Cascading Style Sheets, CSS

BWA Slides #4

Cascading Style Sheets, CSS

Language for describing the presentation semantics of markup documents

- To produce a (nice) 2D view of the DOM tree (XML, HTML, XHTML, ...)
- Possible to set styles (fonts, colors, backgrounds,...)
- Possible to do layout (positioning)

Cascade

- There can be many CSS involved: author, user, user agent (browser), may overlap !!
- The cascade resolves the interaction
- Command "!important" can change precedence

Why CSS?

Prior to CSS, nearly all of the presentational attributes of HTML documents were contained within the HTML markup

- Duplicate code
- Unmaintainable

CSS allows authors to move much of that information to a separate style sheet resulting in considerably simpler HTML markup

CSS have had severe problems (getting better)

- Browser incompatibility
- Bad specifications
- Bugs

CSS Specification

W3C specification

- http://www.w3.org/TR/CSS/ (a wrapper specification)

CSS2.1 Candidate recommendation February 25, 2004, final on 7 June 2011!!

- Single specification (all before 2.1 deprecated)

CSS3

- Modularization of CSS2.1
- Expands features in CSS2.1
- Things previously done in code (JavaScript) now possible in pure CSS (menus)
- Not fully implemented in any browser (?)

We use CSS 3

CSS Language Basics

CSS program in *.css file

Program contains a list of statements

- We say **rule** (simplified)
- White space may surround statements

Illegal parts ignored (check spelling)! Possible not accepted by browser

Case <u>insensitive</u>

Comments: /* */   (not nested)

CSS Rules

Rule ::= Selector Declaration block

- declaration block = { declaration; declaration; ... }
- declaration = property name ":" property value

```
/* A rule */
h1 {                      /* Selector, block start */
    font-size: 34px;   /* Declaration */
    font-weight: bold;
 }                        /* Block end */
```

CSS Selectors

Selectors can be
- Standard HTML element, p, div …
- Element attribute, id, class
- .. and much more ("pattern matching" using the above)

See http://www.w3.org/TR/CSS2/selector.html

Important: Will be used later (JQuery)

CSS Values, Units and Inheritance

Property values examples

- Length: 0.5 em (em = relative font-size for element)
- Length: 12 px (pixel = absolute value, avoid)
- Percentages: line-height: 120%, relative to some other value for element (depends on property)
- URI: background: url("yellow.gif"), relative or absolute (avoid)
- Colors: {color: olive}, {color: #f00} (#rgb), {color: #ff0000 } (#rrggbb), { color: rgb(255,0,0) },{ color: rgb(100%, 0%, 0%)
- Strings: " " or ' '

Elements inherit values so em or % can be relative to parent

CSS Styles

Add declarations for font, colors, borders, backgrounds etc. to the rules

- NetBeans have CSS style builder (had at least?)

NOTE: <u>Graphical design and typography normally not a topic for programmers</u>

Many new fancy possibilities in CSS 3

- Shadows
- Rounded corners
- ...

CSS Layout and Positioning

The CSS box model

- All elements in HTML-DOM surrounded by a 2D box, see http://www.w3.org/TR/CSS2/box.html

Some elements have a **block box** i.e (<div>'s)

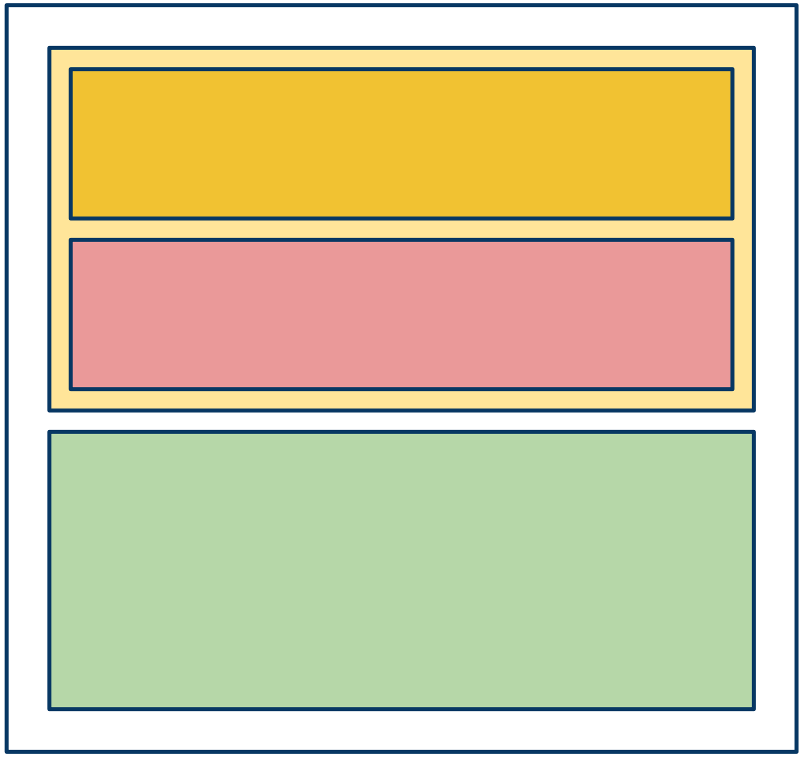
- Element that takes up the full width available
- Has a line break before and after

