
CSS Essentials

Getting started with Cascading Style Sheets



Topics

- HTML Formatting – why not?
- CSS Essentials
 - Rules, Selectors and Declarations
 - Adding style to documents
- CSS Cascade
 - Selectors & specificity
- Units & Colours

Evolution of HTML Formatting

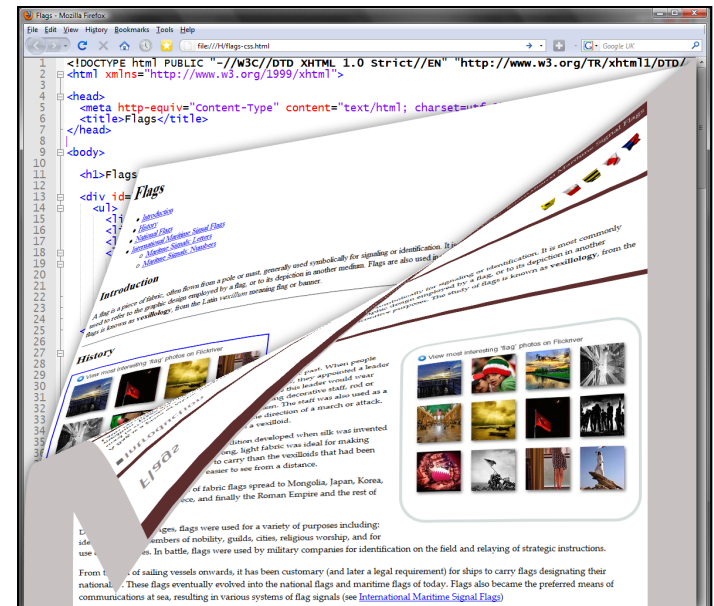
- (X)HTML only for structuring content
 - Specification only contains *guidelines* for visual browsers
- Some tags/attributes added for visual formatting

```
<font face="Arial" color="red">Hello</font> → Hello
```

- This mixes style and structure
 - Often using proprietary mark up with limitations on what can be applied

The Solution: CSS

- **Cascading Style Sheets**
 - Separation of style from structure
 - Control – potentially over every item in the page
 - Easier style management
- *Strict XHTML (and HTML 4.01) deprecated HTML formatting in favour of CSS*



Same content... different view

<http://www.csszengarden.com>

The screenshot shows the original CSS Zen Garden website. It features a traditional Japanese aesthetic with a torii gate in the background. The text is centered and uses a serif font. The layout is clean and minimalist, typical of early CSS-based design.

This collage displays several different redesigns of the CSS Zen Garden website. Each design uses a different visual theme and layout to present the same core content. Examples include:

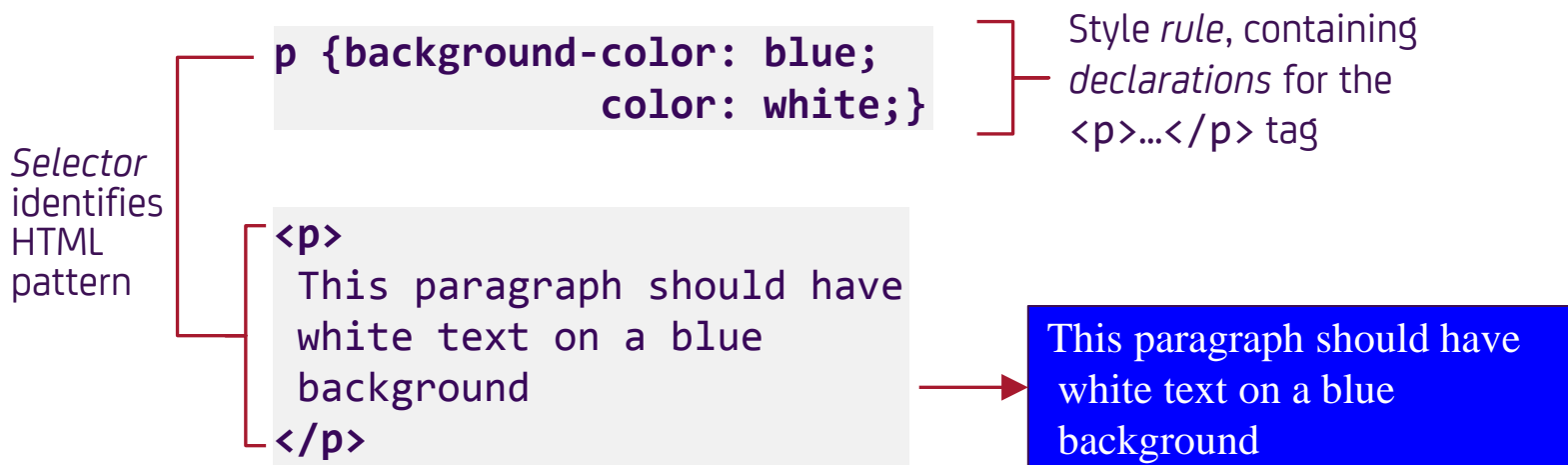
- A design with a dark, ornate interior and a glowing screen.
- A design featuring a large purple orchid in the foreground.
- A design with a blue and white color scheme and a stylized 'CSS' logo.
- A design with a dark background and a city skyline silhouette.
- A design with a light background and a traditional Japanese garden scene.

