CSS2.1 Quick Reference Card1.01

ExplainThat!™

http://www.explainth.at

Color key overleaf

Code Structure

A cascading stylesheet is a collection of one or more rules bearing the form

selector{**property1**:***value***;...;**propertyN**:***value***}

The part within braces constitutes the declaration block consisting of one or more property specifications separated by a semicolon.

CSS comments bear the form /*Comment Text*/. Comments can span multiple lines and can be placed before, between or within a rule.

Use the **@import url(urlname)** directive to import stylesheets. To import stylesheets from another Internet location the full path to the url must be stated.

Spaces are generally ignored. If relevant, e.g. in font-family specification, the text should be quoted.

Selector Types

Class Selectors:**.name** – applies to HTML document elements (**HDE**s) which have the attribute **class**=”**name**”.

ID Selectors:**#name** – applies to all **HDEs** which have the attribute **id**=”**name**”.

Element Selectors:**tag** – where *tag* is a recognized HTML element type such as **h1, p, a** etc.

Selector types can be combined to deliver very targeted styling effects. For example

.name#ID – applies to all **HDEs** with the attributes **class**=”**name**” and **id**=”**ID**”.

h1.name – applies to all **<h1>** elements with the attribute **class**=”**name**”.

h1#ID – applies to all **<h1>** elements with the attribute id=”**ID**”.

.name tag – applies to all **<tag>** elements (.e.g. **<h1>, <a>**) inside a block element with attribute **class**=”**name**”.

.name>tag – applies to all **<tag>** elements **directly** owned by a block element with the attribute **class**=”**name**”. ~~**IE**~~.

Pseudo-Classes

:link – to style unvisited hyperlinks, **<a>**.

:visited – to style hyperlinks that have been visited.

:hover – to style HDEs while the mouse hovers over them. Only supported by hyperlinks in **IE6**.

:focus – to style form elements, e.g. **inputs**, while they are focused. ~~**IE**~~.

:active – to style HDEs while they are being used. e.g. buttons & anchors while they are being clicked. ~~**IE6(7?)**~~.

:first-child – applies to the first child of another element. e.g. to style the first paragraph inside a division you would define **p:first-child**. ~~**IE6**~~

:hover must be specified **after** **:visited** for it to have an effect.

Grouping & Nesting

When defining two or more nearly similar rules do this

.rule1,.rule2...{**property1**:***value***;...;**propertyN**:***value***}
.rule2{**property3**:***value***;...}

The first line defines a **group** of rules sharing identical property values. With this done, override properties that differ as done above for **rule2**.

Descendant Nesting: **.name selector** styles all elements that use **selector** and are owned by block elements with the attribute **class**=”**name**”.

Child Nesting: **.name>selector** styles all elements that use **selector** and are **directly** owned by block elements with the attribute **class**=”**name**”. ~~**IE**~~

CSS in HTML

External CSS

<link rel=stylesheet href=”/styles/screen.css” type=”text/css” [media=”screen”]>

Embedded CSS

<style type=”text/css” [media=”screen”]>
/*Style declarations here*/</style>

Good practice requires providing styles for screen and print media as a bare minimum. Multiple media types should be specified as a comma separated list. Other media types include all, aural, braille, embossed, handheld, print, projection and tv.

Inline CSS

<tag style=”...”> where **tag** is any legal HTML element tag such as **h1**, **p**, **a** etc. Use single quotes in the style assignment if required. Follow the **property1**:***value***;.. format described above. Use sparingly.

Length Units

Absolute Units: in|cm|mm|pt|pc.

px (Pixels) as defined is a relative unit but as used by browsers is an absolute unit.

Relative Units: em|%. Measurements are relative to the parent element.

Lengths, except when 0, must **always** be followed by a unit. No space allowed between the number & the unit.

Color Units & Names

#RRGGBB is, generally speaking, the most economical way of specifying color. e.g. #C515BE.

