WIKIBOOKS

Cascading Style Sheets/Selectors

In CSS, a **selector** is the part of a rule that comes before the first "{", such as "p" in the rule " `p { font-weight:bold; }` ". A selector specifies to which elements a rule should apply, by naming the type of the element, such as "div", the class of the element, or the id of the element. Selectors can only be used in linked and embedded CSS, not in inlined one.

The following table provides an overview of selectors. The sections that follow discuss each type of selector in detail.

Overview of selectors

Type of Selector	Example Selector	Example Rule
type	div	div { margin:7px; padding:7px; }
class	.important	.important { font-weight:bold; color:red; }
id	#onlyThisOne	#onlyThisOne { font-family:courier; }
universal	*	* { color:green; }
pseudo-class	a:link	a:link { color:blue; }
pseudo-element	p:first-letter	p:first-letter { color:red }
descendant	td span	td span { font-weight:bold; }
grouped	h1, h2, h3	h1, h2, h3 { text-align:center; }

Contents

Type
Class
 Internet Explorer 6 bug
ID
Universal
Pseudo-Classes
Pseudo-Elements
Simple selectors
Descendant
Grouping selectors
Later CSS versions

Type

These selectors match elements based on the name of their element type. The example above is using a type selector to make every instance of `p` have bold text. You can use the type selector with any element. Here are a few examples:

```
div{
  margin:7px;
  padding:7px;
}
```

```
body{
  background-image:url("image.gif");
  font-size:.9em;
}
```

Class

The class selector works with (X)HTML documents. It does not work with general XML documents unless the User Agent (web browser or other document reader) can determine which attribute is the class attribute for elements in the document. In (X)HTML documents `class` is defined to be the class attribute.

HTML permits the `class` attribute for all elements that are valid in the body section of an HTML document, including the `body` element itself. This allows the web designer to distinguish between elements of the same type used in different contexts. For example, the designer could differentiate between HTML elements and HTML attributes in a technical document on HTML.

```
The <code class="attribute">alt</code> attribute of the <code class="element">img</code>
```

The class selector allows you to apply CSS rules based on the class of an element.

The first way is to make a global class, which can be added to any element. You do that like this:

```
.important {
  font-weight:bold; color:red;
}
```

That will make any element that has the class of **important** be bold and red.

Sample HTML:

```
<p class="important">Important Text</p>
<p>Ordinary Text</p>
<div class="important">Important Footnote</div>
```

Example rendering:

Important Text

Ordinary Text

Important Footnote

The second way is to attach it to a type selector. The rule is only applied to elements of the given type which are of the specified class.

CSS rule:

```
p.right {
  float:right;
  clear:both
}
```

Sample HTML:

```
<p class="right">Righty Righty</p>
```

Example rendering (look right):

Righty Righty

An element may belong to more than one class, e.g.

```
<p class="right">This paragraph belongs to the class 'right'.</p>
<p class="important">This paragraph belongs to the class 'important'.</p>
<p class="right important">This paragraph belongs to both classes.</p>
```

Example rendering:

This paragraph belongs to the class 'important'. This paragraph belongs to the class 'right'.

This paragraph belongs to both classes.

Class names should describe the purpose of the class, e.g. important, not the effect of the class, e.g. red. If you name classes by effect and then change your mind about the appearance you want you can end up with rules like:

```
 .red {color:blue}
```

If necessary multiple class selectors can be used to select only elements that belong to all the specified classes, e.g.

```
p.important.right {
  border: 2px dashed #666
}
```

Example rendering:

This paragraph belongs to the class 'important'.

This paragraph belongs to the class 'right'.

This paragraph belongs to both classes.

Internet Explorer 6 bug

Multiple class selectors as shown in the previous example do not work in Internet Explorer version 6. It treats the selector as though only the last class selector was present, e.g. `p.important.right` is treated as equivalent to `p.right`.

Previous example rendered in Internet Explorer 6:

This paragraph belongs to the class 'important'.

This paragraph belongs to the class 'right'.

This paragraph belongs to both classes.

