CSS How To...

❮ Previous Next ❯

When a browser reads a style sheet, it will format the HTML document according to the information in the style sheet.

Three Ways to Insert CSS

There are three ways of inserting a style sheet:

- External style sheet
- Internal style sheet
- Inline style

External Style Sheet

With an external style sheet, you can change the look of an entire website by changing just one file!

Each page must include a reference to the external style sheet file inside the <link> element. The <link> element goes inside the <head> section:

Example

```
<head>
<link rel="stylesheet"type="text/css"href="mystyle.css">
</head>
```

Try it Yourself »

An external style sheet can be written in any text editor. The file should not contain any html tags. The style sheet file must be saved with a .css extension.

Here is how the "myStyle.css" looks:

```
body {
    background-color:lightblue;
}
```

```
h1 {
    color: navy;
    margin-left: 20px;
}
```

Note: Do not add a space between the property value and the unit (such as `margin-left: 20 px;`). The correct way is: `margin-left: 20px;`

Internal Style Sheet

An internal style sheet may be used if one single page has a unique style.

Internal styles are defined within the <style> element, inside the <head> section of an HTML page:

Example

```
<head>
<style>
body {
    background-color:linen;
}

h1 {
    color: maroon;
    margin-left: 40px;
}
</style>
</head>
```

Try it Yourself »

Inline Styles

An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

The example below shows how to change the color and the left margin of a <h1> element:

Example

```
<h1style="color:blue;margin-left:30px;">This is a heading.
</h1>
```

Try it Yourself »

Tip: An inline style loses many of the advantages of a style sheet (by mixing content with presentation). Use this method sparingly

---

Multiple Style Sheets

If some properties have been defined for the same selector (element) in different style sheets, the value from the last read style sheet will be used.

Example

Assume that an external style sheet has the following style for the <h1> element:

```
h1 {
    color: navy;
}
```

---

then, assume that an internal style sheet also has the following style for the <h1> element:

```
h1 {
    color: orange;
}
```

If the internal style is defined after the link to the external style sheet, the <h1> elements will be "orange":

Example

```
<head>
<link rel="stylesheet"type="text/css"href="mystyle.css">
<style>
h1 {
    color: orange;
}
</style>
</head>
```

Try it Yourself »

However, if the internal style is defined before the link to the external style sheet, the <h1> elements will be "navy":

Example

```
<head>
<style>
h1 {
    color: orange;
}
</style>
<link rel="stylesheet"type="text/css"href="mystyle.css">
</head>
```

Try it Yourself »

Cascading Order

What style will be used when there is more than one style specified for an HTML element?

Generally speaking we can say that all the styles will "cascade" into a new "virtual" style sheet by the following rules, where number one has the highest priority:

1. Inline style (inside an HTML element)
2. External and internal style sheets (in the head section)
3. Browser default

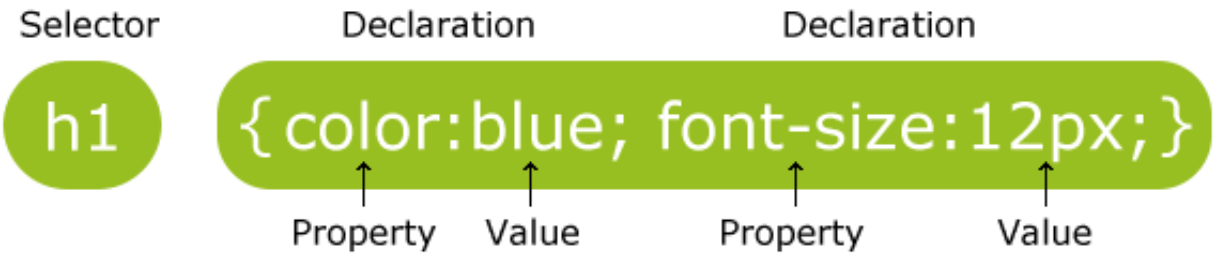
So, an inline style (inside a specific HTML element) has the highest priority, which means that it will override a style defined inside the <head> tag, or in an external style sheet, or a browser default

CSS Syntax and Selectors

❮ Previous Next ❯

CSS Syntax

A CSS rule-set consists of a selector and a declaration block:



The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.

In the following example all <p> elements will be center-aligned, with a red text color:

Example

```
p {
    color: red;
    text-align: center;
}
```

Try it Yourself »

CSS Selectors

CSS selectors are used to "find" (or select) HTML elements based on their element name, id, class, attribute, and more.

The element Selector

The element selector selects elements based on the element name.

You can select all <p> elements on a page like this (in this case, all <p> elements will be center-aligned, with a red text color):

Example

```
p {
    text-align: center;
    color: red;
}
```

Try it Yourself »

The id Selector

The id selector uses the id attribute of an HTML element to select a specific element.

The id of an element should be unique within a page, so the id selector is used to select one unique element!

To select an element with a specific id, write a hash (#) character, followed by the id of the element.

The style rule below will be applied to the HTML element with id="para1":

Example

```
#para1 {
    text-align: center;
    color: red;
}
```

Try it Yourself »

Note: An id name cannot start with a number!

