**Creating Web pages that use Web Parts controls in Visual Studio**

ASP.NET Web Parts controls are an integrated set of controls for creating Web sites that enable end users to modify the content, appearance, and behavior of Web pages directly in a browser. The topics in this section provide information on what Web Parts are, how they work, and how to use them to create user-customizable ASP.NET Web pages.

In many Web applications it is useful to be able to change the appearance of the content, as well as to allow users to select and arrange the content they want to see. ASP.NET Web Parts enable you to create Web pages that present modular content and that enable users to change the appearance and content to suit their preferences.

**1 . create a website with member ship and user login**

**To create a local IIS Web site**

1. Open Visual Studio.
2. On the **File** menu, click **New** **Web Site**.

The **New Web Site** dialog box appears.

1. Under **Visual Studio installed templates**, select **ASP.NET Web Site**.
2. In the **Location** list box, select **HTTP**. Click **Browse**.

The **Choose Location** dialog box appears.

1. Select **Local IIS**.
2. Open **Local Web Servers.**
3. Select **Default Web Site**.
4. Click the **Create New Web Application** icon () above the list of Web sites and then name the new Web site **membership**.
5. Click **Open**.

The **Choose Location** dialog box closes.

1. In the **Languages** box, click the programming language you prefer to work in.

The programming language you choose will be the default for your Web site, but you can set the programming languages for each page individually.

1. Click **OK** in the **New Web Site** dialog box.

Visual Web Developer creates the Web site and a new page named Default.aspx.

**Configuring Membership**

### To add a new folder to the Web site

1. In Solution Explorer, right-click the name of your Web site and click New Folder.
2. Name the folder MemberPages.

Before you work with ASP.NET membership, you must configure your application to enable membership and to set up users. You can use the Web Site Administration tool, which provides a wizard-like interface for making configuration settings. When you complete the Setup Wizard, a SQL Server database named ASPNETDB.MDF is created in the App\_Data folder of the project

**Configuring an ASP.NET Application to Use Membership**

You can also specify the default provider instance and options for that provider by configuring a provider in the membership section. You use the providers element to identify a provider to add to the collection of providers available for the application. You can identify your provider instance as the default provider by using the value of the name attribute as the defaultProvider value. When you specify a provider instance, you must also specify a valid connection string for that instance by using the [connectionStrings](http://msdn.microsoft.com/en-us/library/bf7sd233.aspx) section of the configuration. For example, the following Web.config file identifies a [SqlMembershipProvider](http://msdn.microsoft.com/en-us/library/system.web.security.sqlmembershipprovider.aspx) instance that connects to a SQL Server other than the local server.

<configuration>

 <connectionStrings>

 <add name="MySqlConnection" connectionString="Data

 Source=MySqlServer;Initial Catalog=aspnetdb;Integrated

 Security=SSPI;" />

 </connectionStrings>

 <system.web>

 <authentication mode="Forms" >

 <forms loginUrl="login.aspx"

 name=".ASPXFORMSAUTH" />

 </authentication>

 <authorization>

 <deny users="?" />

 </authorization>

 <membership defaultProvider="SqlProvider" userIsOnlineTimeWindow="15">

 <providers>

 <clear />

 <add

 name="SqlProvider"

 type="System.Web.Security.SqlMembershipProvider"

 connectionStringName="MySqlConnection"

 applicationName="MyApplication"

 enablePasswordRetrieval="false"

 enablePasswordReset="true"

 requiresQuestionAndAnswer="true"

 requiresUniqueEmail="true"

 passwordFormat="Hashed" />

 </providers>

 </membership>

 </system.web>

</configuration>

### To create a membership user

1. On the Website menu, click ASP.NET Configuration.
2. Select the Security tab, click the link to Use the security Setup Wizard to configure security step by step, and then click Next.
3. Proceed to Step 2 of the wizard and select the From the Internet option.

The wizard displays a page where you can select the authentication method that your Web site will use. This option specifies that your application will use Forms authentication, where users will log in to the application using a login page that you will create later in this walkthrough.

1. Click Next.

The wizard displays a message stating that user information will be stored using Advanced provider settings. By default, membership information is stored in a Microsoft SQL Server Express database file in the App\_Data folder of your Web site.

1. Click Next.

The wizard displays an option to create roles. You will perform this step separately later in the walkthrough.

