

CSS3

Prof. Cesare Pautasso

<http://www.pautasso.info>

cesare.pautasso@usi.ch

[@pautasso](#)

Applying CSS

How to apply a stylesheet to a document?

1. Straight into the HTML tags

```
<p style="color: red">text</p>
```

HTML should be presentation-free, so in-line styles should be avoided as much as possible

2. With internal, embedded styles

```
<head><style>p {color:red}</style></head>
```

Embedded styles should be used only with special pages that should look different than the others

3. With external, referenced styles, stored in separate CSS files

```
<head>  
  <style type="text/css">  
    @import url("style.css")  
  </style>  
  <link href="style.css" rel="stylesheet"/>  
</head>
```

CSS Rules

CSS is a declarative language. It uses **rules** to specify how parts of a document should be formatted

```
selector { property: value }
```

Each CSS rule has:

- A **Selector** targeting the document elements
- to be formatted using one or more **property-value** pairs

CSS Rule Example

```
body {  
  font-size: 0.8em;  
  color: navy;  
}
```

Apply this rule to the body element.

Set the font-size to 0.8em and the text color to **navy**.

Property Value Types

Property values have different data types
(String, Size, Enumeration, Color, URL)

```
h1 {  
  font-family: "Arial";           //String  
  font-size: 1.1rem;             //Size  
  font-style: italic;            //Enum  
  color: navy;                   //Color  
background-color: #001122;       //Color  
background-image: url("/img/h1.png"); //URLs  
  text-shadow: #000 2px -2px 1px;  
}
```

Colors

Colors can be represented with:

- RGB values

```
rgb(255,0,0)
```

- Hexadecimal RGB values

```
#ff0000 = #f00
```

- enumeration

```
black, yellow, red,  
transparent
```

- RGBA values

```
rgba(255,0,0,0.5)
```

- HSLA values

```
hsla(0,100,50,1)
```

<http://rightjs.org/ui/colorpicker/demo>

Lengths and Sizes

Lengths can be expressed in several units of measures

- em the height/width of an **M** character...
- rem ...of the font of the root element
- px pixels
- % percentage relative to the parent's value
- 0 (Zero) does not need a unit

A Web page should be flexible, so use pixel sizes sparingly

- Pixels recommended for border thickness or spacings
- Very bad for font sizes (some browsers will not zoom)
- Percentages can be dangerous when used with nested tags

Main Properties

background (-color, -image, -position)

border (-width, -style, -color) x (-top, -right, -bottom, -left)

border-radius

font (-family, -size, -weight)

text (color, direction, letter-spacing, text-align,
text-decoration, text-transform, word-spacing, **shadow**)

tables (border-collapse, caption-side, empty-cells)

list (-style, -style-image, -style-type)

margin (-top, -right, -bottom, -left)

padding (-top, -right, -bottom, -left)

layout (display, clear, float, position, visibility, cursor)

positioning (left, right, bottom, top, clip, overflow,
vertical-align, z-index)

dimension (height, width,
max-height, max-width, min-height, min-width)

transform

transition

<http://www.w3schools.com/cssref/>

Selectors

Control which document elements are affected by a rule.

Selectors refer to:

Tags	<code>h1</code>
Classes	<code>.navigation</code>
IDs	<code>#home</code>
Wildcards	<code>*</code>
Pseudo-classes	<code>:hover</code>

Selectors can be combined, grouped, and nested to make them more selective

Universal Selector

```
* {
  color: #000;
  margin: 0;
  padding: 0;
}
```

- Use the `*` selector to setup default properties of ALL elements of your page that do not have a more specific style
- Browser default may differ, so this helps you to start with known formatting defaults

Tag Selector

```
body {  
  color: #000;  
  background: #fff;  
  font-family: verdana, arial, sans-serif;  
  font-weight: normal;  
}
```

- This applies the formatting to elements of a certain type (e.g., the <body> of a page)
- Also all children elements within the <body> will inherit the same format (unless there is an overriding rule)

Class Selector

```
.navigation {
  text-decoration: underline;
}
```

- The same rule is applied to all document elements of a particular class (independent of their element tags)

```
<p class="navigation">Home - Introduction</p>
<div class="navigation">...</div>
```

- More than one element can share the same class
- The same element can be tagged with multiple classes

```
<p class="navigation active">Home - Introduction</p>
```

ID Selector

```
#home {
  margin-top: 2p;
  border-bottom: 1px #abc solid;
}
```

- The rule applies only to the document element with the given id attribute

```
<p id="home"> Welcome to my Homepage </p>
```

- Important: to distinguish elements, id attributes must have unique values within the same page