The complete HTML document and CSS stylesheet to test this bug are given below.

classBug.css:

```
.important {
  font-weight:bold; color:red;
}

p.right {
  float:right;
  clear:both
}

p.important.right {
  border: 2px dashed #666
}
```

classBug.htm:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html lang="en">
 <head>
    <title>IE6 multiple class bug</title>
    <link rel="stylesheet" type="text/css" href="classBug.css">
  </head>
  <body>
    <p class="right">This paragraph belongs to the class 'right'.</p>
    <p class="important">This paragraph belongs to the class 'important'.</p>
    <p class="right important">This paragraph belongs to both classes.</p>
  </body>
</html>
```

ID

The ID selector works with (X)HTML documents. It does not work with general XML documents unless the User Agent (web browser or other document reader) can determine which attribute is the ID attribute for elements in the document. In (X)HTML documents `id` is defined to be the ID attribute.

The ID selector is used to select a single item on a page. It matches the `id` attribute of an element. Two elements on the same page must not have the same `id` attribute.

However, several different webpages within the same site might reuse an `id`. It is commonly used for the major divisions of a page, e.g. mainContent, navigationBar. It is often used in conjunction with descendant selectors, e.g. to style all list items in the navigation bar whilst not affecting lists in the main content.

CSS rule:

```
 #onlyThisOne {
   font-family:courier;
 }
```

Sample HTML:

```
 <p id="onlyThisOne">Courier</p>
```

Example rendering:

```
Courier
```

Universal

These selectors apply a style to all the items on the page. For example to make all the text on the page green use the rule:

```
 * {
   color:green;
```

```
}
```

Pseudo-Classes

Pseudo-classes and pseudo-elements allow elements to be formatted on the basis of information other than the position of the element in the document tree. Pseudo-class and pseudo-element selectors are prefixed by a colon, `:`, in CSS1 and CSS2.1. In CSS3 pseudo-elements are prefixed by `::`.

CSS level 1 defines three pseudo-classes:

`link`
: unvisited links

`visited`
: visited links

`active`
: active links

These can only be applied to the anchor (`a`) elements.

```
a:link{
  color:blue;
}

a:visited{
  color:purple;
}

a:active{
  color:red;
}
```

CSS level 2.1 introduces several additional pseudo-classes, including `hover`. `hover` can be used to create rollover effects without resorting to Javascript. CSS2.1 also allows `active` to apply to any element that can be active, e.g. a button.

The additional CSS 2.1 pseudo-classes are:

`first-child`
: matches any element that is the first child of its parent.

`lang(...)`
: matches the language of the element. The language may be set directly on the element or inherited from an ancestor element. A valid language code must appear in the parentheses, e.g. `en` for any variant of English or `en-gb` for British English.

`hover`
: applies whilst the user hovers over the element with the mouse.

`active`
: now allowed on any element capable of being 'active' – applies whilst the element is active.

`focus`
: applies whilst the element has the keyboard focus.

Although CSS2.1 allows `hover` on any element Internet Explorer only allows it on anchor elements.

Examples:

```
 p:first-child {
   font-size:120%
 }
```

```
  span:lang(la) { /* render Latin text, e.g. per se, in italics */
    font-style:italic
  }

  li:hover { /* doesn't work in Internet Explorer */
    background-color:blue
  }

  input:active {
    color:red
  }

  input:focus {
    background-color:yellow
  }
```

An example of the `first-child` pseudo-class is given at the end of the descendant selector section.

Multiple pseudo-classes may be specified, e.g. to warn the user that they are hovering over a link they have already visited:

```
a:visited:hover {
  background-color:red
}
```

Care should be taken over the order of rules using pseudo-classes. For example, if you want visited links to be green except whilst the user hovers over them, when they should be yellow, the rules must be in the following order:

```
 a:visited{
   color:green
 }

 a:hover{
   color:yellow
 }
```

If the rules were reversed the cascade would mean that the visited color would take precedence over the hover color. Visited links would remain green even when the user hovered over them.

Pseudo-Elements

Pseudo-classes and pseudo-elements allow elements to be formatted on the basis of information other than the position of the element in the document tree. Pseudo-class and pseudo-element selectors are prefixed by a colon, `:`, in CSS1 and CSS2.1. In CSS3 pseudo-elements are prefixed by `::`.

CSS level 1 defines two pseudo-elements, `first-letter` and `first-line`, which select the first letter and line of the rendered element respectively. These can be used to apply typographic effects, e.g. to create a drop cap at the start of a paragraph.

