CSS3

CS 146
Intro to Web Programming and Project Development

Basics of CSS

- Cascading Style Sheets are a way to define the appearance of elements in your page.
- Can be written in your page, or saved in an external file (.css) and then included in each one of your pages.
- Saves tons of time: Write your code once, and update your style sheet if you want a new look!

Levels of CSS

- CSS1 was released by the W3C in 1996.
  - It included the core capabilities associated with CSS, such as the ability to format text, set fonts, and set margins.
- The CSS2 spec came out in 1998.
  - It included all the attributes of CSS1, ways to position elements, and had increased emphasis on international accessibility and the capability to specify media-specific CSS.

CSS3

- CSS3 has been split into "modules". It contains the "old CSS specification" (which has been split into smaller pieces). In addition, new modules are added.
- Some of the most important CSS3 modules are:
  - Selectors
  - Box Model
  - Backgrounds and Borders
  - Image Values and Replaced Content
  - Text Effects
  - 2D/3D Transformations
  - Animations
  - Multiple Column Layout
  - User Interface

Validating CSS Code

- Use the W3C Validation Service:
  - http://jigsaw.w3.org/css-validator/

CSS Syntax

- selector {
  ```
  selector {
          property: value;
          /* comments */
          otherproperty: value;
  }
  ```
- Example:
  ```
  p {
          color: red;
          text-align: center;
  }
  ```
- Note that simply writing a name of tag will select every tag with that name.

Selectors

- There are MANY selectors, and even more since the introduction CSS3, but we will start with the basics.
- You can select anything with an id by doing #nameofid
  - #para1 {text-align: center; color= red;}
- You can select anything with a "class" attribute by doing .nameofclass
  - .myclass { text-align: left; }
- You can also combine tag name and class by doing things like
  - p.myclass {text-align: right; }
  - This will only select paragraphs that also have the myclass class.

Using CSS in Your Site

- External Style Sheets: add this code in your <head> </head>
  - <link rel=“stylesheet” type=“text/css” href=“filename.css” />
- Internal Style Sheet: add the style sheet in the head
  - <style type=“text/css”>
    /* your style code */
    </style>
- Inline: add the style in the tag itself
  - <p style=“color:red;”>red paragraph</p>

What if you have multiple styles?

- The style "cascades" from lowest to highest priority in this order
  - Browser's default
  - External style
  - Internal style
  - Inline style
- Don't use inline styles, if possible. They have the highest priority and you will have to edit the html files to change the style.
- It is much cleaner and more flexible to use external style sheets!

Styling Backgrounds

- background-color: #FF0000
- background-image: url(‘file.gif’)
- background-repeat:
  - repeat, repeat-x, repeat-y, no-repeat, inherit
- background-attachment:
  - scroll, fixed, inherit
- background-position:
  - [left right center] [top center bottom]
    - e.g. left center
  - x% y% (from 0% 0% top left to 100% 100% bottom right)
  - xpos ypos (in pixels)
- You can combine in one line:
  - background {background: #ffffff url(‘file.png’) no-repeat right top;}

Styling Text – CSS1, CSS2

- color:
  - #FF0000 or #F00
  - rgb(255,0,0)
  - red
- text-align: center, right, left, justify
- text-decoration: none, overline, line-through, underline
- text-transform: uppercase, lowercase, capitalize
- text-indent: 25px; (indents first line)
- letter-spacing: positive or negative value
  - -3px
- vertical-align:
  - length or percentage (negative OK)
  - baseline, sub, super, top, text-top, middle, bottom, text-bottom, inherit
- white-space: normal, nowrap, pre, pre-line, pre-wrap
- word-spacing: positive or negative value

Styling Text in CSS3

- text-shadow:
  - horizontal, vertical, blur distance, color
    - text-shadow: 5px 5px 5px #FF0000;
- word-wrap:
  - normal or break-word

STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY®

Working with Fonts

- Font can have two possible font family names
  - generic family (Serif, Sans-serif, Monospace)
  - font family
  - You can use multiple types in order of preference. First one that could work is used
  - font-family: “Times New Roman”, Verdana, sans-serif;
  - Quotes are only needed for fonts with more than one word in it.
  - PLEASE NEVER USE COMIC SANS! No one will take you seriously!
- font-style: normal, italic, or oblique
- font-variant: normal or small-caps
- font-weight: normal, bold, bolder, lighter
  - You can also use a number 100 to 900 (400 is normal, 700 is bold)
- font-size
  - px: pixels
  - em: value based on the width of the uppercase M whatever typeface is used
  - rem: root “em”

Details about px

- The CSS px unit does not equal one physical display pixel. This has always been true – even in the 1996 CSS 1 spec.
- CSS defines the reference pixel, which measures the size of a pixel on a 96 dpi display. On a display that has a dpi substantially different than 96dpi (like Retina displays), the user agent rescales the px unit so that its size matches that of a reference pixel. In other words, this rescaling is exactly why 1 CSS pixel equals 2 physical Retina display pixels.
- That said, up until 2010 (and the mobile zoom situation notwithstanding), the px almost always did equal one physical pixel, because all widely available displays were around 96dpi.

