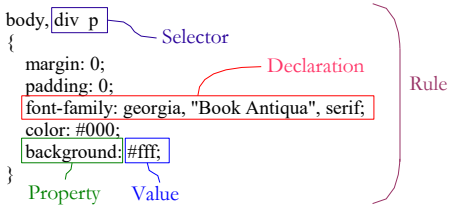
CSS2.1 Core Quick Reference

Syntax

Terminology



Semicolons

Semicolons are separators not terminators. Put a semicolon at the end of each declaration anyway.

Comments

```
/* Comments */
```

Case

All CSS style sheets are case-insensitive, except for parts that are not under the control of CSS.

So CSS elements and attributes that reference XHTML elements and attributes must be lowercase.

See Also John Bentley > Markup & CSS Code Convention - Naming and Style

Placing in XHTML

Linking:

```
<link type="text/css" rel="stylesheet"
href="CssQuickReference1.css" />
```

Embedding:

```
<style>
  <!--
    @import url(CssQuickReference2.css);
    p { color: green };
  -->
</style>
```

Later stylesheets (and rules) take precedence over earlier ones regardless of whether embedded or linked.

Note styles in CssQuickReference2.css, a later stylesheet, take precedence over CssQuickReference1.css, an earlier stylesheet.

Media

Target different media from within stylesheets or when linking from XHTML.

Within Stylesheets:

```
@import url(fancyfonts.css) screen;
@media print {
  /* style sheet for print goes here */
}
@media screen, print {
  body { line-height: 1.2 }
}
```

Linking from XHTML:

```
<link type="text/css" rel="stylesheet"
href="philorumPrint.css" media="print" />
```

Character Encoding

1. Character encoding: UTF-8
2. Byte Order Mark (BOM): None.
3. Charset at-rule: None.

Justification:

```
@charset "utf-8"; /* Don't do this */
```

Inserting a charset at rule fucks up the W3C validator when you have multiple stylesheets and are using the Firefox "Web Developer" plugin to perform a local CSS validation. This is because the plugin will combine all your stylesheets into one before submitting. The validator will choke on an @charset in the middle of this combined file.

Without an explicit @charset rule (and no Byte Order Mark) standard compliant browsers must assume UTF-8. Therefore encode your stylesheet in UTF-8.

Byte Order Marks must not be set for XHTML. For ease of setting options in text editors, and for consistency with XHTML, drop the BOM in stylesheets.

Selectors

Selector types

pattern	Meaning
Common	
e	Matches any e element (i.e., an element of type e).
.warning	Matches any element with class warning. Same as *[class~="warning"].
#myid	Matches the element with id = "myid"
Other Common	
div.warning	**Language specific.** (In HTML, the same as DIV[class~="warning"].)
e#myid	Matches any e element with ID equal to "myid".
e:link e:visited	Matches element e if e is the source anchor of a hyperlink of which the target is not yet visited (:link) or already visited (:visited).
e:focus e:hover e:active	Matches e during certain user actions.
e f	Matches any f element that is a descendant of an e element.
Wildcard	
*	Matches any element.
Relationship	
e > f	Matches any f element that is a child of an element e.
e:first-child	Matches element e when e is the first child of its parent.
e + f	Matches any f element immediately preceded by a sibling element e.
e:lang(c)	Matches element of type e if it is in (human) language c (the document language specifies how language is determined).
More Psuedo Elements	
p:first-line p:first-letter	Matches first line or first letter of the p element
e::before e:after	Use with a content declaration.
Attribute Selectors	
e[foo]	Matches any e element with the "foo" attribute set (whatever the value).
e[foo="warning"]	Matches any e element whose "foo" attribute value is exactly equal to "warning".
e[foo~="warning"]	Matches any e element whose "foo" attribute value is a list of space-separated values, one of which is exactly equal to "warning".
e[lang|="en"]	Matches any e element whose "lang" attribute has a hyphen-separated list of values beginning (from the left) with "en".