```
 p:first-letter { color:red }
```

Only one pseudo-element selector may be used and it must be the last part of the selector chain. The `first-line` selector may only be applied to block-level elements, inline-blocks, table captions and table cells.

CSS2.1 adds two more pseudo-elements, `before` and `after`.

Starting in CSS3 pseudo-elements are prefixed with `::` rather than `:`.

Simple selectors

The selectors described above (type, class, ID, universal, pseudo-class and pseudo-element) are all examples of simple selectors. The full syntax for a simple selector is:

- a type selector or the universal selector
- zero, one or more class, ID and pseudo-class selectors (CSS2.1 also allows attribute selectors)
- zero or one pseudo-element selectors

The following are all examples of simple selectors:

```
p
p.important
p#navigation
a:link
p:first-line
a:visited#homePage.menu2:first-letter
*
*#footer
*.content.abstract
*#mainArticle:first-letter
```

As a shorthand notation the universal selector, `*`, may be omitted if it is not the only component of the simple selector. For example, `#footer` is equivalent to `*#footer` and `:first-line` is equivalent to `*:first-line`.

Combining simple selectors: Simple selectors can be combined to take the context of the element in to account. For example you might want to apply a rule only to list elements in the navigation bar.

Descendant

Descendant selectors allow you to apply style to elements that are nested within another specified element. For example, you could have every `span` element inside of every `p` element be bold. The `span` can be a child element of the `p` element, or a grandchild, or a great-grandchild, etc.

CSS rule:

```
p span{
  font-weight:bold;
}
```

Sample HTML:

```
<p>Start of paragraph. <span>This span is bold.</span> Rest of paragraph.</p>
<div>Start of division. <span>This span is normal.</span> Rest of division.</div>
```

Example rendering:

Start of paragraph. **This span is bold.** Rest of paragraph.

Start of division. This span is normal. Rest of division.

The next example changes the color of visited links in the navigation section. Visited links in other parts of the document would be unaffected.

CSS rule:

```
ul#navigation a:visited {
  color:red
}
```

Sample HTML:

```
<ul id="navigation">
  <li><a href="HomePage">Home</a></li>
  <li><a href="AnotherPage">A page you haven't visited yet.</a></li>
</ul>
```

Example rendering:

- Home
- A page you haven't visited yet.

An example using the `first-child` selector.

CSS rule:

```
div.content strong:first-child {
  color:red
}
```

Sample HTML:

```
<div class="content">
  <p>Some <em>emphasized</em> text and some <strong>strongly emphasized</strong> text.</p>
  <p>Some <strong>strongly emphasized</strong> text and some <em>emphasized</em> text.</p>
</div>
```

Example rendering:

Some *emphasized* text and some **strongly emphasized** text.

Some **strongly emphasized** text and some *emphasized* text.

Two important points to note:

- an element is still the first child if it is preceded by text, so the `em` element in the first paragraph and the `strong` element in the second paragraph are the first children of their respective paragraphs.

- a rule is only applied to an element if it matches all parts of the selector. The `strong` element in the first paragraph is a second child, so it is not matched by the `strong:first-child` selector.

Grouping selectors

Selectors can be grouped into comma separated lists.

```
h1, h2, h3 { text-align:center; }
```

is equivalent to

```
h1 { text-align:center; }
h2 { text-align:center; }
h3 { text-align:center; }
```

An element may be selected by more than one rule:

```
h1, h2, h3, h4, h5, h6 { color: white; background-color: red; }
h1 { color: yellow; }
```

All headings will have a red background. Level 2–6 headings will have white text. Level 1 headings will have yellow text.

The order of the rules is significant. In the following example all headings would have white text.

```
h1 { color: yellow; }
h1, h2, h3, h4, h5, h6 { color: white; background-color: red; }
```

When more than one rule applies to an element the cascade is used to determine the resulting style applied to the element.

Later CSS versions

Additional selectors in CSS 2.1:

- Child selectors
- Adjacent sibling selectors
- Attribute selectors
- More pseudo-classes and pseudo-elements.

Additional selectors in CSS 3:

- More sibling selectors
- More attribute selectors
- More pseudo-classes and pseudo-elements.