**1. CSS and types of Style sheets**

CSS Stands for "Cascading Style Sheet." Cascading style sheets are used to format the layout of [Web pages](https://techterms.com/definition/webpage). They can be used to define text styles, table sizes, and other aspects of Web pages that previously could only be defined in a page's [HTML](https://techterms.com/definition/html).

CSS helps Web developers create a uniform look across several pages of a Web site. Instead of defining the style of each table and each block of text within a page's HTML, commonly used styles need to be defined only once in a CSS document. Once the style is defined in cascading style sheet, it can be used by any page that references the CSS file. Plus, CSS makes it easy to change styles across several pages at once. For example, a Web developer may want to increase the default text size from 10pt to 12pt for fifty pages of a Web site. If the pages all reference the same style sheet, the text size only needs to be changed on the style sheet and all the pages will show the larger text.

To add CSS styles to your website, you can use three different ways to insert the CSS. You can Use an "ExternalStylesheet", an "Internal Stylesheet", or in "Inline Style". The benefit for using each depends on what you are doing with the Style.The following table explains the difference between them.

|  |  |
| --- | --- |
| **Different CSS style linking** | |
| **Internal Stylesheet** | An internal stylesheet holds the CSS code for the webpage in the head section of the particular file. This makes it easy to apply styles like classes or id's in order to reuse the code. The downside of using an internal stylesheet is that changes to the internal stylesheet only effect the page the code is inserted into. |
| **External Stylesheet** | The External Stylesheet is a .css file that you link your website to. This makes it so that what ever you change in the .css sheet, will effect every page in your website. This prevents you from having to make many code changes in each page. This is for "global" site changes. |
| **Inline Styles** | The Inline style is specific to the tag itself. The inline style uses the HTML "style" attribute to style a specific tag. This is not recommended, as every CSS change has to be made in every tag that has the inline style applied to it. The Inline style is good for one an individual CSS change that you do not use repeatedly through the site. |

Below are examples of what the code looks like for each type of CSS.

**1. *Internal CSS Stylesheet***

When creating a stylesheet internally in the web page, you will need to use the HTML tags in the Head section of your webpage. All the code for the Internal CSS stylesheet is contained between the section of your websites code. Below is an example of what an Internal stylesheet looks like.

<head>

<style type="text/css">

h1 {color:blue;}

h2 {color:red;}

p {color:green;}

</style>

</head>

**2. *External CSS Stylesheet***

When using an external stylesheet you must reference the stylesheet in the HTML page that is using it. You would add the code below to your HTML document to reference a stylesheet in the same location as the HTML page called "style.css". You can upload the "style.css" page can be located anywhere in your files. You can name your stylesheet whatever you like and link to as many as you like. You can simply link to it in your head section and every edit your make to the "style.css" sheet will be globally changed through out the site. Below is what the code looks like.

<head><br />

<link rel="stylesheet" type="text/css" href="/support/style.css" /><br />

</head>

**3. *Inline CSS Styles***

The inline style uses the HTML "style" attribute. This allows CSS properties on a "per tag" basis. The following is an example of how the inline style is used.

This is a paragraph!

This inline style will change the color of the paragraph to red and make the font size 18 pixels.

**2. Master Page**

A master page is an ASP.NET file with the extension .master (for example, MySite.master) with a predefined layout that can include static text, HTML elements, and server controls. The master page is identified by a special [@ Master](https://msdn.microsoft.com/en-us/library/ms228176.aspx) directive that replaces the [@ Page](https://msdn.microsoft.com/en-us/library/ydy4x04a.aspx) directive that is used for ordinary .aspx pages. The directive looks like the following:

%@ Master Language="C#" %>

In addition to the **@ Master** directive, the master page also contains all of the top-level HTML elements for a page, such as **html**, **head**, and **form**. For example, on a master page you might use an HTML table for the layout, an **img** element for your company logo, static text for the copyright notice, and server controls to create standard navigation for your site. You can use any HTML and any ASP.NET elements as part of your master 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/system.web.ui.webcontrols.contentplaceholder.aspx) controls. These placeholder controls define regions where replaceable content will appear. In turn, the replaceable content is defined in content pages. After you have defined the [ContentPlaceHolder](https://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.contentplaceholder.aspx) controls, a master page might look like the following:

<%@ Master Language="C#" %>

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

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

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

<head runat="server" >

<title>Master page title</title>

</head>

<body>

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

<table>

<tr>

<td><asp:contentplaceholder id="Main" runat="server" /></td>

<td><asp:contentplaceholder id="Footer" runat="server" /></td>

</tr>

</table>

</form>

</body>

</html>

## 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.