Select Elements Based On Context

```
p em strong
{
    color: red;
}
```

Select `strong` elements that are descendants of `em` `that`, in turn, are descendants of `P`.

Apply the same rule to groups of Elements

```
h1, p
{
    color: purple;
}
```

Use a comma.

Selecting Link States

Use the right order: a:link; a:visited; a:focus; a:hover; a:active.

Mnemonic: LoVe beFore HAte.

Conflict Resolution

Basic

Obtain the value (specified) of an element's property:

-- Value available from Cascade? Use it else:

---- Value available from inheritance (from Parent)? (True if "is Inherited") Use it else:

------ Value available from initial setting. (it's default).

Cascade

For each Element's Property match for **Media type** then sort by:

1. **Weight: Importance** (normal or important) and **origin** (author, user, or user agent).
   a. user agent style sheets
   b. user normal style sheets
   c. author normal style sheets
   d. author important style sheets
   e. user important style sheets
2. **Specificity** of selector: more specific selectors will override more general ones. See below.
3. Finally, by order specified: if two rules have the same weight (importance, origin) and specificity, the **latter in the Stylesheet** wins. Rules in imported style sheets are considered to be before any rules in the style sheet itself.

Cascade Specificity

The procedure involves counting four items:
1. Count 1 if an element has a style attribute (in the html).
2. The number of ID attributes in the selector.
3. The number of other attributes in the selector.
4. The number of tag names in the selector.

Higher the number the more specific. Eg:

```
{ ... }
 /* styleattr=0, ID = 0, Attributes = 0, Tags = 0 :   0 */

div { ... }
 /* styleattr=0, ID = 0, Attributes = 0, Tags = 1 :   1 */

.message { ... }
 /* styleattr=0, ID = 0, Attributes = 1, Tags = 0 :  10 */

div.message { ... }
  /* styleattr=0, ID = 0, Attributes = 1, Tags = 1 :  11 */

.message.big { ... }
/* styleattr=0, ID = 0, Attributes = 2, Tags = 0 :  20 */

#blurb { ...}
 /* styleattr=0, ID = 1, Attributes = 0, Tags = 0 : 100 */

(in HTML) style="…"
/* styleattr=1, ID = 0, Attributes = 0, Tags = 0 : 1000 */
```

Note that, for practical purposes, a higher ID count always beats any number of attributes, and in turn a higher attribute count always beats any number of tags. So a rule selector with an ID would weigh more than one with nine class attributes but no ID.

See Mike Hall > Using Style Sheets < http://www.brainjar.com/css/using/>

Generic Fonts

Monospace
Serif
Sans-Serif
FANTASY
Cursive

Shorthand

Some properties serve as a shorthand for setting multiple properties.

Font Shorthand

Instead of

```
p {
  font-style: normal;
  font-variant: normal;
  font-weight: 600;
  font-size: 1.5em;
  line-height: 1.2;
  font-family: Chiller;
}
```

We can use the shorthand:

```
p { font: italic small-caps 300 .9em/1.4
"verdana ref" , "Papyrus"; }
```

Properties omitted get a cascade, that is value from a previous rule, or get their initial value.

```
/* Here font-style and font-variant both get
their initial value 'normal' (if not obtained
from a cascade, a previous rule). */
p { font: 700 1.9em/1.4 "Rockwell"; }
```

This cascade is not W3C but occurs in IE6 and Moz1.6.

At a minimum you should define font-size and font-family otherwise the declaration is ignored.

```
p { font: 1em garamond; }
```

You must use the order as shown in the initial 2 examples.

Border Shorthand

Instead of:

```
p
{
  border-style: inset;
  border-width: thin;
  border-color: Maroon;
}
```

We can use the shorthand:

```
p { border: solid 1px blue; }
```

Properties get their initial value. No cascade occurs, that is previous rules are ignored.

