CSS cheat-sheet notes (quick reference)

Selectors

Basic

```
h1
   select all <h1> tags
h1.special
   select h1 tags of the special class
h1#special
   select h1 tags of special id

p a
   select <a> tags descendant from p

h1, h2
   select both h1, h2

*
   select all tags
implicit in
   .class = *.class
   #class = *#class

#banner *
   select all tags inside banner

a:link
   selects link that guest hasn’t visited
   regular style

a:visited
   link that visitor has clicked before

a:hover
   link when mouse is hovered over

a:active
   link when it’s clicked (just a ms)

.class1.class2
   select elements that have both class1 and class2
```
Pseudo-classes

generate content
:before
  inserts content before
  p.tip:before {content: "HOT TIP!" }

:after
  like :before but after

properties
  generated pseudo-classes can be styled like any other
display
  color
  border

content
  string
    content: "(link)"

urls
  content: url(path/to/image)

attributes
  content: attr(href)

combinations
  content: " [" attr(href) "]"

:first-child
  select and format just the first child of an element
e.g.,
  li:first-child { font-weight: bold; }
  formats all first elements in bold

:focus
  when element receives focus (e.g., click or tab)
e.g.,
  input:focus { background-color: #FFFFCC; }

Advanced selectors

child selectors
  body > h1
  select any h1 that is a child of body

adjacent siblings
  h2 + p
  select p right after h2

attribute selectos
  img[title]
select only img tags with "title" attribute

```html
.photo[title]
```

select any element with photo class and a title attribute

```html
input[type="text"]
```

input fields of type text

```html
a[href][title]
```

select a that has both href and title attrs

```html
*[title]
```

any element that has title attribute

```html
img[title~="Figure"]
```

any img with title that contains Figure in it
regexp match?

```html
img[title="bar"]
```

any img with title that starts with "bar"

```html
img[title$="bar"]
```

any img with title that ends with "bar"

```html
img[title*="bar"]
```

any img with title that contains substring "bar"

```html
*[lang|="en"]
```

any element with lang attr that equals en
begins with en-

### Pseudo-classes

- `:first-line`
- `:first-letter`
  - just the first letter
  - can only be applied to block elements (not inline)

### Pseudo-elements

- `:before`
- `:after`

# examples

```css
h2:before {content: ""]"; color: silver;}
body:after {content: " The End.";}
```
Floats

containing block
nearest block-level ancestor

floated element generates a block box
regardless of its type
will be laid out as a float

Rules

1) stay within borders of containing element
   left/right outer edge may not be to the left of inner left/right containing block.

2) give precedence to existing floats
   prevent floats from overwriting each other  floats are safe
   left outer edge must be to the right of the right outer edge of a preceding element.

3) no overlapping between float elements
   if they collide the loosing element will be floated down

4) can’t be higher than inner top (pre-padding) of containing parent

5) floats can’t be higher than the tops of preceding floats
   If we have 3 floats, and the first two are floated to the left, the third float will float only as
   high as the second float.

6) floats can’t float higher than the top of the line box generated by a preceding element

7) floats get pushed down to a new line height if there isn’t room for them in the containing
   box

8) given the above constraints, float as high as possible

9) left floating elements try to float as left as possible, right floating elements try to float as
   right as possible
   BUT
   a higher position is preferred to floating more right

Rule consequences

when floated element is taller than container?
the bottom sticks out of the container

countermeasure
floated element will expand to contain floated
descendants

negative margins
can cause floats to appear to escape their parents

4
just like negative margins on non-floated elements can make the child appear wider than it’s parent

if floated element is wider than it’s parent it can stick out

negative margins may cause a float to overlap inline elements inline elements rendered on top of the float

a float expands to contain anything in it

move to the left or right edge of the browser window (or containing window)

floated inline methods are treated like blocks

content with a background or border runs underneath the float how to prevent this - overflow: hidden

clear property don’t wrap around a floated item left - drop below left floated items right - drop below right floated items both - drop below both

