**Style Sheet Language – Cascading Style Sheets**

Having learned how to structure the content of your documents using HTML’ s wide variety of

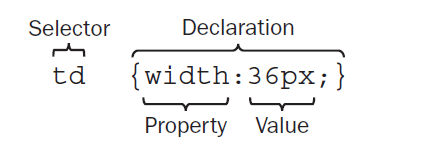
elements and attributes, you’ re now going to start making your pages look a lot more exciting.

You’ re going to learn how to use *cascading style sheets* (or CSS for short) to take control of the style of your pages, including the colors and size of fonts, the width and colors of lines, and the amount of space between items on the page. The cascading style sheets specification works by allowing you to specify *rules* that say how the content of elements within your document should appear. For example, you can specify that the background of the page is a cream color, the contents of all <p> elements should be displayed in gray using the Arial typeface, and that all <h1> elements should be in red using the Times New Roman typeface.

**Introducing CSS:**

CSS works by allowing you to associate *rules* with the elements that appear in a web page. These rules govern how the content of those elements should be rendered. Following shows you an example of a CSS rule, which is made up of two parts:

* The *selector*, which indicates which element or elements the declaration applies to (if it applies to more than one element, you can have a comma - separated list of several elements)
* The *declaration*, which sets out how the elements referred to in the selector should be styled.



The rule in above figure applies to all <td> elements and indicates that they should be 36 pixels wide.

The declaration is also split into two parts, separated by a colon:

* A *property*, which is the property of the selected element(s) that you want to affect, in this case the width property.
* A *value*, which is a specification for this property; in this case it is that the table cells should be 36 pixels wide.

This is very similar to the way that HTML/HTML elements can carry attributes and how the attribute controls a property of the element; the attributes’ value would be the setting for that property. For example, a <td> element could have a width attribute whose value is the width you want the table to be:

**<td width=”36”> </td>**

With CSS, however, rather than you’re having to specify the attribute on each instance of the <td> element, the selector indicates that this one rule applies to all < td > elements in the document.

Here is an example of a CSS rule that applies to several different elements (in this example, the <h1>, < h2 >, and <h3> elements). A comma separates the name of each element that this rule will apply to. The rule also specifies several properties for these elements with each property - value pair separated by a semicolon. Note how all the properties are kept inside the curly braces:

h1, h2, h3 {

font-weight : bold;

font-family : arial;

color : #000000;

background-color : #FFFFFF;

}

Even if you have never seen a CSS rule before, you should now have a good idea of what this rule is doing. There are three heading elements named in the selector (<h1>, <h2>, and <h3>), and this rule says that where these headings are used, they will be written in a bold Arial font in black with a white background.

*If there is only one property - value pair in the declaration, you do not need to end it with a semicolon. However, because a declaration can consist of several property - value pairs, and each property - value pair within a rule must be separated by a semicolon, it is good practice to start adding semicolons every time you write a rule in case you want to add another rule later. If you forget to add the semicolon, any further property - value pairs will be ignored.*

**A Basic Example:**

Let ’s have a look at an example that shows how a set of CSS rules can transform the look of an HTML page. CSS rules can live inside the HTML document, although for this example we will be making a separate file to hold the CSS rules, and the HTML page will contain a link to this file, which is known as a *style sheet*.

Example/Source Code:

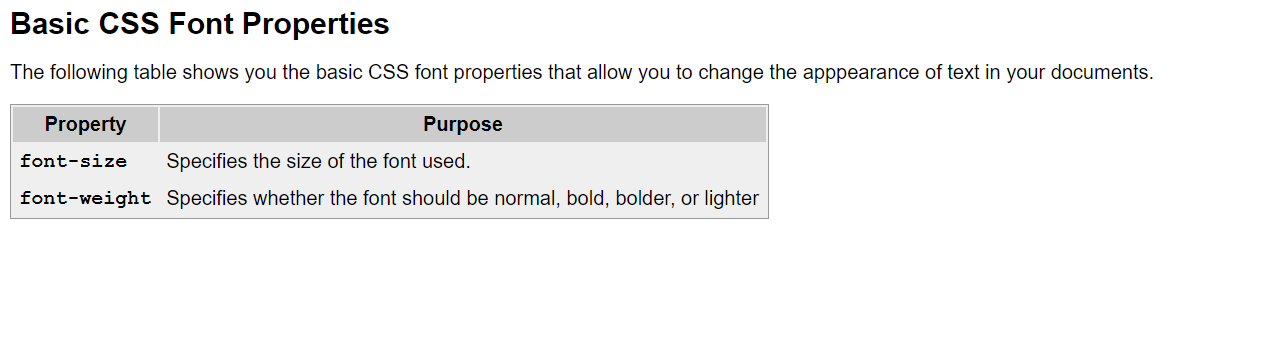
(HTML file)



(CSS File)



Output:



**Inheritance:**

One of the powerful features of CSS is that, when a property has been applied to one element, it will often be *inherited* by child elements (elements contained within the element that the rules were declared upon). For example, once the font - family property had been declared for the <body> element in the previous example, it applied to all of the elements inside the <body> element. This saves you from having to repeat the same rules for every single element that makes up a web page.