Attribute Selector

```
a[href^="http:"] {
  background: url("external.png") no-repeat right top;
  padding-right: 10px;
  border-bottom: 2px solid;
}
a[href^="http://www.mysite.com"] {
  background-image: none;
  padding-right: 0;
  border-bottom: 1px dotted;
}
```

The rule applies only to the document element with the matching **attribute value**

```
<a href="http://www.google.com"> External Link </a>
```

Matching Operators

- = equals
- ~= contains word
- ^= starts with
- \$= ends with
- *= contains substring

Nested Selectors

Selectors separated by ' ' (space) target nested element tags (at any level)

```
ul li
```

defines a rule for any element `` within any ``

```
#top a
```

defines a rule for any element `<a>` within the element with `id="top"`

```
#top .navbar a
```

defines a rule for any element `<a>` within some element of `class="navbar"` contained within the element having `id="top"`

Combined Selectors

Selectors can be concatenated to further qualify the target element

```
ul.publications
```

defines a rule for any element `` of `class="publications"`

```
div#top
```

defines a rule to match the element `<div id="top"/>`

```
a.navbar#top
```

defines a rule for the element `<a>` of `class="navbar"` with `id="top"`

CSS3 Pseudo-Classes

Structural pseudo-classes can filter the selected elements based on their position on the document tree

```
:nth-child(an+b)  
:nth-last-child(an+b)
```

Note: Elements are counted starting from 1

```
:first-child  
:last-child  
:only-child  
:empty  
:root
```

```
:nth-child(2n+1)  
:nth-child(odd)  
  
:nth-child(2n)  
:nth-child(even)
```

Useful shorthands

Target Selector

Apply a rule only to the document element with the id attribute matching the URL fragment of the page

```
:target { background-color: red }
```

```
<p id="first">No Match</p>
```

```
<p id="second">Match</p>
```

<http://www.pautasso.info/index.html#second>

Negative Selector

Apply a rule to all elements which do **not** match a selector

```
.red {
    background-color: #F00;
}
:not(.red) {
    background-color: #000;
}
```

```
<p class="red">Match</p>
```

```
<p class="black">Not Match</p>
```

```
<p class="white">Not Match</p>
```

Selector Specificity

Rules with more specific selectors override rules with more generic ones

```
<h1>
```

```
<h1 class="summary">
```

```
<h1 class="summary" id="intro">
```

```
<h1 class="summary" id="intro" style="color:navy">
```

```
h1 { color: green }
```

```
h1.summary { color: orange }
```

```
h1.summary#intro { color: red }
```

Priority: Inline Styles > ID > Class > Tag > *

Conflicts

What happens if two rules apply to the same element?

Rules are prioritized:

- More specific styles override general styles
- Page styles replace user-defined styles

User-defined Inline Internal (head) External Browser default
--

Use the `!important` flag to prioritize styles

If two styles have the same priority, the one defined "last" takes precedence

Page Layout with CSS

Much better than using nested table elements (originally the only way, but no more!)

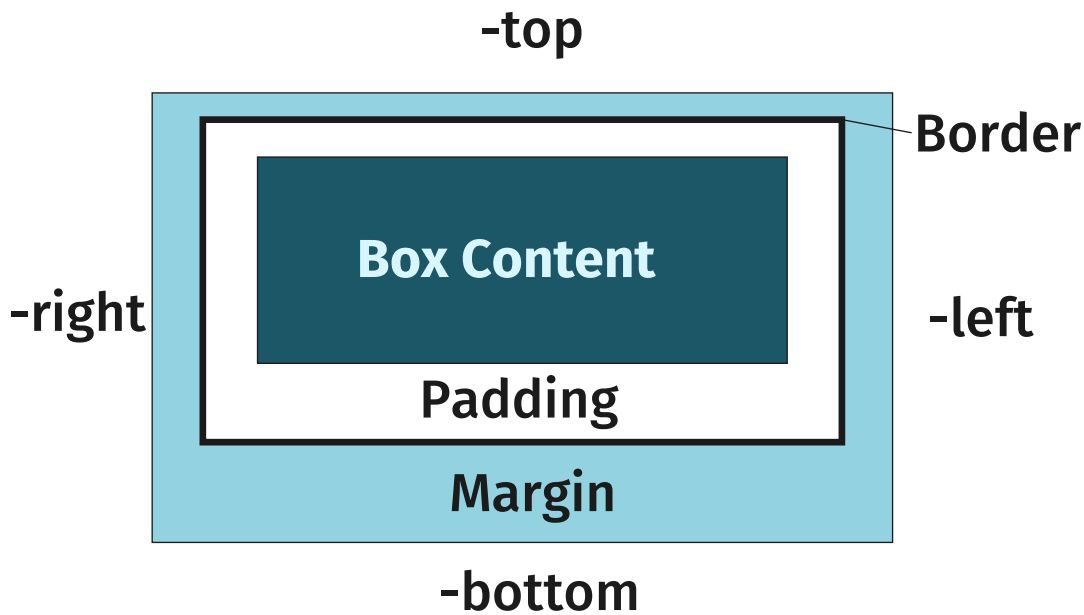
1. Control the size and positioning of each page element:

- margin
- padding
- border

2. Specify the layout properties of each page element:

- display
- position
- float
- width, height
- top, left, bottom, right

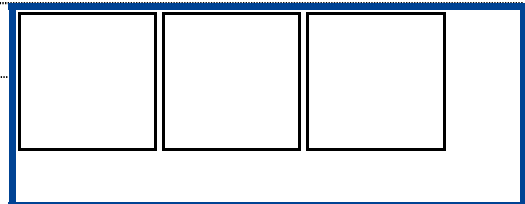
CSS Box Model



Display

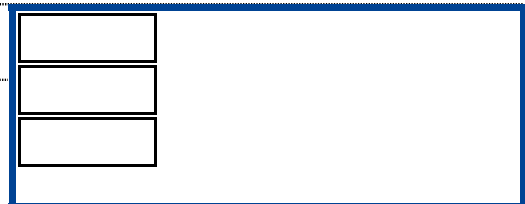
`display: inline`

Do not break the flow of text
(like ``)



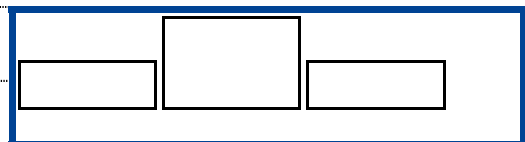
`display: block`

Line break before and after the
element (like `<div>`)



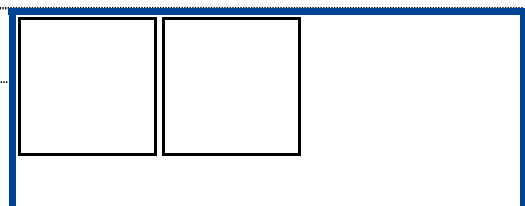
`display: inline-block`

Keep the block on the same line



`display: none`

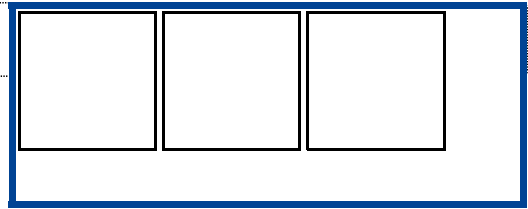
Hide the element



Position

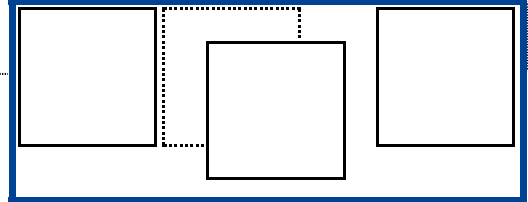
`position: static`

Automatic Layout (default, ignore left, top, right, bottom)



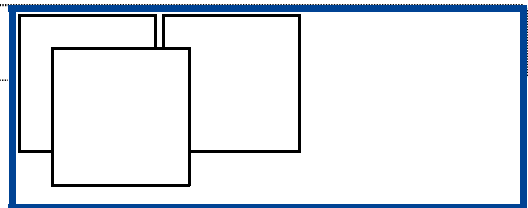
`position: relative`

Offset the position with respect to the automatic one



`position: absolute`

Position with respect to the first non-static parent box. The element is pulled out of the normal layout flow.



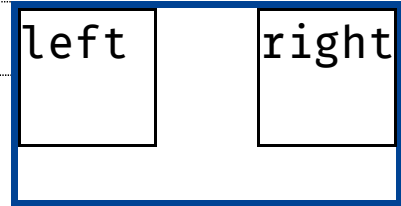
`position: fixed`

Position with respect to the window (instead of the page)

Floating Boxes

`float: left | right | none`

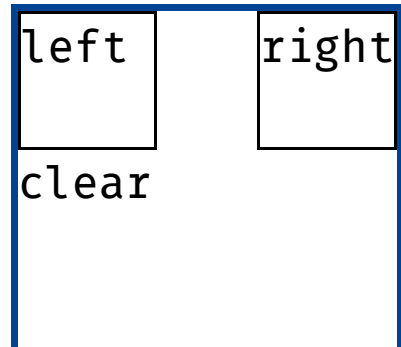
Shift a box to the right/left of the container box.