1. Clear the Enable roles for this web site check box, and then click Next.

The wizard displays a page where you can create new users.

1. Enter information that defines a user of your application. Use the following values as guidelines (you can use any values that you like, but be sure to note your entries for later in the walkthrough):
	* User Name   Your name (with no spaces), or a sample name.
	* Password   A password. A strong password is required (one that includes uppercase and lowercase letters, punctuation, and that is at least eight characters long).
	* E-mail   Your personal e-mail address. Later in the walkthrough you will send yourself an e-mail message, so you need a legitimate e-mail address.
	* Security Question and Security Answer   Type a question and answer that can be used later if you need to recover your password.
2. Click Create User.

The wizard displays a confirmation page.

**Go to home page and click on access rules**

### To set up access rules for the MemberPages subdirectory

1. In the wizard, click Next.

The wizard displays a page that allows you to create access rules.

1. In the Add New Access Rule box, expand the node for your Web site.
2. Select MemberPages, the folder you created earlier.
3. Under Rule applies to, select Anonymous users.
4. Under Permission, select Deny.

The rule you are creating denies access to anonymous users — that is, users who have not logged in.

1. Click Add This Rule.

The new rule is displayed in the grid below. When users request a page from the MemberPages subdirectory, the rules are checked to determine whether the user is allowed access to the page.

1. Click Finish.

You are now done with the wizard. The wizard closes and you are returned to the Security tab of the Web Site Administration tool.

 Logging the User In

As part of your application, you need to establish the user's identity so that the application can perform actions — such as showing or hiding information — based on who the user is. To get the user's identity, you have the user log in.

In this walkthrough, you will add a link on the home page that takes users to a login page, and then you will create the login page.

**To create a home page with a login button**

1. Open or switch to the **Default.aspx** page of your site. (If you do not have a Default.aspx page, you can add one or use a different page.)
2. Switch to **Design** view.
3. Type static text such as **Welcome to our site** and, in the **Formatting** toolbar, use the **Block Format** drop-down list to format the text as Heading 1.
4. From the **Login** groupof the **Toolbox**, drag a [LoginStatus](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.loginstatus.aspx) control onto the page.

By default, the [LoginStatus](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.loginstatus.aspx) control is rendered as a link. When users click it, the application displays a login page. You can now create the login page.

**To create a login page**

1. In **Solution Explorer**, right-click your Web application and select **Add New Item**. Add a **Web Form** named **Login.aspx** to your site.

### Displaying Login Errors

The [Login](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.login.aspx) control includes validation to help users enter correct information. For example, if a user skips the password, a validator control displays an asterisk (\*) next to the Password box. You can provide more complete information for login errors by adding a [ValidationSummary](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.validationsummary.aspx) control to the page.

### To display detailed login errors

1. From the Validation group of the Toolbox, drag a [ValidationSummary](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.validationsummary.aspx) control onto the page.
2. In the Properties window for the [ValidationSummary](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.validationsummary.aspx) control, set the [ValidationGroup](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.validationsummary.validationgroup.aspx) property to Login1, which is the default ID of the [Login](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.login.aspx) control you added previously.

Creating a Simple Page with Web Parts

In this part of the walkthrough, you create a page that uses Web Parts controls to show static content. The first step in working with Web Parts is to create a page with these elements:

* A [WebPartManager](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpartmanager.aspx) control, which coordinates all Web Parts controls.
* One or more zones, which are composite controls that contain [WebPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpart.aspx) or other server controls and occupy a specified region of a page.

**To create a page for containing Web Parts controls**

1. Close the default page and add a new page named **WebPartsDemo.aspx**.
2. Switch to **Design** view.
3. In the **View** menu, make sure that the **ASP.NET Non-Visual Controls** option IS selected.

This enables you to see layout tags and controls that do not have a UI.

1. Position the insertion point in the **div** element.
2. From the **WebParts** tab of the **Toolbox**, drag a [WebPartManager](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpartmanager.aspx) control onto the page, between the newline character and the opening **<div>** tag.

The [WebPartManager](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpartmanager.aspx) control does not render any output, so it appears as a gray box on the designer surface.