The class Selector

The class selector selects elements with a specific class attribute.

To select elements with a specific class, write a period (.) character, followed by the name of the class.

In the example below, all HTML elements with class="center" will be red and center-aligned:

Example

```
.center {
    text-align: center;
    color: red;
}
```

Try it Yourself »

You can also specify that only specific HTML elements should be affected by a class.

In the example below, only <p> elements with class="center" will be center-aligned:

Example

```
p.center {
    text-align: center;
    color: red;
}
```

Try it Yourself »

HTML elements can also refer to more than one class.

In the example below, the <p> element will be styled according to class="center" and to class="large":

Example

```
<p class="center large">This paragraph refers to two classes.
</p>
```

Try it Yourself »

> **Note:** A class name cannot start with a number!

Grouping Selectors

If you have elements with the same style definitions, like this:

```
h1 {
    text-align: center;
    color: red;
}

h2 {
    text-align: center;
    color: red;
}

p {
    text-align: center;
    color: red;
}
```

It will be better to group the selectors, to minimize the code.

To group selectors, separate each selector with a comma.

In the example below we have grouped the selectors from the code above:

Example

```
h1, h2, p {
    text-align: center;
    color: red;
}
```

Try it Yourself »

CSS Comments

Comments are used to explain the code, and may help when you edit the source code at a later date.

Comments are ignored by browsers.

A CSS comment starts with /* and ends with */. Comments can also span multiple lines:

Example

```
p {
    color: red;
    /* This is a single-line comment */
    text-align: center;
}

/* This is
a multi-line
comment */
```

JSF pages can be styled using cascading stylesheets (CSS), which define rules for styling page elements.

This involves both:

- **Planning** — this can be done without knowing the syntax of CSS files or how they are incorporated into JSF pages
- **Stylesheet Creation and Use** — this involves placing them into resource folders and accessing them from JSF pages.

Style Planning Cascading Style Sheets
Style planning involves determining the visual characteristics of your application's web pages, including:

- Font style, size, and weight
- Foreground and background color
- Borders, etc

Once you know the desired contents of a JSF page you first need to identify and name the *types* of components that need styles.

For example, error messages are often rendered in styles that are different than the rest of the components on a page so they are easily recognized.

Style types should be named according to how they are *used*, not how they *appear*.

For example, a style type name for error messages should be something like **error** rather than **red-font**, even if the error message is in a red font, because an error message's appearance is subject to change.

Once identified, style type names can then be used as values for **class** attributes of plain HTML elements or **styleClass** attributes of JSF HTML elements.
Cascading styles sheet provide a mechanism for separating data in web pages from its presentation, similar to the separation provided by the **Model-View-Controller** design paradigm:

- Web designers can control the presentation of numerous data items with minimal effort
- The same data can be presented in several different visual forms

A cascading style sheet consists of a sequence of **CSS rules**, each defining a set of style attributes that can be applied to elements in a JSF page.

These rules are gathered together into files that can be used in several JSF pages.

The stylesheet file names use a **.css** suffix.
CSS Rules Style Attributes Applying Styles References

A CSS rule has the following form:

***selector* {**
sequence of *declaration*s
}

- The ***selector*** part identifies where the rule applies. The simplest and most useful form of a ***selector*** has the form **`.class-name`**.
- The ***class-name*** is then used as the value for the **`class`** attribute in plain HTML tags or the

styleClass attribute in JSF HTML tags.

Each ***declaration*** has the following form:

***keyword*: *value*;**

- Each ***keyword-value*** pair specifies a style attribute to be applied at places specified by the ***selector***.
- They can specify a variety of attributes such as colors, fonts, sizes, and spacing.

For example, here is a rule that styles the text used to prompt a user for input:

```
.prompt {
   font-size: larger;
   color: forestgreen;
}
```

Here is a partial list of style keywords and legal values for style attributes. This just scratches the surface of what is available in CSS.

Keyword	Values
color	red, orange, yellow, green, blue, gray, black, white, ...
background-color	red, orange, yellow, green, blue, gray, black, white, ...
font-family	monospace, serif, sans-serif, cursive, Times, Helvetica, ...
font-size	small, medium, large, 14px, ...
font-weight	normal, bold, lighter, ...
font-style	normal, italic, oblique
border	thin solid blue, thick dotted green, 1em dashed yellow ...

Page components are styled by adding the following attributes to tags that can be styled.

- For unprefixed (HTML) tags: **class="*class-name*"**. Example:

```
<h1 class="header"> ... </h1>
```

- For **h:** prefixed tags: **styleClass="*class-name*"**. Example:

```
<h:outputText styleClass="loginError" ... />
```

- Some **h:** prefixed tags have multiples styles that depend on the values of the tag attributes. For these tags there are multiple style attributes whose names depend on the tag. Example:

```
<h:message errorClass="error" warnClass="warn" ... />
```

The ***class-name***s should be names used in stylesheet rule selectors.

- Eric A. Meyer, "Cascading Style Sheets: The Definitive Guide", O'Reilly, 2004.
- CSS 2.1 Specification (W3C)
- CSS Tutorial (www.w3schools.com)