Positioning

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

static
generated as normal

relative
offset by some distance
element retains it’s shape
space it would ordinarily have occupied preserved

absolute
removed from the flow

positioned with respect to its *containing* block may be another element in the document

space the element might have occupied is closed up as if the element didn’t exist

positioned element generates a block-level box even if it’s inline

fixed
\[ \text{element box behaves like absolute} \]
but containing block is the viewport itself

containing block

in HTML the root element is HTML
some browsers use body
initial containing block
rectangle the size of the viewport

nearest ancestor (of any kind) that has position value != static
block-level containing ancestor
padding edge (I.e., border bounded area)

inline-level containing ancestor
content edge of the ancestor
in ltr lang - top left corner, right bottom corner

no ancestors?
initial containing block

important - elements can be positioned outside containing block
maybe it should be called "positioning context" instead

offset properties

top, right, bottom, left := <length> | <percentage> | auto | inherit

percentage
containing width for left / right
height for top / bottom

auto
element position if it were static

positioning can cause height/width to be calculated automatically
e.g.,
height 100%
top: 0
bottom: 0

width 100%
left: 0
right: 0

margins apply to the position boundaries
you're really specifying the position of the outer edge

setting width/height can make a difference
IF you set borders, padding, margin
because width/height set inner edges
the content box

fixed position
can be used to simulate frames of old
just like absolute position except
the containing block is the viewport

Properties

formatting text
line-height
letter-spacing # how much space to add between letters

font-family
sans # clean and simple appearance (for headlines)

  Arial
  Helvetica
  Verdana
  Tahoma
  Formata
  Sans

serif # better for long passages of text
  Georgia
  Times
  Times New Roman

popular combinations
  Arial, Helvetica, sans-serif
  "Times New Roman", Times, serif
  "Courier New", Courier, monospace
  Georgia, "Times New Roman", Times, serif

font size
keywords
  xx-small x-small small medium large x-large xx-large

  each increase or decreases by 1.2

  ems and percentages the same thing

font styles
  font-style: italic|bold|normal
  text-transform: uppercase
  font-variant: small-caps

  text-decoration: underline|overline|line-through|blink

spacing attributed
  letter-spacing
  word-spacing
  line-height
  normal setting is 120%
alignment
text-align: left|center|right|justify

text-indent
e.g., text-indent: 3em
indent first line 3em
text-shadow
<color> <offset-x> <offset-y> <blur-radius>?
text-shadow: green 5px 0.5em;
not supported in Firefox

white-space
normal | nowrap | pre | pre-wrap | pre-line | inherit

normal := discards extra whitespace
collapses multiple "spaces"

pre := treated as if it's pre
whitespace is not ignored

pre-line := whitespace collapsed, linefeeds honored

pre-wrap :=
like pre except lines wrap

no-wrap := prevent wrap breaks from being rendered
line breaks must be inserted with <br />

lists
list-style-type: square|disc|circle
list-style-position: outside|inside

margins and padding
margin: 5px
5px on all sides
margin: 5px 1px
5px top and bottom
1px on the sides

margins between elements don’t get added
the larger of the two margins is applied

padding is added
whenever vertical margins touch the margins collapse

inline vs block
display: inline|block;
inline elements don’t get any taller with padding or margins except for img tags

colors
rgb(30%, 30%, 30%)
#FFF #FFFFFF

names
aqua
fuchsia
lime
olive
red
white
black
gray
maroon
orange
silver
yellow
blue
green
navy
purple
teal

border
each side can have a different property

width / height
em is the text size
percentages = size of containing element

max-width
min-width

max-height
min-height

width/height do not include borders, padding and margins just the content

displayed width = width + left padding + right padding + left border + right border + left margin + right margins

height property
dangerous because it’s hard to know in advance the size of content in a box

amount of margins between paragraphs, headlines varies from browser to browser
for consistent predictable results set this yourself

background images

background-image: url(images/bg.gif);
url is relative to the stylesheet not the HTML page
root relative
absolute