If an element's border color is not specified with a border property, user agents must use the value of the element's 'color' property

```
/* Here border color is set to the elements
color, blue. */
p { border: solid 1px; }

/* Here the border is set to solid, medium (initial value
ignoring any previous rule), and element's color blue */
p { border: solid; }
```

Margin, Padding & Border-Width Shorthand

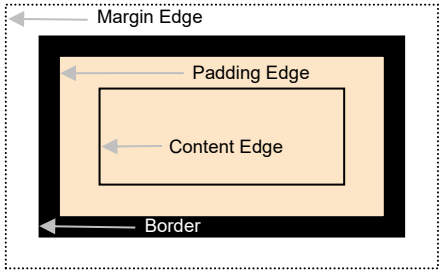
```
[all]
body { margin: 2em }

[top bottom] [sides]
body { margin: 1em 3em }

Middle is sides
[top] [sides] [bottom]
body { margin: 3em 1em 2em  }

Clockwise from top:
[top] [right] [bottom] [left]
body { margin: 1em 2em 3em 4em  }
```

Box Model



Margins are always transparent.

Background settings for an element includes both the padding and content areas.

Box types

Block V Inline boxes: A fundamental distinction.

Positioning Schemes

static (Normal Flow)
The box is a normal box, laid out according to the normal flow.

relative
The box's position is calculated according to the normal flow (this is called the position in normal flow). Then the box is offset relative to its normal position.

absolute
The box's position (and possibly size) is specified with the 'top', 'right', 'bottom', and 'left' properties. These properties specify offsets with respect to the box's containing block's margin edge. Absolutely positioned boxes are taken out of the normal flow. This means they have no impact on the layout of later siblings.

Absolutely positioned box's don't move out of the the position they would have if static until they have one of top, right, bottom or left set. However they are always removed from the flow.

fixed
The box's position is calculated according to the 'absolute' model, but in addition, the box is fixed with respect to some reference. In the case of handheld, projection, screen, tty, and tv media types, the box is fixed with respect to the viewport and doesn't move when scrolled. In the case of the print media type, the box is rendered on every page, and is fixed with respect to the page.

Length

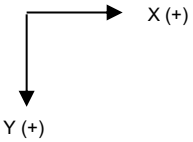
Relative:
em: the 'font-size' of the relevant font. ("M")
ex: the 'x-height' of the relevant font. ("x")
px: pixels.

Percentage:
%

Absolute:
in: inches — 1 inch is equal to 2.54 centimeters.
cm: centimeters
mm: millimeters
pt: points — the points used by CSS 2.1 are equal to 1/72th of an inch.
pc: picas — 1 pica is equal to 12 points.

Use relative/percent for screen, absolute for print.

Coordinate Axes



Containing Block

Absolutely positioned box's don't move out of the flow (they don't become "absolute" for containing block alignment purposes) until they have one of top, right, bottom or left set. And only for each axis, x or y, set. Likewise for fixed.

The Initial Containing box is defined by the Body element. The body can have its margins and padding set. The body background color covers both the padding and margin area.

Initial Containing Box = Body Margin Edge = Viewport = PageBox.

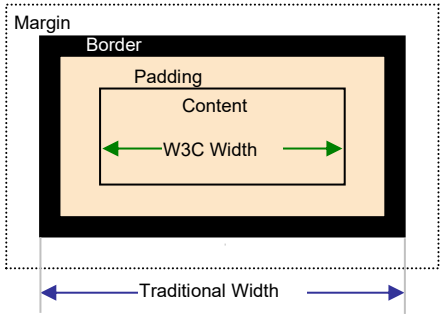
Table 1 How the inner block is aligned to its containing block

Inner Box	Which is Ancestor Containing Box?	If Containing box.	Which Edge of Containing box?
Static or Relavtive	Nearest Block-Level, Inline-Level, Table Cell		Content Edge
	If none then Init. CB		Body Content Edge
Absolute	Nearest Absolute, Relative, or Fixed	Inline	Content Edge
		Block	Padding Edge†
	If none then Init. CB		Body Margin Edge
Fixed	Init. CB		Body Margin Edge

† WinIE6 aligns inner box to containing box's top padding edge only (as it should). It wrongly aligns the left edge, that is the x axis, to the content edge. To remove this problem set containing box padding to 0.