If there is not enough space, boxes can wrap around

`clear: left | right | both`

Force the following boxes to continue under the previous highest floated box (like `\clearpage` in LaTeX)



Flexible Layout

```
display: flex
```

Apply the flexbox mode to the content.

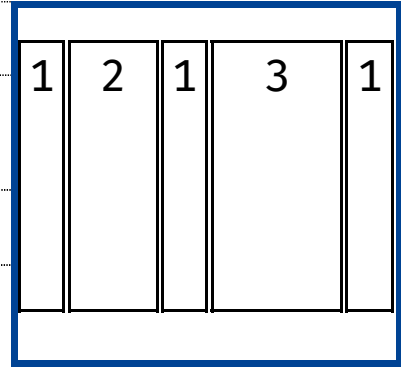
```
flex: n
```

Grow the content proportionally

```
min-width: length
```

Also works with unflexible content

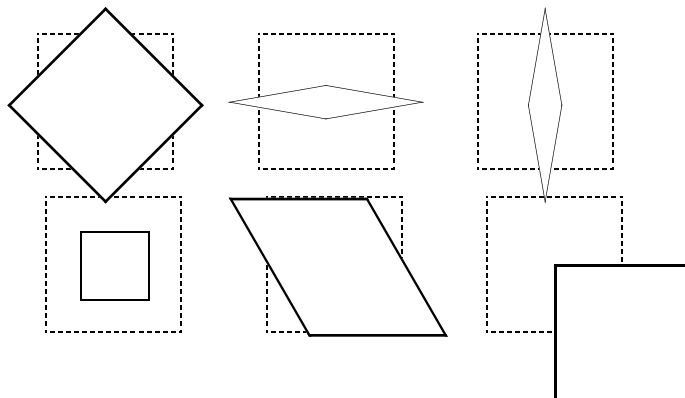
<http://css-tricks.com/snippets/css/a-guide-to-flexbox/>



2D/3D Transform

Any page element can be transformed!

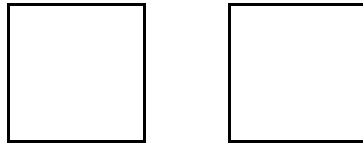
```
#box { [-webkit-|-o-|-moz-]transform:
  rotate(N deg)
  scale(x,y)
  skewX(N deg)
  translate(X px, Y px);
}
```



Transition

Declaratively smooth the transition between changes in the formatting of any page element (no need of JavaScript for animations)

```
#box {  
  transition-property: background-color, width;  
  transition-delay: 1s;  
}
```



Media-specific styles

```
@media print {
  #navigation { display: none; }
  body { font-size: 10pt; font-family: times; }
}
```

all	every media (default)
aural	speech synthesizers
braille	⠠ ⠠⠠ ⠠⠠⠠ ⠠⠠⠠⠠ ⠠⠠⠠⠠⠠
handheld	small screen devices
print	paper
projection	beamers
screen	computer screens

Media Queries

Expressions can use logical and, or, not operators

```
@media (expression) {
  /* CSS */
}
```

```
<link rel="stylesheet" media="(expr)" href="m.css" />
```

color	0 for b/w devices
orientation	landscape portrait
min/max-width	width of the browser window
min/max-height	height of the browser window
resolution	2dppx retina display

Tools

- [CSS Validator \(http://jigsaw.w3.org/css-validator/\)](http://jigsaw.w3.org/css-validator/)
- [HTML5 Validator \(http://html5.validator.nu/\)](http://html5.validator.nu/)
- [XHTML Validator \(http://validator.w3.org/\)](http://validator.w3.org/)
- [Firefox Web Developer Extension \(http://addons.mozilla.org/en-US/firefox/addon/60\)](http://addons.mozilla.org/en-US/firefox/addon/60)
- [Firefogg \(http://firefogg.org/\)](http://firefogg.org/)

References

- <http://www.w3schools.com/css/>
(<http://www.w3schools.com/css/>)
- <http://www.alistapart.com/articles/>
(<http://www.alistapart.com/articles/>)
- Andy Budd, Cameron Moll, Simon Collison, CSS Mastery – Advanced Web Standards Solutions, APress 2006
<http://www.cssmastery.com/>
(<http://www.cssmastery.com/>)
- Andy Clarke, Transcending CSS. The Fine Art of Web Design, New Riders ISBN 0321410971 -
<http://www.transcendingcss.com/>
(<http://www.transcendingcss.com/>)