 Each redesign maintains the essential text and navigation elements while showcasing unique CSS techniques and visual styles.

<http://www.mezzoblue.com/zengarden/alldesigns/>

CSS Style Sheets

- Style sheets specify formatting *rules*
- Rules consist of *selectors* and *declarations*



Basic Style Sheet Syntax

Declaration(s) defined inside curly braces as style-property: value;

Selector → `p {background-color: blue;}`



Semi-colon ; separates declarations

`ul {margin-left: 15%; font-weight: bold;}`



Multiple selectors as comma separated list

"Apply declarations to h1 and h2 and h3 and h4"

`h1,h2,h3,h4 {background-color: white;
 color: blue;
 font-style: italic;}`



Internal Style Sheets

- Rules set out in `<style>` tags in the `<head>` section of the page

```
<html>
<head>
  <title>Internal Example</title>
  <style type="text/css">
    h1 {color: green; font-style: italic;}
  </style>
</head>
<body>
  <h1>Heading 1 in green italics</h1>
</body>
</html>
```

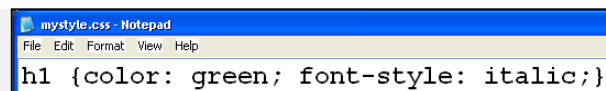


Heading 1 in green italics

External Style Sheets

- Style sheets are stored in separate files
 - Linked to current document
 - Multiple style sheets can be linked to a single page

```
<html>
<head>
  <title>CSS example</title>
  <link rel="stylesheet" type="text/css" href="mystyle.css" />
</head>
<body>
  <h1>Heading 1 in green italics</h1>
</body>
</html>
```



Heading 1 in green italics

Using @import Rules

- Alternative way to include external style sheets

```
<style type="text/css">  
  @import url("styles.css");  
</style>
```

- No difference in effect or behaviour, but can be more convenient
 - Only need one hard-coded `<link>` in XHTML document
 - Style sheets can be edited/attached/renamed without touching XHTML document

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

```
@import url("default.css");  
@import url("navbar.css");  
@import url("print.css");
```

Single linked style sheet
used to import actual
styles from separate files

Inline Styles

- Style can also be added *inline*
 - Uses `style` attribute with CSS rule(s) as value

```
<p style="color:white; background-color: blue;">Hello</p>
```

Hello

- Try and avoid if possible – mixes style and structure back up
- Can be a useful option if needed to overcome a *specificity* issue

More on CSS Selectors

- Three basic selector types define patterns to find in the mark-up
 - **Tag** – match all instances of the tag e.g. every `<p>...</p>`
 - **Class** – match tags containing this `class` attribute
 - **Id** – match the unique tag containing this `id` attribute
- Can be combined for more *specific* matches
- Additional syntax and operators allow precise control
- Combine with `<div>` and `` to build a *framework* for display

Classes as Selectors

- Used to apply styles to specific sub-sets of HTML tags
 - Tags are grouped using a `class` attribute
 - Tags can be in more than one class

```
<h1 class="special">A heading</h1>
<p>This is a normal paragraph</p>
<p class="special">A different class of paragraph</p>
```

- Define style rule(s) in the style sheet

```
p {text-align: left; color: red;}
.special {text-transform: uppercase;}
p.special {text-align: right; color: green;}
h1.special {text-decoration: underline;}
```

Dot (.) in selector pattern indicates a class e.g.