1. Add a new line just after the [WebPartManager](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpartmanager.aspx) control.
2. In the **Block Format** list in the toolbar, select **Heading 1.**
3. In the heading, add the text "Web Parts Demonstration Page".
4. Add a new line after the text.
5. In the **Table** menu, click **Insert Table**, specify a table with one row and three columns, and then click **OK**.
6. Drag a [WebPartZone](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpartzone.aspx) control into the left table column.
7. Right-click the [WebPartZone](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpartzone.aspx) control, choose **Properties**, and set the following properties:

**ID**: "SidebarZone"

**HeaderText**: "Sidebar"

1. Drag a second [WebPartZone](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpartzone.aspx) control into the middle table column and set the following properties:

**ID**: "MainZone"

**HeaderText**: "Main"

1. Save the file, but do not close it yet.

Your page now has two zones that you can control separately. However, neither zone has any content, so the next step is to create content. For this walkthrough, you work with Web Parts controls that display only static content.

The layout of a Web Parts zone is specified by a **ZoneTemplate** element. Inside the zone template, you can add any ASP.NET control, including a custom Web Parts control, a user control, or a server control. In this walkthrough, you use the [Label](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.label.aspx) control to display static text. When you place a server control in a [WebPartZone](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpartzone.aspx) zone, ASP.NET treats the control as a Web Parts control at run time, which enables Web Parts features for the control.

**To create content for the main zone**

1. Switch to **Design** view.
2. From the **Standard** tab of the **Toolbox**, drag a [Label](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.label.aspx) control into the contents area of the zone whose **ID** property is set to MainZone.
3. Switch to **Source** view.

Notice that a **ZoneTemplate** element is added to wrap the [Label](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.label.aspx) control in the MainZone zone.

1. Add an attribute named **title** to the [Label](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.label.aspx) control and set its value to "Content".
2. Remove the **Text** attribute from the [Label](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.label.aspx) control.
3. Inside the [Label](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.label.aspx) control, add some text such as "<h2>Welcome to my Home Page</h2>".

The markup will look like the following example:

<asp:WebPartZone id="MainZone" runat="server" headertext="Main">

 <ZoneTemplate>

 <asp:Label ID="Label1" runat="server" Title="Content">

 <h2>Welcome to My Home Page</h2>

 </asp:Label>

 </ZoneTemplate>

</asp:WebPartZone>

1. Save the file.

Next, you will create a user control that can also be added to the page as a Web Parts control.

**To create a user control**

1. Add a new Web user control to your site and name it SearchUserControl.ascx. Make sure that the **Place source code in a separate file** is cleared.

This control will act as a search control.

|  |
| --- |
|  |
|  |

1. Switch to **Design** view.
2. From the **Standard** tab of the **Toolbox**, drag a [TextBox](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.textbox.aspx) control onto the page.
3. Add a blank line after the text box that you just added.
4. Drag a [Button](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.button.aspx) control onto the page and drop it below the text box.
5. Set the **Text** property of the [Button](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.button.aspx) control to "Search".
6. Switch to **Source** view.
7. Make sure that the markup for the user control looks like the following example:

C#

<%@ Control language="C#" classname="SearchUserControl" %>

<asp:TextBox runat="server" ID="TextBox1"></asp:TextBox>

<p>&nbsp;</p>

<asp:Button runat="server" ID=" Button1" text="Search" />

1. Save and close the file.

|  |
| --- |
|  |
|  |

Now you can add Web Parts controls to the Sidebar zone. You will add two controls to the Sidebar zone. One contains a list of links, and the other is the user control you created earlier in the walkthrough. You create the links by using a [Label](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.label.aspx) server control, similar to the way you created the static text for the Main zone. However, although the individual server controls contained in the user control could be contained directly in the zone (like the [Label](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.label.aspx) control), in this case they are not. Instead, they are part of the user control you created in the previous procedure. This demonstrates a common way to package controls and extra functionality in a user control, and then reference that control in a zone as a Web Parts control.

At run time, the Web Parts control set wraps both controls inside a [GenericWebPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.genericwebpart.aspx) controls. The generic part control becomes the parent control, and you can access the server control through the parent control's [ChildControl](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.genericwebpart.childcontrol.aspx) property. Using generic part controls enables standard Web server controls to have the same basic behavior and attributes as Web Parts controls that derive from the [WebPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpart.aspx) class.

