**Working with ASP.NET Master Pages Programmatically**

[**http://msdn.microsoft.com/en-us/library/c8y19k6h%28v=vs.100%29.aspx**](http://msdn.microsoft.com/en-us/library/c8y19k6h%28v=vs.100%29.aspx)

**.NET Framework 4**

[Other Versions](javascript:;)

29 out of 43 rated this helpful - [Rate this topic](http://msdn.microsoft.com/en-us/library/c8y19k6h%28v=vs.100%29.aspx#feedback)

You can perform a number of common tasks programmatically with master pages, including the following:

* Accessing members that are defined on the master page, which can consist of public properties and methods or controls.
* Attaching master pages to a content page dynamically.

A Visual Studio project with source code is available to accompany this topic: [Download](http://go.microsoft.com/fwlink/?LinkId=192413).

[Accessing Members on the Master Page](javascript:void(0))

To provide access to members of the master page, the [Page](http://msdn.microsoft.com/en-us/library/system.web.ui.page%28v=vs.100%29.aspx) class exposes a [Master](http://msdn.microsoft.com/en-us/library/system.web.ui.page.master%28v=vs.100%29.aspx) property. To access members of a specific master page from a content page, you can create a strongly typed reference to the master page by creating a [@ MasterType](http://msdn.microsoft.com/en-us/library/ms228274%28v=vs.100%29.aspx) directive. The directive allows you to point to a specific master page. When the page creates its [Master](http://msdn.microsoft.com/en-us/library/system.web.ui.page.master%28v=vs.100%29.aspx) property, the property is typed to the referenced master page.

For example, you might have a master page named MasterPage.master that is the class name MasterPage\_master. You might create [@ Page](http://msdn.microsoft.com/en-us/library/ydy4x04a%28v=vs.100%29.aspx) and @ MasterType directives that look like the following:

<%@ Page  masterPageFile="~/MasterPage.master"%>

<%@ MasterType  virtualPath="~/MasterPage.master"%>

When you use a @ MasterType directive, such as the one in the example, you can reference members on the master page as in the following example:

C#

[VB](http://msdn.microsoft.com/en-us/library/c8y19k6h%28v=vs.100%29.aspx?cs-save-lang=1&cs-lang=vb#code-snippet-1)

CompanyName.Text = Master.CompanyName;

The [Master](http://msdn.microsoft.com/en-us/library/system.web.ui.page.master%28v=vs.100%29.aspx) property of the page is already typed to MasterPage\_master.

**Getting the Values of Controls on the Master Page**

At run time, the master page is merged with the content page, so the controls on the master page are accessible to content page code. (If the master page contains controls in a [ContentPlaceHolder](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.contentplaceholder%28v=vs.100%29.aspx) control, those controls are not accessible if overridden by a [Content](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.content%28v=vs.100%29.aspx) control from the content page.) The controls are not directly accessible as master-page members because they are protected. However, you can use the [FindControl](http://msdn.microsoft.com/en-us/library/system.web.ui.page.findcontrol%28v=vs.100%29.aspx) method to locate specific controls on the master page. If the control that you want to access is inside a [ContentPlaceHolder](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.contentplaceholder%28v=vs.100%29.aspx) control on the master page, you must first get a reference to the [ContentPlaceHolder](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.contentplaceholder%28v=vs.100%29.aspx) control, and then call its [FindControl](http://msdn.microsoft.com/en-us/library/system.web.ui.control.findcontrol%28v=vs.100%29.aspx) method to get a reference to the control.

The following example shows how you can get a reference to controls on the master page. One of the controls being referenced is in a [ContentPlaceHolder](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.contentplaceholder%28v=vs.100%29.aspx) control and the other is not.

C#

[VB](http://msdn.microsoft.com/en-us/library/c8y19k6h%28v=vs.100%29.aspx?cs-save-lang=1&cs-lang=vb#code-snippet-2)

// Gets a reference to a TextBox control inside a ContentPlaceHolder

ContentPlaceHolder mpContentPlaceHolder;

TextBox mpTextBox;

mpContentPlaceHolder =

(ContentPlaceHolder)Master.FindControl("ContentPlaceHolder1");

if(mpContentPlaceHolder != null)

{

mpTextBox = (TextBox) mpContentPlaceHolder.FindControl("TextBox1");

if(mpTextBox != null)

{

mpTextBox.Text = "TextBox found!";

}

}

// Gets a reference to a Label control that is not in a

// ContentPlaceHolder control

Label mpLabel = (Label) Master.FindControl("masterPageLabel");

if(mpLabel != null)

{

Label1.Text = "Master page label = " + mpLabel.Text;

}

You can access the contents of the master page's [ContentPlaceHolder](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.contentplaceholder%28v=vs.100%29.aspx) controls by using the [FindControl](http://msdn.microsoft.com/en-us/library/system.web.ui.page.findcontrol%28v=vs.100%29.aspx) method, as shown above. If the [ContentPlaceHolder](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.contentplaceholder%28v=vs.100%29.aspx) control has been merged with content from a [Content](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.content%28v=vs.100%29.aspx) control, the [ContentPlaceHolder](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.contentplaceholder%28v=vs.100%29.aspx) control will not contain its default content. Instead, it will contain the text and controls that are defined in the content page.