Color Names: aqua|black|blue|fuchsia|gray|green|lime|maroon|navy|olive|purple|red|silver|teal|white|yellow

Background+1

background-attachment:scroll|*fixed*|*inherit*

background-color:transparent|<C>|*inherit*

background-image:*none*|*url(<urlname>)*

background-position:**[[<L>|*left*|*center*|*right*] [<L>|*top*|*center*|*bottom*]]|[[*left*|*center*|*right*]||[*top*|*center*|*bottom*]]** L is an offset from top-left. Default **0 0**

background-repeat:no-repeat|*repeat*|*repeat-x*|*repeat-y*|*inherit*

Border+23

border-?-color:color|*<C>*|*transparent*|*inherit*

border-?-style:none|*hidden*|*dotted*|*dashed*|*solid*|*double*|*groove*|*ridge*|*inset*|*outset*|*inherit*

border-?-width:*medium*|*thin*|*thick*|*<L>*|*inherit*

? is one of **left**, **top**, **right** or **bottom**

Color

color:*inherit*|*<C>*

Default depends on the browser. Also sets border color unless overridden.

Cursor

cursor:auto|*default*|*pointer*|*crosshair*|*move*|*?-resize*|*groove*|*text*|*wait*|*help*|*progress*|*inherit*

where **?** = *n*|*e*|*w*|*s*|*ne*|*nw*|*se*|*sw*

Use to assign cursor shape when mouse pointer enters the **HDE**. What “enter” means is browser dependent.

Dimensions

Dimension specifications only make sense for

- positioned **HDEs**, i.e. **HDEs** which have **position** set to a value other than static
- floated **HDEs**, i.e. **HDEs** which have **float** set to ***left*** or ***right***, in which case a **width** must be specified.

bottom|top|left|right:*auto*|*<L>*|*inherit*

top & **bottom** and **left** & **right** form pairs. If both are set to ***auto***, both are zero. If one is set to ***auto***, it is the negative of the other. If both are assigned a length and one is not the negative of the other, the browser may, at its discretion, ignore the setting.

width|height:*auto*|*<L>*|*inherit*

width & **height** are only relevant for block **HDEs**.

max|min-?:*<L>*|*inherit* where **?** = **width|height**. The **max** properties are 0 by default. The **min** properties have no initial value. Excess content spills out of the block by default. Assign **overflow** to avoid this. ~~**IE6**~~

Display4

display:*none*|*inline*|*inline-block*|*block*

Determines the type of display box generated by an **HDE** during page layout. The default value is **HDE** dependent **<span>** and **<a>** **HDEs** for instance are ***inline*** while **<div>** and **<p>** **HDEs** are ***block***s. Floating an **HDE** sets **display:*block*.**

Inline **HDEs** only take up as much horizontal space as required for displaying their contents. Block **HDEs** take the available client area of their container – unless **width** or **max-width** require otherwise.

inline-block causes the **HDE** to behave like an ***inline*** element while still using all block like attributes such as margin and padding. For compatibility with Firefox use

display:*-moz-inline-stack*;display:*inline-block*

display:*none* hides the **HDE** and generates no display block for it. Other **HDEs** are laid out as though it were not there.

Float

float:none|*left*|*right*|*inherit*

Causes **HDE** to “float” to the left|right of its container if empty or to the right|left of its previously floated sibling if non-empty. Floated **HDEs** are always blocks. The **width** property must be specified. If available client width is insufficient floating begins afresh from the appropriate boundary of the container - below previously floated siblings.

Assign **clear:**none|*left*|*right*|*both* to force subsequently floated **HDEs** to start afresh from the appropriate boundary of the container – a sort of “carriage return” directive.

Font5

font-family:inherit|*[*<F>*,]*<F>*[,*<G>*]*

font-size:*inherit*|<L>

font-family:*inherit*|*italic*|*normal*

font-weight:*inherit*|*normal*|*bold*

font:*inherit*|[[*<font-style>* || *<font-weight>*] *<font-size> <font-family>*]|*caption*|*icon*|*menu*|*message-box*|*small-caption*|*status-bar*

List+

list-style-image:*none*|*url(urlname)*|*inherit*

list-style-position:*outside*|*inside*|*inherit*

list-style-type:*disc*|*circle*|*square*|*decimal*|~~*decimal-leading-zero*~~|*lower-roman*|*upper-roman*|~~*lower-greek*~~|~~*lower-latin*~~|*inherit*