**To add Web Parts controls to the sidebar zone**

1. Open the WebPartsDemo.aspx page.
2. Switch to **Design** view.
3. Drag the SearchUserControl.ascx user control from **Solution Explorer** into the SidebarZone zone.
4. Add an attribute named **title** to the user control element and set its value to "Search".
5. Save the WebPartsDemo.aspx page.
6. Switch to **Source** view.
7. Inside the **asp:WebPartZone** element for the SidebarZone zone, add a **Label** control that contains links. In the opening tag for the user control, add a **title** attribute with a value of "My Links".

The markup looks like the following example:

 <asp:WebPartZone id="SidebarZone" runat="server"

 headertext="Sidebar">

 <ZoneTemplate>

 <asp:Label runat="server" id="linksPart" title="My Links">

 <a href="http://www.asp.net">ASP.NET site</a>

 <br />

 <a href="http://www.gotdotnet.com">GotDotNet</a>

 <br />

 <a href="http://www.contoso.com">Contoso.com</a>

 <br />

 </asp:Label>

 <uc1:SearchUserControl id="SearchUserControl1" runat="server"

 title="Search" />

 </ZoneTemplate>

</asp:WebPartZone>

1. Save and close the file.

Now you can test the page.

**To test the page**

* Load the page in a browser.

The page displays the two zones. The following figure shows the page.



In the title bar of each control is a down arrow that provides access to a verbs menu of actions that you can perform on a control. Click the verbs menu in one of the controls, and then click the **Minimize** verb and note that the control is minimized. From the verbs menu, click **Restore**, and the control returns to its normal size.

**Enabling Users to Edit Pages and Change Layout**

Web Parts lets users change the layout of Web Parts controls by dragging them from one zone to another. In addition to allowing users to move [WebPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpart.aspx) controls from one zone to another, you can allow users to edit various characteristics of the controls, including their appearance, layout, and behavior. The Web Parts control set provides basic editing functionality for [WebPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpart.aspx) controls. Although you will not do so in this walkthrough, you can also create custom editor controls that allow users to edit the features of [WebPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpart.aspx) controls. As with changing the location of a [WebPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpart.aspx) control, editing a control's properties relies on ASP.NET personalization to save the changes that users make.

In this part of the walkthrough, you add the ability for users to edit the basic characteristics of any [WebPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpart.aspx) control on the page. To enable these features, you add another custom user control to the page, along with an **asp:editorzone** element and two editing controls.

**To create a user control that enables changing page layout**

1. In Visual Studio, in the **File** menu, click **New**, and then click **File**.
2. In the **Add New Item** dialog box, select **Web User Control**. Name the new file DisplayModeMenu.ascx. Clear the **Place source code in separate file** box.
3. Click **Add** to create the new control.
4. Switch to **Source** view.
5. Remove all the existing markup in the new file, and then paste in the following code.

This user control code uses features of the Web Parts control set that enable a page to change its view or display mode. It also enables you to change the physical appearance and layout of the page while you are in certain display modes.

<%@ control language="C#" classname="DisplayModeMenuCS"%>

<script runat="server">

 // Use a field to reference the current WebPartManager control.

 WebPartManager \_manager;

 void Page\_Init(object sender, EventArgs e)

 {

 Page.InitComplete += new EventHandler(InitComplete);

 }

 void InitComplete(object sender, System.EventArgs e)

 {

 \_manager = WebPartManager.GetCurrentWebPartManager(Page);

 String browseModeName = WebPartManager.BrowseDisplayMode.Name;

 // Fill the drop-down list with the names of supported display modes.

 foreach (WebPartDisplayMode mode in

 \_manager.SupportedDisplayModes)

 {

 String modeName = mode.Name;

 // Make sure a mode is enabled before adding it.

 if (mode.IsEnabled(\_manager))

 {

 ListItem item = new ListItem(modeName, modeName);

 DisplayModeDropdown.Items.Add(item);

 }

 }

 // If Shared scope is allowed for this user, display the

 // scope-switching UI and select the appropriate radio

 // button for the current user scope.

 if (\_manager.Personalization.CanEnterSharedScope)

 {

 Panel2.Visible = true;

 if (\_manager.Personalization.Scope ==

 PersonalizationScope.User)

 RadioButton1.Checked = true;

 else

 RadioButton2.Checked = true;

 }

 }

 // Change the page to the selected display mode.

 void DisplayModeDropdown\_SelectedIndexChanged(object sender,

 EventArgs e)

 {

 String selectedMode = DisplayModeDropdown.SelectedValue;

 WebPartDisplayMode mode =

 \_manager.SupportedDisplayModes[selectedMode];

 if (mode != null)