`p.special`
matches

`<p class="special">`

ID as a Selector

- Used to identify *unique* elements in the page
 - Uses an `id` attribute in the tag
 - Each `id` value can only be used *once* in any page (same `id` can be used on multiple pages though)

```
<p>The <span id="oneoff">Important</span> bit of...</p>
```



`oneoff` now provides a unique `id` for a single element in this document

- Hash (`#`) in the CSS selector pattern indicates an `id`

```
#oneoff {font-style: italic; font-weight: bold;}
```



The *Important* bit of...

More Selector Syntax

Selector	Pattern matched
p	All <p>
.special	<anytag class="special">
p.special	All <p class="special">
#thisBox	The only <anytag id="thisBox">
#thisBox p	All <p> nested <i>anywhere</i> inside the only <anytag id="thisBox">
#thisBox > p	All <p> that are <i>direct children</i> of <anytag id="thisBox">
#thisBox p.special	All <p class="special"> nested <i>anywhere</i> inside the <i>only</i> tag with the <i>id</i> of thisBox
div#thisBox p	All <p> nested <i>anywhere</i> inside the <i>only</i> <div id="thisBox">

<http://www.w3.org/TR/CSS2/selector.html>

Combining Selectors

CSS Rules

```
#section1 {color:red; text-align:center;}
#section2 {color:blue;}
.caps {text-transform:uppercase;}
#section2 p {text-decoration:underline;}
```

HTML

```
<div id="section1">
  <h1>Section one</h1>
  <p class="caps">A paragraph in
    section one</p>
</div>
<div id="section2">
  <h1 class="caps">Section two</h1>
  <p>A paragraph in section two</p>
</div>
```

#section2 p styles only applied to
<p>...</p> nested inside #section2



Section one

A PARAGRAPH IN SECTION ONE

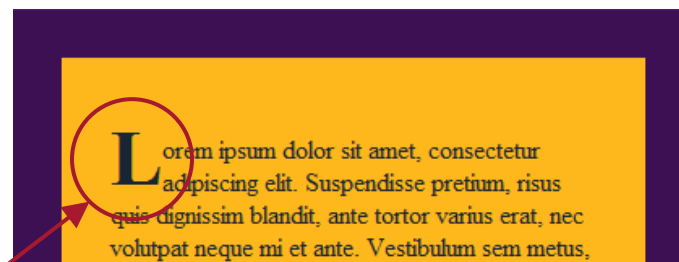
SECTION TWO

[A paragraph in section two](#)

Pseudo Elements

Selectors for special parts of some elements

```
p.opening:first-letter {  
  font-size: 300%;  
  font-weight: bold;  
  float: left; }  
  
```



```
<p class="opening">Lorem ipsum dolor sit amet,  
consectetur adipiscing elit. Suspendisse pretium,  
risus quis dignissim blandit, ante tortor... etc  
</p>
```

Pseudo Classes

Selectors for special status of some elements

```

a {text-decoration: none; font-weight: bold; color: #3c1053;}
a:hover {background-color: #3c1053; color: #ffb81c;}
a:active {font-style: italic;}
a:visited {color: #8c4779;}

```

```

<p>
<a href="http://lipsum.com">
Lorem ipsum</a> dolor... etc </p>

```



Cascading Style Sheets

- All available styles for a page are combined as it loads
 - Final appearance for each element is composite of all appropriate rules
- Conflicting property values resolved by simple rules
 1. **Source:** User-specified styles (in the browser) are more specific
 2. **Specificity:** Relative weighting of selector priority
 3. **Order declared:** If specificity value are the same then "last one wins" (means inline styles are always more specific)
- Specificity – a measure of importance
 - The more *specific* the rule is the greater priority its declarations have
 - Easy to calculate...

