddd' align = center>
      tlx <INPUT type="text" id = "txtTLX" name = "txtTLX" STYLE="color:
#FF0000; font-family: Verdana;font-weight: normal;font-size: 12px;
      background-color: #99FFFF;" size="22" maxlength="30">
      tly <INPUT type="text" id = "txtTLY" name = "txtTLY" STYLE="color:
#FF0000; font-family: Verdana;font-weight: normal;font-size: 12px;
      background-color: #99FFFF;" size="22" maxlength="30">
      brx <INPUT type="text" id = "txtBRX" name = "txtBRX" STYLE="color:
#FF0000; font-family: Verdana;font-weight: normal;font-size: 12px;
      background-color: #99FFFF;" size="22" maxlength="30">
      bry <INPUT type="text" id = "txtBRY" name = "txtBRY" STYLE="color:
#FF0000; font-family: Verdana;font-weight: normal;font-size: 12px;
      background-color: #99FFFF;" size="22" maxlength="30">
   <SCRIPT LANGUAGE=javascript>
      ECWProgressBar = new NCSProgressbar("NCS_UID_PROGRESS", null, 40,
true);
      ECWProgressBar.label = "Progress : ";
      document.writeln(ECWProgressBar.build());
      </SCRIPT>
  </TD>
</tr>
```

This code adds four text boxes for the display of the map extents via the onExtentsChangeCB() function.

Now we will move the end of the form1 tag to a new location to tighten up the page formatting. It is currently located at:

```
      document.writeln(ECWToolbar1.rebuild());
            ECWToolbar1.setCurrentItem("UID_VIEW_PAN");
            ECWToolbar1.getTable().border=0;
            ECWToolbar1.getTable().cellSpacing=2;
            ECWToolbar1.getTable().cellPadding=0;

      </SCRIPT>
      </td>
   </form>
</tr>
  <tr>
```

Delete it from there and place it at the following location:

```
   <SCRIPT LANGUAGE=javascript>
            ECWProgressBar = new NCSProgressbar("NCS_UID_PROGRESS", null, 40,
true);
            ECWProgressBar.label = "Progress : ";
            document.writeln(ECWProgressBar.build());
      </SCRIPT>
  </TD>
</tr>
</form>
<tr>
```

Add in the onExtentsChangeCB JavaScript Function

Now need to add the text boxes that were referenced above.

Copy the code in the box below to your clipboard.

```
// this function handles the on extents change callback events
function onExtentsChangeCB(dWorldTLX, dWorldTLY, dWorldBRX, dWorldBRY){
      document.form1.txtTLX.value =
document.ECWView1.GetTopLeftWorldCoordinateX();
      document.form1.txtTLY.value =
document.ECWView1.GetTopLeftWorldCoordinateY();
      document.form1.txtBRX.value =
document.ECWView1.GetBottomRightWorldCoordinateX();
      document.form1.txtBRY.value =
document.ECWView1.GetBottomRightWorldCoordinateY();
}
```

Paste the code into your web page at the location (**you will be overwriting some existing code so be careful**)

```
// this function handles the on extents change callback events
function onExtentsChangeCB(dWorldTLX, dWorldTLY, dWorldBRX, dWorldBRY){
      document.form1.txtTLX.value =
document.ECWView1.GetTopLeftWorldCoordinateX();
      document.form1.txtTLY.value =
document.ECWView1.GetTopLeftWorldCoordinateY();
      document.form1.txtBRX.value =
document.ECWView1.GetBottomRightWorldCoordinateX();
      document.form1.txtBRY.value =
document.ECWView1.GetBottomRightWorldCoordinateY();
}

//-->
</SCRIPT>
</head>
<body onLoad="onLoadCB();" onUnload="onUnloadCB();" border=20>
```

This code uses view methods to derive the projection values for the top left and bottom right corners of the map view and writes them to the four text boxes which we added earlier.

Experimenting with other ECW Images

This completes the basic tutorial. Hopefully by now you have gained a great deal of knowledge and confidence in how to set up web pages for displaying images from Image Web Servers via the ecwp protocol.

We encourage you to experiment with your own images by adding them to your Image Web Server and substituting your ecwp url's for the once sample images we have been working with. Remember however, that the Layered

View control requires that all the images be in the same coordinate system (same Datum / Projection) so you must chose your images with this in mind.

You may link to images from our Image Web Server at **www.earthetc.com** and here are three images which are all Geodetic and can be used for your testing purposes:

A world mosaic of LandSat imagery **(over 2 Tbytes uncompressed!)**

ecwp://www.earthetc.com/images/geodetic/world/landsat742.ecw

A world topographic image:

ecwp://www.earthetc.com/images/world/gtopo30_2.ecw

USA DEM image:

ecwp://www.earthetc.com/images/usa/usadem.ecw