Inline boxes should only contain other inline boxes. Therefore if a block-level box is to be positioned its parent ought be an block-level box (not an inline level box).

Width



Properties that can get a value of auto:

'left', 'margin-left', 'width', 'margin-right', 'right'

'top'
'margin-top'
'height'
'margin-bottom'
'bottom'

Browsers never adjust the width of the borders or padding (Elizabeth Castro > XHTML, 5th > P191).

To center a block box set margin-right and margin-left to auto and width to a specific size.

```
#logo
{
    display: block;
    margin: 0 auto;
    width: 248px;
}
```

Percentages refer to width of containing block.

Using a combination of specific or the 'auto' value on box properties will make the box:
⇒ Expand to fill the containing block
⇒ Shrink to its own contents.
⇒ A single auto expands to fill the containing block.
⇒ Scales replaced elements. Eg img.
⇒ Is centered (see above).

This depends on whether the block is:
⇒ Block-level V inline-level
⇒ Replaced V Not replaced.
⇒ Floating or Not.
⇒ Absolutely/relatively positioned or not.

Make width shrink to fit. Use either of:
* display: inline-block
* float
* position: absolute

http://www.brunildo.org/test/shrink-to-fit.html

Cross Browser Width (& Height) Setting

Introduction

In the W3C box model, the width of an element gives the width of the content of the box, excluding padding and border. In the traditional box model, the width of an element gives the width between the borders of the box, including padding and border.

If you use a doctype to switch to 'Strict mode', Explorer 6 Windows and Opera 7 switch to the W3C box model. To move Explorer 5 and 5.5, which don't support doctype switching, to the W3C model you need Tantek Çelik's famous Box Model Hack.

Koch > BC

Instructions

You can set width and height on static blocks as well as for the other position schemes.

Use one of three cross browser strategies for setting the width (& analogously for height) of a box:

1. If you want expand to containing block or shrink to fit: Set width to auto.

2. If, on the target element, you need to fix the width; and If you also need to set one of padding or borders: use the Tantek Çelik box model hack.

   Calculate the total available space.

   Calculate space used-by-element:

   For W3C browsers this is:
   margin-left + border-left + padding-left + padding-right + border-right + margin-right

   For Traditional browsers this is:
   margin-left + margin-right

   Subtract from total available space the used-by-element space to get two values for width. One for W3C, the other for traditional.

   Plug the two width values into the Tantek Çelik box model hack.

3. If, on the target element, you need to fix the width; and If you also DON'T need to set neither padding nor borders: you don't need any hacks.

   You just calculate a single width value, as above, using:
   margin-left + margin-right.

   Note you can set margins without need a the Tantek hack.

   Furthermore you can **nest** another element within this and apply all the positioning and margin/border/padding settings you like.

   So you can probably avoid the Tantek Çelik box model technique altogether.

Try to avoid applying padding/borders and a fixed width to an element.

IE5 gets the box model wrong... There are ways around this, but it's best to side-step the issue by applying the padding to the parent element instead of the child that gets a fixed-width.

Dave Shea > CSS Crib Sheet

Position the element to which you are fixing a width. Then nest another element within. Apply as much padding, border, and margin, positioning as you like to this nested element.

A cross browser (IE5 OK) box model technique. This helps avoid having to perform complex box model hack calculations.

Bentley

Height

By default the height is shrink to fit (generally).

To make a div 100% of parent element, set the containing div height to 100%, all the way to html and body.
http://www.tutwow.com/tips/quick-tip-css-100-height/

Images

Explicitly specify the dimensions (width and height) of your images in XHTML not CSS.

```
' Do this in XHTML
<img src="images\YosemiteElCapitan.jpg"
alt="Yosemite El Capitan" width="256"
height="192" />

' Not this is CSS
#background img
{
    width: 256px;
    height: 192px;
}
```

There are three ways you can deal with dimensioning images: not setting them; setting them in CSS; setting them in XHTML.

Setting dimensions in XHTML is the only way to ensure the page layout remains still as downloading occurs.