Specificity Calculator

Count the number of ID, class and tag names in each selector

Selector	IDs	Classes	Tags
#thisbox	1	0	0
.special	0	1	0
p	0	0	1
p.special	0	1	1
#thisbox p	1	0	1
#thisBox > p	1	0	1
div#thisBox p	1	0	2
#thisBox p.special	1	1	1

Basic values of:
ID = 100
Class = 10
Tag = 1

More *specific*
combinations have
higher values

CSS Units

- CSS supports many types of measurement unit
- **Absolute** units calculated independently of other page content and/or browser defaults
 - Useful for precise layout
 - Include Pixels (px), Points (pt), Millimetres (mm)
- **Relative** units calculated proportionally against other page content or a browser default
 - e.g. currently available width, default text size etc.
 - Include Percentages (%), Ems (em), Exes (ex)
 - Also special relative units for text e.g. small, large, x-large ... etc.
- Good design uses a combination of both

CSS Colours

- CSS allows rich control over colour (color!)
 - Any colour can be specified using RGB or Hex (hexadecimal) codes
 - Only 17 names are actually valid in CSS2

FFF	CCC	999	666	333	000	FFC	FF9	FF6	FF3										
FFF	CCC	999	666	333	000	FFC	FF9	FF6	FF3										
99C	CCC					CC9	FFC	FFC	FF9	FF6	CC3							CC0	033
C00					CC9	900	C33	C66			300								
CCF	CF3	333	666	999	CCC	FFF	F00	CC9	C06	330	660	990	CC0	FF0	FF3	FF0			
F00	CF3	300	600	900	CCC	FFF	F00	933	633	000	000	000	000	000	366	033			
99F	CCF	99C	666	999	CCC	FFF	996	996	663	993	CC3	FF3	CC3	FF6	FF0	FF0			
F00	F66	C33	633	933	C33	F33	600	900	333	333	333	333	333	966	699	066			
66F	99F	66C	669	999	CCC	FFF	996	663	996	CC6	FF6	990	CC3	FF6	FF0	FF0			
F00	F66	C33	900	966	CC6	096	666	666	666	666	666	666	033	399	6CC	099			
33F	66F	339	66C	99F	CCC	FFF	CC9	CC6	CC9	FF9	FF3	CC0	990	FF3	FF0	FF0			
F00	F33	900	000	F33	CC9	F99	966	606	999	999	399	066	066	3CC	0CC	0CC			
00C	33C	336	669	99C	CCF	FFF	FFC	FF9	FFC	FF9	CC6	993	660	CC0	330	330			
C00	C00	C00	600	933	CC6	F99	CCC	9CC	9CC	9CC	699	366	033	099	033				
33C	66C	00F	33F	66F	99F	CCF			CC9	996	993	990	663	660					
633	096	F00	F33	F66	F99	FCG			9CC	699	399	999	366	066					
006	336	009	339	669	99C				FFC	FF9	FF6	FF3	FF0	CC3	CC3				
600	633	900	933	966	CC9				CCF	9FF	6FF	3FF	0FF	6CC	3CC				
003	00C	006	339	66C	99F	CCF	339	99C	CCC	CC9	996	663	330	990	CC0	CC0			
300	C00	C33	633	966	CC9	FCC	FFF	9FF	CCF	9FF	6CC	399	066	0CC	0CC				
00F	33F	009	00C	33F	99F	99C	006	669	999	999	993	660	660	CC3	CC0				
F33	F66	933	CC6	F99	FFF	CCC	6CC	9CC	9FF	9CC	3FF	0CC	099	3FF	0FF				
00F	66F	33C	009	66F	66C	669	003	336	666	666	330	993	CC6	990					
F66	F99	CC0	990	FFF	CCC	999	300	099	6FF	6CC	099	099	3CC	FFF	0FF				
00F	66F	33C	33F	33C	339	006	003	333	333	333	333	663	666	669					
F99	FCC	CC9	FFF	CCC	999	666	669	999	3FF	3CC	399	366	3CC	6FF	0FF				
00F	33F	00F	00C	009	006	003	339	336	000	000	000	000	000	663	330				
FCC	FCC	FFF	CCC	999	666	333	9CC	6CC	9FF	0CC	099	066	033	3FF	0FF				
00C					009	33C	66C	669	336	003									
699					9CC	CCF	CCF	9FF	6FF	3CC									
					00C	009	006	003											
					CCF	9FF	6FF	3FF											

color: red; ✓
 color: magenta; ✗
 color: rgb(0,32,234); ✓
 color: #0000ff; ✓





© Copyright information and licensing

This material from JISC Netskills Share is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License.

<http://creativecommons.org/licenses/by-nc-sa/3.0/>

You are free to download, embed, adapt and share this material within the terms of that license.

Find out more at: <http://www.netskills.ac.uk/share>

A decorative graphic at the bottom of the page consisting of several overlapping, wavy, light purple lines that create a sense of motion and depth.