MASTER PAGE

ASP.NET master pages allow you to create a consistent layout for the pages in your application. A single master page defines the look and feel and standard behavior that you want for all of the pages (or a group of pages) in your application. You can then create individual content pages that contain the content you want to display. When users request the content pages, they merge with the master page to produce output that combines the layout of the master page with the content from the content page.

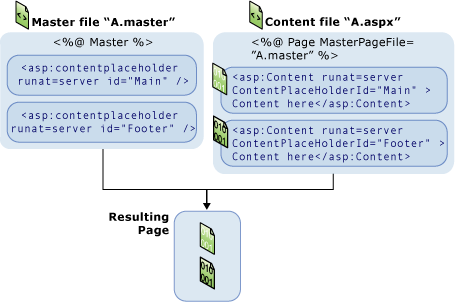
**Replaceable Content Placeholders**

In addition to static text and controls that will appear on all pages, the master page also includes one or more [ContentPlaceHolder](https://msdn.microsoft.com/en-us/library/1k8dts1d(v=vs.100)) controls. These placeholder controls define regions where replaceable content will appear. In turn, the replaceable content is defined in content pages.

### Content Pages

You define the content for the master page's placeholder controls by creating individual content pages, which are ASP.NET pages (.aspx files and, optionally, code-behind files) that are bound to a specific master page. The binding is established in the content page's @ Page directive by including a [MasterPageFile](https://msdn.microsoft.com/en-us/library/w3sts40t(v=vs.100)) attribute that points to the master page to be used. For example, a content page might have the following @ Page directive, which binds it to the Master1.master page.

**Replacing placeholder content**



After creating [Content](https://msdn.microsoft.com/en-us/library/h7896hef(v=vs.100)) controls, you add text and controls to them. In a content page, anything that is not inside the [Content](https://msdn.microsoft.com/en-us/library/h7896hef(v=vs.100)) controls (except script blocks for server code) results in an error. You can perform any tasks in a content page that you do in an ASP.NET page. For example, you can generate content for a [Content](https://msdn.microsoft.com/en-us/library/h7896hef(v=vs.100)) control using server controls and database queries or other dynamic mechanisms.

<% @ Page Language="C#" MasterPageFile="~/Master.master" Title="Content Page 1" %>

<asp:Content ID="Content1" ContentPlaceHolderID="Main" Runat="Server">

Main content.

</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderID="Footer" Runat="Server" >

Footer content.

</asp:content>

The @ Page directive binds the content page to a specific master page, and it defines a title for the page that will be merged into the master page. Note that the content page contains no other markup outside of the [Content](https://msdn.microsoft.com/en-us/library/h7896hef(v=vs.100)) controls. (The master page must contain a head element with the attribute runat="server" so that the title setting can be merged at run time.)

You can create multiple master pages to define different layouts for different parts of your site, and a different set of content pages for each master page.

## Advantages of Master Pages

Master pages provide functionality that developers have traditionally created by copying existing code, text, and control elements repeatedly; using framesets; using include files for common elements; using ASP.NET user controls; and so on. Advantages of master pages include the following:

* They allow you to centralize the common functionality of your pages so that you can make updates in just one place.
* They make it easy to create one set of controls and code and apply the results to a set of pages. For example, you can use controls on the master page to create a menu that applies to all pages.
* They give you fine-grained control over the layout of the final page by allowing you to control how the placeholder controls are rendered.
* They provide an object model that allows you to customize the master page from individual content pages.