Details about em

- 1 em is 16 pixels.
- Sizes specified in ems are relative to the parent element. This leads to the em's "compounding problem" where nested elements get progressively larger or smaller. For example:

```
body { font-size: 20px; }
div { font-size: 0.5em; }
```

Gives us:

```
<body> - 20px
  <div> - 10px
    <div> - 5px
      <div> - 2.5px
        <div> - 1.25px
```

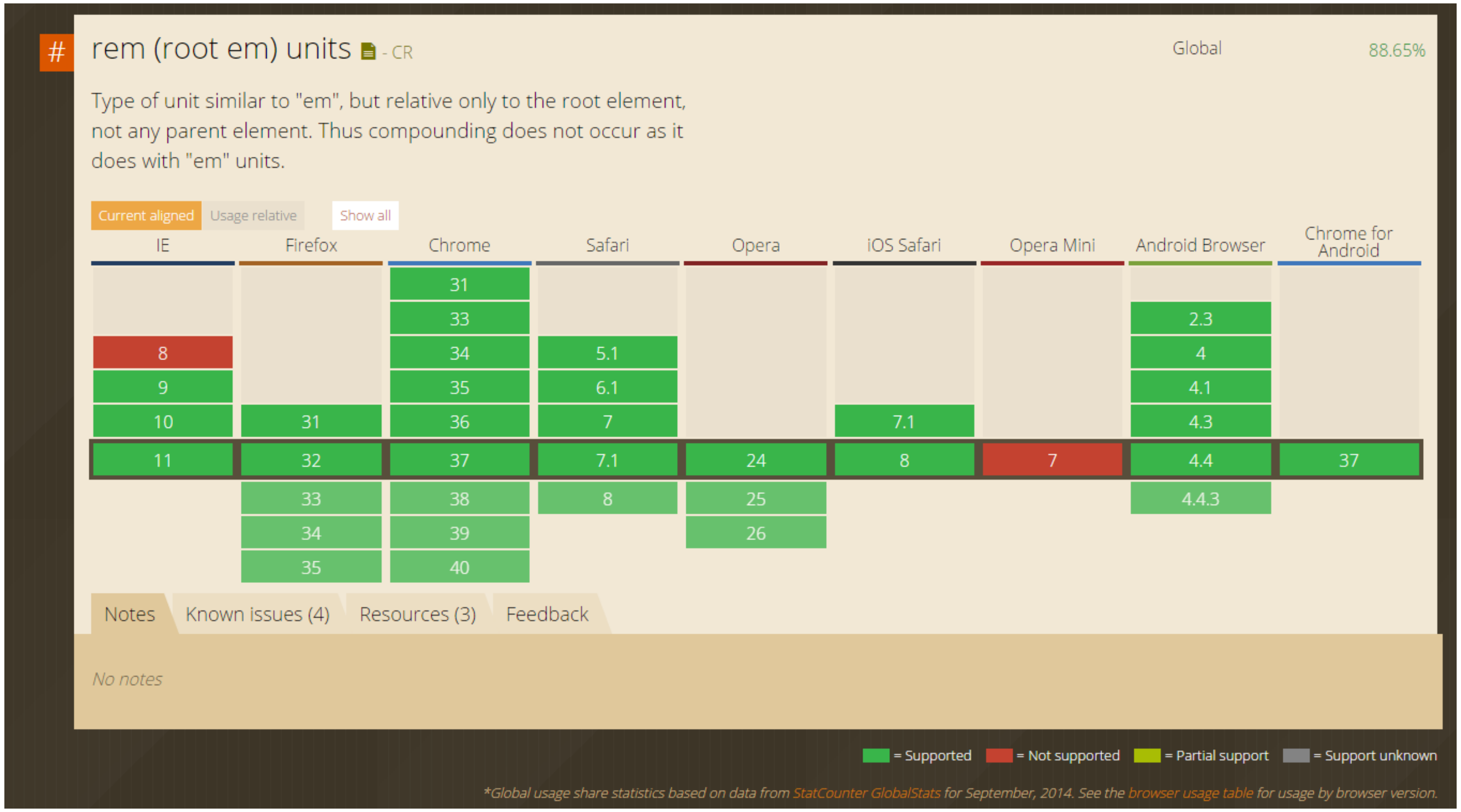
Details about rem

- The rem unit is relative to the root—or the html—element. That means that we can define a single font size on the html element and define all rem units to be a percentage of that.

```
html { font-size: 62.5%; }
body { font-size: 1.4rem; } /* =14px */
h1   { font-size: 2.4rem; } /* =24px */
```

Can I Use Rem?