It is often claimed that explicitly setting the dimensions (in XTHML) gives speedier rendering of the page (eg *Castro > P 106.*) However, experiment proves this to be negligible.

*Bentley > Experiment > CSSExamples > imageLoadTimeTest.**

Handle img aspect ratios

To properly maintain image dimension ratios as the image is resized…

… When setting an img src

```
figure img {
  max-height: 152px;
  width: auto;
  height: auto;
}
```

https://stackoverflow.com/a/17183996/872154

… When setting a background img

```
/* cover will give you a scaled up image */
background-size: cover;

/* contain will give you a scaled down image
*
background-size: contain;
width: 150px;
height: 100px;
background-image:
url("http://i.stack.imgur.com/2OrtT.jpg");
background-size: cover;
background-repeat: no-repeat;
background-position: 50% 50%;
```

https://stackoverflow.com/a/11763434/872154

(CSS Level 3) Fixed image height.

```
figure img {
  height: 152px;
  width: auto;
  object-fit: cover;
}
```

Overflow and Clip

Overflow

Apply overflow to the **ancestor** element that contains the child(ren) that are too large.

The overflow region occurs at the **padding edge**.

Clipping

The clipping region, by default, occurs at the **border edge**.

A clipping value moves a point along the coordinate axis (in X or Y direction) then clips everything in the direction of that value ie

↑

Top

← left right →

bottom

↓

You must enter all four values in the rectangle. Auto values start from the default border edge.

```
clip: rect(10px auto 200px 10px);
```

Floats

Apply the float to the element you are floating, not to the elements that flow around it.

You must explicitly set the width property on the floated element.

Always place the element before the content that will flow next to it.w

You use clear on those elements that you want clear of floating objects. That is, you don't set clear on the floated object.

Expanding floats (aka "clearing floats")

To prevent inner floats from busting out beyond an outer block there are two options.

```
<div id="outer" class="clearfix">
  <div id="inner"> <h2>A Column</h2> </div>
   <h1>Main Content</h1>
   <p>Lorem ipsum</p>
 </div>
```

CSS Technique 1 (Preferred): float the parent.

```
.clearFix {
  float: left;
  width: 100%
}
```

This technique allows children that are two big for the parent to be give absolute position and bust out of the parent without triggering scroll bars (or being hidden), as might occur using CSS technique 2 below.

See David Shea > http://www.mezzoblue.com/archives/2005/03/03/clearance/

CSS technique 2: overflow: hidden the parent.

```
.clearFix {
  overflow: hidden;
  width: 100%
}
```

We used "hidden" rather than "auto" to avoid scoll bars appearing.

If floating the parent causes problems you might like to resort to CSS technique 2.

http://www.quirksmode.org/css/clearing.html

Z-Index

z-index only determines the order within each group of siblings.

Counters

```
/* Step 1: define the counter scope with an
identifier.*/
table { counter-reset: phase; }

/* Step 2: define the counter string suing
identifier. */
tbody td:before {
    content:  counter(phase) ". ";
}
/* Step 3: define where the increment takes
place. */
tbody tr { counter-increment: phase; }
```

Content with :before and :after

```
label:after
{
  content: ":";
}
```

White Space

Set whitespace like this

```
    #targetPre { white-space: pre; }

    <p id='targetPre'>
        Text with tabs, space, and line
breaks in source (shown as \r\n). Lorem ipsum
dolor sit amet consectetuer\r\n
        nulla sed tincidunt lacus enim Sed.
Morbi libero\r\n
        congue                nibh Morbi ac
ante\r\n
        nunc commodo                  turpis
tortor. Condimentum
   </p>
```

white-space value effects

	Source space/tab sequence	Source break line	Insert break lines to wrap in box.
normal	collapse	ignore	yes
pre	honour	honour	no
nowrap	collapse	ignore	yes
pre-wrap	honor	honor	yes
pre-line	collapse	honor	yes

See C:/Data/Sda/Code/Web/Examples/CssExamples/white-space.html

In practice, at 05 Jan 20100 and given that IE7 doesn't support pre-wrap, you will either want to use 'pre' (and format your source code correctly) or no style at all.