 \_manager.DisplayMode = mode;

 }

 // Set the selected item equal to the current display mode.

 void Page\_PreRender(object sender, EventArgs e)

 {

 ListItemCollection items = DisplayModeDropdown.Items;

 int selectedIndex =

 items.IndexOf(items.FindByText(\_manager.DisplayMode.Name));

 DisplayModeDropdown.SelectedIndex = selectedIndex;

 }

 // Reset all of a user's personalization data for the page.

 protected void LinkButton1\_Click(object sender, EventArgs e)

 {

 \_manager.Personalization.ResetPersonalizationState();

 }

 // If not in User personalization scope, toggle into it.

 protected void RadioButton1\_CheckedChanged(object sender, EventArgs e)

 {

 if (\_manager.Personalization.Scope ==

 PersonalizationScope.Shared)

 \_manager.Personalization.ToggleScope();

 }

 // If not in Shared scope, and if user has permission, toggle

 // the scope.

 protected void RadioButton2\_CheckedChanged(object sender,

 EventArgs e)

 {

 if (\_manager.Personalization.CanEnterSharedScope &&

 \_manager.Personalization.Scope ==

 PersonalizationScope.User)

 \_manager.Personalization.ToggleScope();

 }

</script>

<div>

 <asp:Panel ID="Panel1" runat="server"

 Borderwidth="1"

 Width="230"

 BackColor="lightgray"

 Font-Names="Verdana, Arial, Sans Serif" >

 <asp:Label ID="Label1" runat="server"

 Text="&nbsp;Display Mode"

 Font-Bold="true"

 Font-Size="8"

 Width="120" />

 <asp:DropDownList ID="DisplayModeDropdown" runat="server"

 AutoPostBack="true"

 Width="120"

 OnSelectedIndexChanged="DisplayModeDropdown\_SelectedIndexChanged" />

 <asp:LinkButton ID="LinkButton1" runat="server"

 Text="Reset User State"

 ToolTip="Reset the current user's personalization data for

 the page."

 Font-Size="8"

 OnClick="LinkButton1\_Click" />

 <asp:Panel ID="Panel2" runat="server"

 GroupingText="Personalization Scope"

 Font-Bold="true"

 Font-Size="8"

 Visible="false" >

 <asp:RadioButton ID="RadioButton1" runat="server"

 Text="User"

 AutoPostBack="true"

 GroupName="Scope"

 OnCheckedChanged="RadioButton1\_CheckedChanged" />

 <asp:RadioButton ID="RadioButton2" runat="server"

 Text="Shared"

 AutoPostBack="true"

 GroupName="Scope"

 OnCheckedChanged="RadioButton2\_CheckedChanged" />

 </asp:Panel>

 </asp:Panel>

</div>

**To enable users to change the layout**

1. Open the WebPartsDemo.aspx page.
2. Switch to **Design** view.
3. Add a blank line after the "Web Parts Demonstration Page" text that you added earlier.
4. From **Solution Explorer**, drag the DisplayModeMenu.ascx user control into the WebPartsDemo.aspx page and drop it on the blank line.
5. From the **WebParts** tab of the **Toolbox**, drag an [EditorZone](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.editorzone.aspx) control to the remaining open table cell in the WebPartsDemo.aspx page.
6. Switch to **Source** view.
7. From the **WebParts** tab of the **Toolbox**, drag an [AppearanceEditorPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.appearanceeditorpart.aspx) control and a [LayoutEditorPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.layouteditorpart.aspx) control into the [EditorZone](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.editorzone.aspx) control.

The resulting markup in the table cell will look similar to the following code:

<td valign="top">

 <asp:EditorZone ID="EditorZone1" runat="server">

 <ZoneTemplate>

 <asp:AppearanceEditorPart ID="AppearanceEditorPart1"

 runat="server" />

 <asp:LayoutEditorPart ID="LayoutEditorPart1" runat="server" />

 </ZoneTemplate>

 </asp:EditorZone>

</td>

**o test layout changes**

1. Load the page in a browser.
2. In the **Display Mode** menu, click **Edit**.

The zone titles are displayed.

1. Drag the **My Links** control by its title bar from the Sidebar zone to the bottom of the Main zone.

The page will look like the following:



1. Click **Display Mode**, and then click **Browse**.

The page is refreshed, the zone names disappear, and the **My Links** control remains where you positioned it.