- Sure, check with caniuse.com/#feat=rem



rem (root em) units - CR

Global 88.65%

Type of unit similar to "em", but relative only to the root element, not any parent element. Thus compounding does not occur as it does with "em" units.

Current aligned | Usage relative | Show all

IE	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Chrome for Android
		31						
		33					2.3	
🟥 8		34	5.1				4	
9		35	6.1				4.1	
10	31	36	7		7.1		4.3	
11	32	37	7.1	24	8	🟥 7	4.4	37
	33	38	8	25			4.4.3	
	34	39		26				
	35	40						

Notes | Known issues (4) | Resources (3) | Feedback

No notes

🟩 = Supported 🟥 = Not supported 🟨 = Partial support ⬜ = Support unknown

**Global usage share statistics based on data from StatCounter GlobalStats for September, 2014. See the browser usage table for usage by browser version.*

CSS3 Revolutionizes Fonts!

- Up until CSS3 you could only use web-safe fonts.
- You can now use ANY font you want
  - as long as you have a True Type Font or Embedded Open Type file
- First you define a new font in your style with @font-face
- @font-face {

```
@font-face {
        font-family: theNameYouWant;
        src: url('filename.ttf'),
        url('filename.eot') format("opentype");
}
```

- Then simply use "theNameYouWant" as font-family
- Other font optional descriptors
  - font-stretch: [ultra-, extra-, semi-]condensed, [ultra-, extra-, semi-]expanded, normal
  - font-style: normal, italic, oblique
  - font-weight: normal, bold, 100, 200, …, 900

Styling Links

- Links have 4 different states, all of which can be changed
  - a:link (standard)
  - a:visited (visited link)
  - a:hover (mouse over link)
  - a:active (selected link)
- If you are defining these, keep them in that order! That's the rule!

Styling Lists

- Use classes to make different lists, this way you can use ul.name or ol.name for different styles
- list-style-type:
  - For unordered use circle, disc, none, square
  - For ordered, options galore! armenian, cjk-ideographic, decimal, decimal-leading-zero, georgian, hebrew, hiragana, hiragana-iroha, katakana, katakana-iroha, lower-alpha, lower-greek, lower-latin, lower-roman, upper-alpha, upper-greek, upper-latin, upper-roman
- list-style-image: url('filename.gif');
- list-style-position: inside or outside

IE and Opera Strike Back

- When using images for lists, they display a bit off in “certain” browsers
- You can “fix it” like this

```
ul {
    list-style-type: none;
    padding: 0px;
    margin: 0px;
}
li {
    background-image: url(‘file.gif’);
    background-repeat: no-repeat;
    background-position: 0px 5px;
    padding-left: 14px;
}
```

Styling Tables

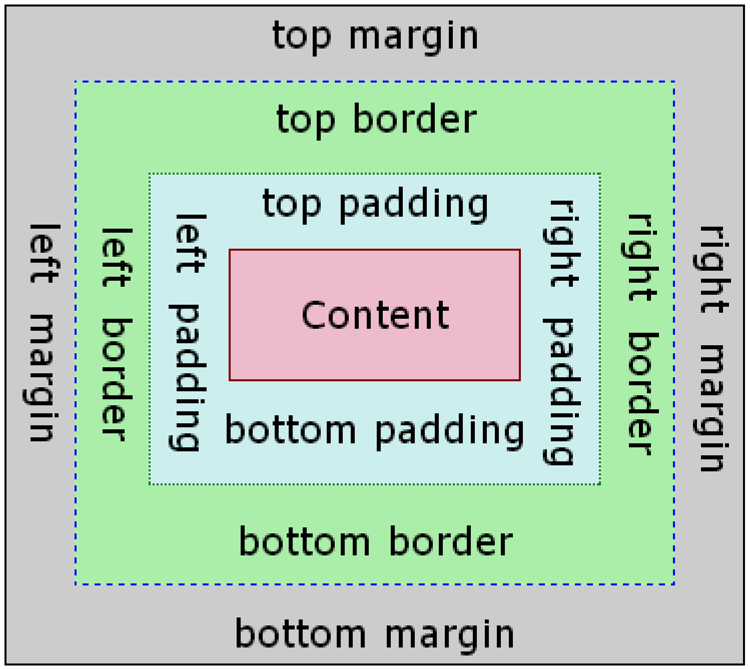
- You can assign certain styles to more than one item

-
  ```
  table {
      border-collapse: collapse;
      width: 100%;
  }
  table, th, td {
      border: 1px solid black;
  }
  td {
      height: 50px;
      padding: 15px;
  }
  ```
- You can also set captions with the <caption> tag which can be style as caption-side: bottom; for example.