[Attaching Master Pages Dynamically](javascript:void(0))

In addition to specifying a master page declaratively (in the @ Page directive or in the configuration file), you can attach a master page dynamically to a content page. Because the master page and content page are merged during the initialization stage of page processing, a master page must be assigned before then. Typically, you assign a master page dynamically during the [PreInit](http://msdn.microsoft.com/en-us/library/system.web.ui.page.preinit%28v=vs.100%29.aspx) stage, as in the following example:

C#

[VB](http://msdn.microsoft.com/en-us/library/c8y19k6h%28v=vs.100%29.aspx?cs-save-lang=1&cs-lang=vb#code-snippet-3)

void Page\_PreInit(Object sender, EventArgs e)

{

this.MasterPageFile = "~/NewMaster.master";

}

**Strong Typing for Dynamic Master Pages**

If the content page assigns a strong type to the master page by using a @ MasterType directive, the type must apply to any master page that you assign dynamically. If you intend to select a master page dynamically, it is recommended that you create a base class from which your master pages derive. The base master-page class can then define the properties and methods that the master pages have in common. In the content page, when you assign a strong type to the master page by using a @ MasterType directive, you can assign it to the base class instead of to an individual master page.

The following examples show how to create a base master-page type that can be used by multiple master pages. The examples consist of a base type that is derived from the [MasterPage](http://msdn.microsoft.com/en-us/library/system.web.ui.masterpage%28v=vs.100%29.aspx) control, two master pages that inherit from the base type, and a content page that allows users to select a master page dynamically by using a query string (?color=green). The base master type defines a property named MyTitle. One of the master pages overrides the MyTitle property, and the other one does not. The content page displays the MyTitle property as the page's title. The title of the page will therefore vary depending on which master page has been selected.

This is the base master-page type. It belongs in the App\_Code directory.

C#

[VB](http://msdn.microsoft.com/en-us/library/c8y19k6h%28v=vs.100%29.aspx?cs-save-lang=1&cs-lang=vb#code-snippet-4)

using System;

using System.Data;

using System.Configuration;

using System.Web;

using System.Web.Security;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Web.UI.WebControls.WebParts;

using System.Web.UI.HtmlControls;

public class BaseMaster : System.Web.UI.MasterPage

{

public virtual String MyTitle

{

get { return "BaseMaster Title"; }

}

}

This is the first master page, which displays a blue background. Notice that the Inherits attribute in the @ Master directive references the base type.

C#

[VB](http://msdn.microsoft.com/en-us/library/c8y19k6h%28v=vs.100%29.aspx?cs-save-lang=1&cs-lang=vb#code-snippet-5)

<%@ Master Language="C#" Inherits="BaseMaster"

ClassName="MasterBlue" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"

"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<script runat="server">

// No property here that overrrides the MyTitle property of the base master.

</script>

<html xmlns="http://www.w3.org/1999/xhtml" >

<head id="Head1" runat="server">

<title>No title</title>

</head>

<body>

<form id="form1" runat="server">

<div style="background-color:LightBlue">

<asp:contentplaceholder id="ContentPlaceHolder1"

runat="server">

Content from MasterBlue.

</asp:contentplaceholder>

</div>

</form>

</body>

</html>

This is the second master page. It is the same as the first master page, except that it displays a green background, and it overrides the MyTitle property that is defined in the base type.

C#

[VB](http://msdn.microsoft.com/en-us/library/c8y19k6h%28v=vs.100%29.aspx?cs-save-lang=1&cs-lang=vb#code-snippet-6)

<%@ Master Language="C#" Inherits="BaseMaster"

ClassName="MasterGreen" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"

"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<script runat="server">

public override String MyTitle

{

get { return "MasterGreen Title"; }

}

</script>

<html xmlns="http://www.w3.org/1999/xhtml" >

<head id="Head1" runat="server">

<title>No title</title>

</head>

<body>

<form id="form1" runat="server">

<div style="background-color:LightGreen">

<asp:contentplaceholder id="ContentPlaceHolder1"

runat="server">

Content from MasterGreen.

</asp:contentplaceholder>

</div>

</form>

</body>

</html>

This is the content page, which allows users to select a master page based on a query string provided with the request. The @ MasterType directive, which assigns a strong type to the page's [Master](http://msdn.microsoft.com/en-us/library/system.web.ui.page.master%28v=vs.100%29.aspx) property, references the base type.

C#

[VB](http://msdn.microsoft.com/en-us/library/c8y19k6h%28v=vs.100%29.aspx?cs-save-lang=1&cs-lang=vb#code-snippet-7)

<%@ Page Language="C#" Title="Content Page" MasterPageFile="~/MasterBlue.master"%>

<%@ MasterType TypeName="BaseMaster" %>

<script runat="server">

protected void Page\_PreInit(Object sender, EventArgs e)

{

this.MasterPageFile = "MasterBlue.master";

if(Request.QueryString["color"] == "green")

{

this.MasterPageFile = "MasterGreen.master";

}

this.Title = Master.MyTitle;

}

</script>

<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1"

Runat="Server">

Content from Content page.

</asp:Content>

[See Also](javascript:void(0))

**Other Resources**

[ASP.NET Master Pages](http://msdn.microsoft.com/en-us/library/wtxbf3hh%28v=vs.100%29.aspx)