Margin+6

margin-?:*auto*|*<L>*|*inherit*

? is one of **left**, **top**, **right** or **bottom**

Overflow

overflow:*auto*|*visible*|*hidden*|*scroll*|*inherit*

Padding+

padding-?:*<L>*|*inherit*

? is one of **left**, **top**, **right** or **bottom**

padding:*<L>*|*inherit*

Position

position:*static*|*relative*|*absolute*|*fixed*|*inherit*

Absolutely positioned **HDEs** have as their container the nearest ancestor that has a **position** attribute other than ***static***. If no such ancestor exists, the container is the initial containing block – for all intents the browser window.

Fixed **HDEs** always have the viewport as their container. Not very useful until **IE6** becomes obsolete.

Absolute **HDEs** define their own stacking context This

CSS2.1 Quick Reference Card 1.01

means that **z-index** settings for their children are internal to the **HDE**.

Space

letter-spacing:*normal*|<*L*>

word-spacing:*normal*|<*L*>

Use negative values for effects such as overlap.

white-space:*normal*|*pre*|*pre-wrap*|*pre-line*|*inherit*

Stacking Order

z-index:*auto*|<*N*>|*inherit*

The **+ve** z-axis is orthogonal to the screen/paper in the direction of the viewer. Use values separated by 10 or more to simplify future page redesigns.

In the absence of a **z-index** specification, **HDEs** are displayed in the order of their occurrence in the document – subject to the **z-index** specification of any absolutely positioned container.

Text

text-align:*left*|*right*|*center*|*justify*

Default is browser and locale dependent

text-decoration:*none*|[*underline*||*overline*||*line-through*]

text-indent:|<*L*>|*inherit*

Indents first line of block **HDEs**. Use negative values to outdent.

text-transform:*none*|*lowercase*|*uppercase*|*capitalize*|*inherit*

vertical-align:*baseline*|*sub*|*super*|*top*|*text-top*|*middle*|*bottom*|*text-bottom*|<*L*>|*inherit*

Relative **<*L*>** values refer to the **line-height** of the element itself. Use **-ve <*L*>** values for subscripting. **+ve** values are an offset from the line top while **-ve** values are an offset from the line bottom.

line-height:*normal*|<*N*>|<*L*>|*inherit*

<*N*> = scaling factor. Use **<*N*> < 1** or **<*L*> < 100%** to compress lines in block **HDEs**. Defines **minimum** line-height. Maximum is determined by inline elements, such as images.

Visibility

visibility:*inherit*|*visible*|*hidden*|*collapse*

hidden **HDEs** are not visible but still take up space and affect the layout of the document. ***collapse*** causes table rows/columns to be hidden.

Element Types

Replaced elements are rendered by the browser using information not available in the document itself. Examples are images, **<img>**, and inputs, **<input>**. Nearly all other elements are **non-replaced** , i.e. their content is available in the document itself and is displayed in a box specified by the element attributes.

Block elements generate a box that, unless otherwise specified, occupies the full client width of the parent. Examples: **<div>**, **<p>** & **<li>**. The vertical margins of adjacent blocks collapse – i.e. only the bigger of the two is used. This does not apply to elements that are blocks by virtue of being **float**ed.

Inline elements generate a box which only consumes as much horizontal space as is required to display its contents. Inline elements do not use vertical margin settings. Examples: **<a>**, **<span>**, **<b>** etc.

Block elements can act as containers for other elements

Element types can be changed by setting the **display** attribute. ***inline-block*** (***-moz-inline-stack***) offers a useful half-way house between ***block*** and ***inline***.