CSS Box Model

- Every HTML item is considered a box with
  - Content: where text and images appear.
  - Padding: clear area around the content, affected by background color.
  - Border: goes around padding & content, also affected by background color.
  - Margin: clear area around the border, always transparent.
- When setting size understand the four regions!
- `width:250px; padding:10px; border: 5px solid gray; margin: 10px;`
  - will have an actual width of 300
  - (250 + 2*10 + 2*5 + 2*10)

CSS Box Model

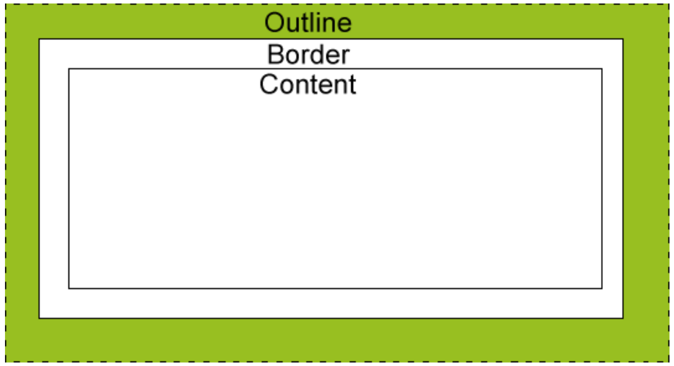


Box Properties

- border-style: none, dotted, dashed, solid, double, groove, ridge, inset, outset
- border-width: can be a number of pixels or thin, medium, thick
- border-color: same as usual
- You can set side individually by doing
  - border-top-style, border-right-style, border-bottom-style, or border-left-style
- You can write in one line
  - border-style: *top right bottom left*;
  - border-style: *top right&left bottom;*
  - border-style: *top&bottom right&left;*
  - border-style: *allborders;*

CSS Outlines

- An outline is a line that is drawn around elements (outside the borders) to make the element "stand out".
- However, the outline property is different from the border property.
- The outline is not a part of an element's dimensions; the element's total width and height is not affected by the width of the outline.
- Properties are the same, just replace border with outline
  - outline-color
  - outline-style
  - outline-width



CSS Dimension

- The CSS dimension properties allow you to control the height and width of an element.
  - height
  - max-height
  - max-width
  - min-height
  - min-width
  - width

Visibility

- You can hide items with
  - display:none
    - The item will not take any space
  - visibility:hidden
    - Item won’t be visible but will take up space
- display can also be set to
  - inline
    - Use for example on lists, to have them shown in a single line
  - block
    - To have an element as a single block

Positioning

- Static positioning (default)
  - Position decided by the normal flow of the page, unaffected by top, bottom, left and right
- Fixed Positioning
  - Position relative to browser window
- Relative Positioning
  - Relative to its normal position (can take negative values)
- Absolute Positioning
  - Relative to the first parent element that is not static (or <html> if no such parent)
- If you have overlapping, you can use z-index to determine the layers order (higher number means top most)

Positioning

- top, right, left & bottom: (auto, length, %)
- clip: rect(top, right, bottom, left)
  - Allows to cut an absolutely positioned element
- overflow: (auto, hidden, scroll, visible)
- position: (absolute, fixed, relative, static)
- z-index: (number, auto)

Floating Items

- Use to have elements placed in different locations relative to the currently present ones. Take one out, the rest shift
- Elements float horizontally and float as far left or right as it can
  - img {float: right; }
- Floating an image right when it is followed by text, will place the image to the right of the text with text wrapping that image
- It can be used to automatically place images in a grid that adapts to the size of the window
  - .thumbnail
    {float: left; width: 100px; height: 75px; margin:2px;}
- To avoid text wrapping automatically, use
  - clear: both; (other values are left, right, none, inherit)

Alignment

- You can center an item by setting auto margins left & right. Auto will split them evenly

```
.center {
   margin-left: auto;
   margin-right: auto;
   width: 60%;
}
```

- To align things to the right, you can do it with position:absolute; right: 0px;
- You can also use float to place items to the right.

CSS Cursor Property

- cursor:
  - *url('image.gif'), url('something.cur'), auto;*
  - crosshair
  - default
  - {n, ne, e, se, s, sw, w, nw}-resize
    - For example se-resize
  - help
  - move
  - pointer
  - progress
  - text
  - wait
- Try it out: http://www.w3schools.com/cssref/pr_class_cursor.asp