background-repeat
repeat,
normal setting, repeat both on x and y
repeat-x
repeat-y

no-repeat,
displays the image a single time

background-position
precise values
<distance-from-left> <distance-from-top>

keywords
<horizontal> <vertical>
<horizontal> := left|center|right
<vertical> := top|center|bottom
bottom is the bottom of the content
not necessarily the bottom of the page

e.g.,
background-position: left center

percentages

short hand
background := <background-color> <background-image>
<background-attachment> <background-position>

it’s possible to replace borders with hand-drawn lines

overflow := hidden | visible | scroll | auto
visible - normal setting what browsers usually do
scroll - add scroll bars
auto - make scroll bars optional
hidden - hides extended content
clipping can provide precise control over hidden content overflows

visibility := visible | hidden | collapse | inherit

in invisible state the element is still there
you just can’t see it
it effects layout normally
descendant of invisible element can be visible

Outlines
Like borders except:
  • they don’t cause reflow: good for adding an outline to emphasize focus
  • they can be non-rectangular
  • they can invert the background
Syntax:
  outline := <outline-color> || <outline-style> || <outline-width>

  outline-width := thin | medium | thick | <length>
  only one (unlike borders with top bottom left right)
  outline-style := dotted | dashed | solid | double | groove | ridge | inset | outset

  outline-color := <color> | invert | inherit

Tips
  separation of concerns
  HTML should describe structure
    don’t use class names that describe style

  CSS should describe styling

  start with an inline style sheet
    after perfection move CSS code to an external sheet

  title attribute adds tooltips to images and links

  don’t use div when you can use HTML tags

  classes vs ids
    classes for repeating elements
    id selectors for unique elements

    ids get priority

  margin: 0 auto
    lets content float
    equivalent to <center>

  margins collapse

  line-height creates implicit padding
Workflow

put temporary code on top of the stylesheet
makes it easiest to work with firebug

name classes by structure, not styling
separation of concerns
  HTML for content structure
  CSS for styling

refactor
  eliminate code
  CSS
  HTML
  let the future take care of itself

use advanced selectors (instead of tons of classes)
use more tag types when available
  not everything should be a div

minimize reloads
  edit CSS mode
  better than editing on server

  copy to server at the end
  refactor in gvim

debugging
  draw borders to better understand box model

use two firebug windows
  one for selecting
  one for editing live

Misc

element types
  replaced vs non-replaced
  whether or not the content is included in the HTML
  most html tags non-replaced
  replaced tags
    img
    input tag

block vs inline
  block
  by default
  fills its parent element
  can not have other elements at its sides
  I.e., generates a break before and after box
e.g.,

```html
p
div
li
```

special case that generates marker

```html
inline
generated line boxes that are flowed within parent
```

internal style sheet
```
<style type="text/css"></style>
only in the head
```

inline style sheets
no time and bandwidth saving

class names
letters, numbers, hyphens and underscores
must start with a letter
case sensitive

terminology
ancestor
descendent
parent
closest ancestor
child
closest descendent
siblings

inheritance
not passed
border properties
placement properties
margins
paddings

Specificity of CSS properties

highest specificity wins

weights
tag selector = 1
class selector = 10
ID selector = 100
inline style = 1000

overruling specificity
!important after any property

```css
a { color: teal !important; }
```
organizing styles and stylesheets
name styles by purpose not appearance
refactor common styles to separate classes
combine multiple classes

group styles
apply to related parts of a page
group styles with related purpose

xhtml
differences from HTML
lowercase tags
quotation marks required for xhtml
all tags must be closed
<br />

validator.w3.org
tags
p # for paragraphs
li
dl # definition list
dt # definition term
blockquote # quotes
q # one line quotes
cite # referencing
address # identify and supply contact info
table
tags
caption
attribute align=top|bottom
colgroup
col
thead
tr
th
tbody
tr
td

cell properties
text-align
vertical-align top|baseline|middle|bottom

border-collapse: collapse|separate