If another rule is more specific about which elements it applies to, then it will override any properties associated with the <body> element or any other containing element. In the preceding example, most of the text was in an Arial typeface, as specified in the rule associated with the <body> element, although there were a few table cells that used a Courier typeface. The table cells that were different had a class attribute whose value was code:

**<td class = ”code”> font-size </td>**

Here you can see the rule associated with these elements:

td.code{

font-family: courier, courier-new, serif;

font-weight: bold;

}

This rule takes precedence over the one associated with the <body> element because the selector is more specific about which element it applies to. The way in which some properties inherit saves you from having to write out rules and all the property - value pairs for each element and makes for a more compact style sheet.

**Where you can add CSS rules: (Types of CSS)**

The example that you saw at the beginning of the chapter placed the CSS rules in a separate file known as an ***external style sheet***. CSS rules can also appear in two places inside the HTML document:

* Inside a <style> element, which sits inside the <head> element of a document
* As a value of a style attribute on any element that can carry the style attribute

When the style sheet rules are held inside a <style> element in the head of the document, they are referred to as an ***internal style sheet****.*

Example:

<head>

<title> Internal Style sheet </title>

<style type=”text/css”>

body {

color: #000000;

background-color: #ffffff;

font-family: arial, verdana, sans-serif;

}

h1 {

font-size: 18pt;

}

p {

font-size: 12pt;

}

</style>

</head>

When style attributes are used on HTML elements, they are known as ***inline style rules***. For example:

<td style = ”font-family:courier; padding:5px; border-style:solid; border-width:1px; border-color:#000000;”>

Here you can see that the properties are added as the value of the style attribute. You still need to separate each property from its value with a colon and each of the property - value pairs from each other with a semicolon. However, there is no need for a selector here (because the style is automatically applied to the element that carries the style attribute), and there are no curly braces.

**The <link> Element:**

The <link> element is used in web pages to describe the relationship between two documents; for example, it can be used in an HTML page to specify a style sheet that should be used to style a page. You may also see the <link> element used in HTML pages for other purposes, for example to specify an RSS feed that corresponds with a page.

It is a very different kind of link than the <a> element because the two documents are automatically associated — the user does not have to click anything to activate the link.

The <link> element is always an **empty element**, and when used with style sheets it must carry three attributes:

type, rel, and href. Here is an example of the < link > element used in an HTML

page indicating that it should be styled by a CSS file called interface.css, which lives in a

subdirectory called CSS:

< link rel=”stylesheet” type=”text/css” href=”../CSS/interface.css” / >

In addition to the core attributes, the <link> element can also take the following attributes:

* charset
* dir
* href
* hreflang
* media
* rel
* rev
* style
* target
* type

You have met many of these already, so the more important ones are discussed in the following sections along with some of the less common ones.

**The rel Attribute:**

The rel attribute is required and specifies the relationship between the document containing the link and the document being linked to. The key value for working with style sheets is stylesheet.

rel=”stylesheet”

**The type Attribute:**

The type attribute specifies the MIME type of the document being linked to; in this case, we are dealing with a CSS style sheet, so the MIME type is text/css :

type=”text/css”

**The href Attribute:**

The href attribute specifies the URL for the document being linked to.

href=”../stylesheets/interface.css”

The value of this attribute can be an absolute or relative URL, but it is usually a relative URL because the style sheet is part of the site.