Bentley

Aphorism (Tips and Best Practice)

Move and size only block-level boxes.

Inline-level boxes are cross browser buggy and not meant to contain other shit.

Bentley

Build and test your CSS in the most advanced browser available before testing in others, not after.

Dave Shea > CSS Crib Sheet

Don't use quotation marks around paths/URLs.

When setting a background image, or loading in an imported file, resist the urge to surround the path with quote marks. They're not necessary, and IE5/Mac will choke.

Dave Shea > CSS Crib Sheet

It's quite common to fix the width of the navigational column in pixels so that it is not affected by the browser resizing.

Elizabeth Castro > XHTML, 5th > p220

Template

```
/*
  Ensure the following ...
  Character encoding: UTF-8
  Byte Order Mark (BOM): None.
  Charset at-rule: None.
*/
/*
    CSS Template Version 1.3
    Last Updated: 2011-01-04 16:23
    By John Bentley
*/

/*
    We don't use the consistent font size hack. We just let the font
    appear larger in bad browsers like IE5. This means we can have valid css
*/

/*****************************************************************************
    Hack Name:  Inheritance deeper than table hack.
    Remarks:    In IE5win many text and font properties are not inherited
                from the body into the table elements.
                Is valid CSS.

    Tested:     IE6win, Moz1.6, IE5win
    Fixes Bad Browsers:
                IE5win
    Source:     John Bentley
*/
body, td, th
{
    font-size: small;
}
/********************************** End Hack ******************************/

body
{
    margin: 0;
    padding: 0;
    font-family: 'Verdana Ref', Sans-Serif;
}

@media print
{
    body
    {
        margin: 0;
        padding: 0 5mm;
    }

    body, td, th
    {
        font:  12pt garamond, serif;
    }

    /* for Mozila */
    li
    {
        font:  12pt garamond, serif;
    }

    blockquote
    {
        font:  12pt garamond, serif;
    }

    thead,  tbody
    {
        background-color: #fff;
    }

    em
    {
        margin-left: 0.3em;
    }

    #navigation, #validators
    {
        display: none;

    }
}
```

See Also

⇒ John Bentley > hacks.css
A list of common CSS hacks
C:\Data\Dev\Code\Web\Libraries\CssLibrary\hacks.css

⇒ John Bentley > CSS Layout Principles
C:\Data\Dev\SDA\Web\KB\CSS\Technique\CSS Layout Principles.doc

⇒ John Bentley > Character Sets and Encoding in XHTML and CSS Quick Reference
C:\Data\Dev\SDA\Web\KB\Common\Character Sets and Encoding in XHTML and CSS Quick Reference.doc

⇒ John Bentley > Markup & CSS Code Convention - Naming and Style
C:\Data\Dev\SDA\Web\Method\Markup & CSS Code Convention - Naming and Style.doc

Sources

⇒ Mike Hall > Using Style Sheets http://www.brainjar.com/css/using/default4.asp
Cited: 13 Jul 2004

⇒ Mike Hall > CSS Positioning
http://www.brainjar.com/css/positioning/
Cited: 13 Jul 2004

⇒ Dave Shea > CSS Crib Sheet
http://www.mezzoblue.com/archives/2003/11/19/css_crib_she/?cssfile=/css/radar-sm.css
Cited: Jul 2004

⇒ Koch > BC
Peter-Paul Koch
CSS contents and browser compatibility http://www.quirksmode.org/css/contents.html Cited: 01 May 2004

⇒ Elizabeth Castro > XHTML, 5th
HTML for the world wide web with XHTML and CSS
Published: 2003

⇒ W3C > Character Sets Tutorial
Tutorial: Character sets & encodings in XHTML, HTML and CSS (DRAFT)
http://www.w3c.org/International/tutorials/tutorial-char-enc/#choosing
Cited: 27 Jul 2004

Document Licence