Others have an **inline box**; not full width, no breaks

Normal Flow Layout

If no CSS, browser uses **normal flow** for layout

```
<div id="white">
    <div id="yellow">
        <div id="orange">
        </div>
        <div id="red">
        </div>
    </div>
    <div id="green">
    </div>
</div>
```



CSS Positioning

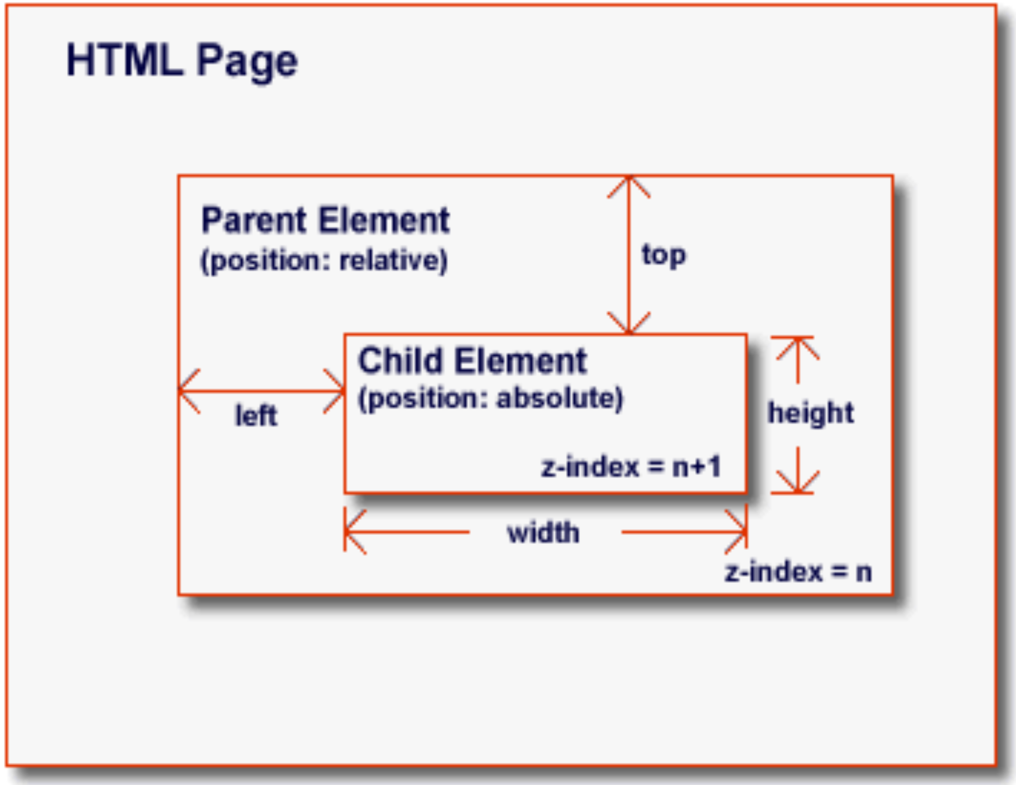
How boxes are laid out relative other boxes (containing). Complex (see spec).

- Relative, offset from normal flow
- Absolute positioning. Removed from normal flow, positioned relative first non-static positioned parent (if none, browser window)
- Fixed, fixed relative browser window http://www.brainjar.com/css/positioning/

Floats (floating boxes)

- Shifted to the left or right on the current line (if no room shifted downwards)
- Content flows down sides of float (prohibit with "clear") Must have width set

CSS Positioning cont.



CSS Positioning

CSS3 Advanced

Previously need of JavaScript replaced by advanced CSS3 features

- 2D/3D transformation

Modularization

Stylesheets grow, sometimes need for modularization

- Possible to break up style sheets and import

```
<!-- In HTML -->
<link rel="stylesheet" type="text/css" href="/css/styles.css" />

// In styles.css (first in file)
@import url('/css/typography.css');
@import url('/css/layout.css');
@import url('/css/color.css');
```

Adding CSS to HTML

Browser will download (and cache) and apply style sheet

HTML/XHTML

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

XML

```
<?xml-stylesheet type="text/css" href="mystyle.css"?>
```

Responsive Web Design

Hard to know on with devise the page will be displayed (size, etc). Using http://www.w3.org/TR/css3-mediaqueries/ we can tailor different CSS to match different devices

A media query consists of a media type and zero or more expressions that check for the conditions of particular media features.

CSS3 Media Queries

```
<!-- Use in HTML, if device passes width test then
use shetland.css else skip -->
<link rel="stylesheet" type="text/css"
  media="screen and (max-device-width: 480px)"
  href="shetland.css" />

// Use in CSS
@media screen and (max-device-width: 480px) {
  .column {
    float: none;
  }
}

// Use in import in CSS
@import url("shetland.css") screen and (max-device-width: 480px);
```