**The hreflang Attribute:**

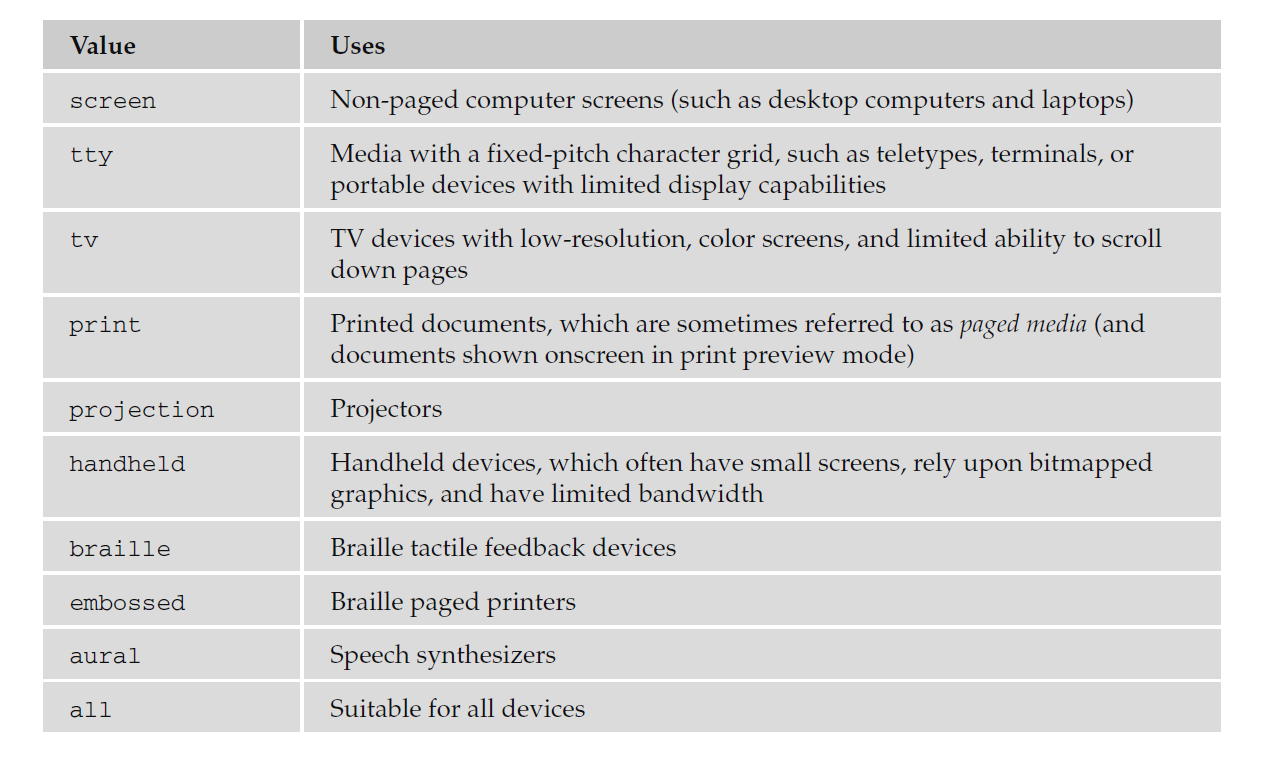
The hreflang attribute specifies the language that the resource specified is written in.

hreflang=”en-US”

**The media Attribute:**

The media attribute specifies the output device that is intended for use with the document:

media=”screen”

While this attribute is not always used, it is becoming increasingly important as people access the Internet in different ways using different devices. See the following table for the possible values.

**The <style> Element:**

The <style> element is used inside the <head> element to contain style sheet rules within a web page, rather than linking to an external document. It is also sometimes used when a single page needs to contain just a few extra rules that do not apply to the other pages of the site which all share the same style sheet.

For example, here is a style sheet attached to the HTML document using the <link> element you just learned about, as well as a <style> element containing an additional rule for <h1> elements:



When you use the <style> element it should always carry the type attribute. Here is a full list of the attributes it can carry:

* dir
* lang
* media
* title
* type

**Advantages of External CSS Style Sheets:**

If two or more documents are going to use a style sheet, you should use an external style sheet. There are several reasons for this, including:

* It saves you repeating the same style rules in each page.
* You can change the appearance of several pages by altering just the style sheet rather than each individual page. This means it is easier to update your site if you want to, for example, change the style of font used in all headings or alter the color of all links.
* Once a visitor to your site has downloaded the CSS style with the first page of your site that uses it, subsequent pages will be quicker to load (because the browser retains a copy of the CSS style sheet and the rules do not have to be downloaded for every page). This also puts less strain on the server (the computer that sends the web pages to the people viewing the site) because the pages it sends out are smaller.
* The style sheet can act as a style template to help different authors achieve the same style of document without learning all of the individual style settings.
* Because the web pages do not contain the style rules, different style sheets can be attached to the same document. So, you can use the same HTML document with one style sheet when the viewer is on a desktop computer, another style sheet when the user has a handheld device, another style sheet when the page is being printed, another style sheet when the page is being viewed on a TV, and so on. You can reuse the same document with different style sheets for different visitors’ needs.
* A style sheet can import and use styles from other style sheets, allowing for modular development and good reuse. (For example, I have a style sheet that I import into other style sheets whenever I want to include examples of programming code in a web page — I do not need to write the style rules again and again.)
* If you remove the style sheet, you can make the site more accessible for those with visual impairments, because you are no longer controlling the fonts and color schemes.

**CSS Properties:**

You now know that styling a web page using CSS involves creating rules, and that these rules contain two parts: firstly, a selector to indicate which elements the rule applies to, and secondly, one or more properties which control the presentation of these elements.

So, if there is a part of the page that you want to make a certain color or size, then you need to find the corresponding property to control those elements.

The properties are grouped together into related functionality; for example, there are properties that allow you to control the presentation of tables, lists, and backgrounds. The following table shows the main properties available to you:

***<scroll more>***

***<scroll more>***

***<scroll more>***