1. To demonstrate that personalization is working, close the browser, and then load the page again. The changes you made are saved for future browser sessions.
2. In the **Display Mode** menu, click **Edit**.

Each control on the page is now displayed with a down arrow in its title bar, which contains the verbs drop-down menu.

1. Click the arrow to display the verbs menu on the **My Links** control and then click **Edit**.

The [EditorZone](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.editorzone.aspx) control appears. It displays the [EditorPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.editorpart.aspx) controls that you added.

1. In the **Appearance** section of the edit control, change the title to **My Favorites**. In the **Chrome Type** list, select **Title Only**, and then click **Apply**.

The following figure shows the page in edit mode.



1. In the **Display Mode** menu, click **Browse** to return to browse mode.

The control now has an updated title and no border, as shown in the following figure.



**Adding Web Parts at Run Time**

You can also enable users to add Web Parts controls to their page at run time. To do so, configure the page with a Web Parts catalog, which contains a list of Web Parts controls that you want to make available to users.

**To allow users to add Web Parts at run time**

1. Open the WebPartsDemo.aspx page.
2. Switch to Source view.
3. From the **WebParts** tab of the Toolbox, drag a [CatalogZone](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.catalogzone.aspx) control into the right column of the table, beneath the [EditorZone](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.editorzone.aspx) control.

Both controls can be in the same table cell because they will not be displayed at the same time.

1. In the Properties pane, assign the string **Add Web Parts** to the **HeaderText** property of the [CatalogZone](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.catalogzone.aspx) control.
2. Switch to Design view
3. From the **WebParts** tab of the Toolbox, drag a [DeclarativeCatalogPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.declarativecatalogpart.aspx) control into the content area of the [CatalogZone](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.catalogzone.aspx) control.
4. Click the arrow in the upper right corner of the [DeclarativeCatalogPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.declarativecatalogpart.aspx) control to expose its Tasks menu, and then select **Edit Templates**.
5. From the **Standard** tab of the Toolbox, drag a [FileUpload](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.fileupload.aspx) control and a [Calendar](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.calendar.aspx) control into the **WebPartsTemplate** section of the [DeclarativeCatalogPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.declarativecatalogpart.aspx) control.
6. Switch to **Source** view and inspect the source code of the **CatalogZone** control.

Notice that the [DeclarativeCatalogPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.declarativecatalogpart.aspx) control contains a **WebPartsTemplate** element with the two enclosed server controls that you will be able to add to your page from the catalog.

|  |
| --- |
|  |
|  |

1. Add a **Title** property to each of the controls that you added to the catalog, and set the property to the names shown in the following example. Even though the title is not a property you can normally set for these controls at design time, when a user adds these controls to a [WebPartZone](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.webpartzone.aspx) zone from the catalog at run time, they are each wrapped with a [GenericWebPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.genericwebpart.aspx) control. This enables them to act as Web Parts controls. Therefore, they will be able to display titles.

The markup for the controls contained in the [DeclarativeCatalogPart](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.webparts.declarativecatalogpart.aspx) control will look like the following example:

Copy Code

<asp:DeclarativeCatalogPart ID="DeclarativeCatalogPart1"

 runat="server">

 <WebPartsTemplate>

 <asp:Calendar ID="Calendar1"

 runat="server"

 title="My Calendar" />

 <asp:FileUpload ID="FileUpload1"

 runat="server"

 title="Upload Files" />

 </WebPartsTemplate>

</asp:DeclarativeCatalogPart>

1. Save the page.

You can now test the catalog.

**To test the Web Parts catalog**

1. Load the page in a browser.
2. In the **Display Mode** menu, click **Catalog**.

The catalog titled **Add Web Parts** is displayed.

1. Drag the **My Favorites** control from the Main zone back to the top of the Sidebar zone.
2. In the **Add Web Parts** catalog, select both check boxes, and then select **Main** from the list of available zones
3. Click **Add** in the catalog.

The controls are added to the Main zone. If you want, you can add multiple instances of controls from the catalog to your page. The following figure shows the page with the file upload control and the calendar in the Main zone:

**Controls added to Main zone from the catalog**



1. In the **Display Mode** menu, click **Browse**.

The catalog disappears and the page is refreshed.

1. Close the browser and then load the page again.

**End of the document**