Box Model

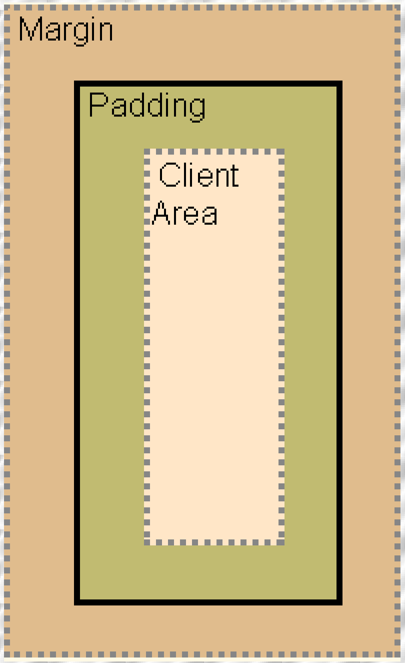
HDEs are rendered by browsers in a notional box made up of the following

1. The margin –
    1. vertical, where relevant.
    2. horizontal, always
2. The padding
3. The border
4. The client area

The precise manner in which CSS **height/width** settings are used is browser and DTD dependent

With the strict DTD the figures reported by Javascript for **e.clientWidth/Height** include the padding but not the border. **e.offsetWidth/Height**, on the otherhand, reports figures that include both the border width and the padding. Padding is nevertheless **outside** the client area – i.e the available area for child **HDEs** is **CSS height x CSS width**.

With the transitional DTD nothing changes – with Firefox and Opera. However, **IE** (even **IE7**) treats the CSS settings as the **whole** area of the element – border and padding included. The padding and border eat into the client area.



Notes

Tested under IE6+, Firefox 1.5+ & Opera 9.1+.

Most CSS properties are not inherited. Notable exceptions are **color**, **font**, **letter-spacing**, **word-spacing**, **text-align**, **text-indent** & **line-height**.

CSS properties describe **HDE** qualities. In other words, they are always adjectives, not verbs. Thus **visibility:*hidden*** **not** **visibility:*hide***.

Key

rule – CSS Selector

property – CSS Property Identifier

value – CSS Property Value

value – CSS Property Value (Default)

attrib– HTML Element Attribute Identifier

value – HTML Element Attribute Value

a|b – a or b

a||b – a or b or both, any order

a* - a zero or more times

IE – not supported by **IE**, even **IE7**

IE6(7) – not supported by **IE6**. **IE7** support incomplete.

IE6 – not supported by **IE6**

IE – not supported by **IE**, even **IE7**

IE – not supported by **IE6**.

[...] - option

[a b] – group of **a** and **b**.

<*text*> - replace ***text*** with suitable value

C – color value, e.g. fuchsia or #FF00FF

F – Family-name, e.g. arial, "comic sans ms" etc. Names containing spaces should be wrapped in quotes.

G – Generic-family name = **serif|sans-serif|cursive|fantasy|monospace.** Optional but recommended.

L – length value with unit, e.g. 2em, 10% 30px etc

N – simple number (no unit).

[+] – accepts shorthand. Shorthand properties are specified by combining the individual property specifications and assigning them to the root property, e.g. **border:** or **font:**. **Note:** Omitting a sub-property in shorthand specification causes it to take its default value. With the notable exception of **font:**, shorthand properties can be specified in any order.

[1] **IE6** bug:***20% right*** will move background right. Firefox and Opera will use ***20%*** and ignore ***right***.

[2] There are five distinct shorthand formats for the border property

- **border-?:** where ? is one of left, top, right or bottom
- **border:**

[3] for consistency specify a border width in length units. Each browser has its own interpretation of **medium**.

[4] only a partial list

[5] **Not** a copy of CSS standards. Use relative units for font sizes. **Always** specify font information – the defaults depend on, user-configured, browser settings. Other font-weights are often not available in the selected font. The shorthand, **font**, property requires the sub-property **order** given here to be followed. The system settings options, such as ***menu*** and ***caption***, are handy when creating dialogs which feel like system dialogs.

[6] The default is zero. Assign shorthand **margin** first, then change others individually - if required. Vertically adjacent block **HDEs** collapse – i.e. only the bigger margin is used. Inline **HDEs** do not use vertical margins. Horizontal margins never collapse. Nor do any of the margins of floated **HDEs**. To center **HDE** horizontally in its parent, set horizontal margins to ***auto***. Margins can be negative – use with care for overlap effects.

An extensive range of free quick reference cards is available at http://www.explainth.at