CS 696 Emerging Web and Mobile Technologies
Spring Semester, 2011
Doc 6 CSS3
Feb 3, 2011

References

HTML5 & CSS3 Develop with Tomorrow's Standards Today, Hogan, Pragmatic Programmers

Various Web pages as indicated on individual slides

New/Improved Features in CSS3

Borders
border-color
border-image
border-radius
box-shadow

Backgrounds
background-origin and background-clip
background-size
multiple backgrounds

Color
HSL colors
HSLA colors
opacity
RGBA colors

Text effects
text-shadow
text-overflow
word-wrap

User-interface
box-sizing
resize
outline
nav-top, nav-right, nav-bottom, nav-left

Selectors
attribute selectors

Basic box model
overflow-x, overflow-y

Generated Content
content

Other
media queries
multi-column layout
Web fonts
speech

Sources: http://www.css3.info/preview/

Some Pseudo-classes

:nth-of-type	p:nth-of-type(2n+1){color: red;}	Finds all n elements of a certain type.
:first-child	p:first-child{color:blue;}	Finds the first child element.
:nth-child	p:nth-child(2n+1){color: red;}	Finds a specific child element counting forward.
:last-child	p:last-child{color:blue;}	Finds the last child element.
:nth-last-child	p:nth-last-child(2){color: red;}	Finds a specific child element counting backward.
:first-of-type	p:first-of-type{color:blue;}	Finds the first element of the given type.
:last-of-type	p:last-of-type{color:blue;}	Finds the last element of the given type.
:after	span.weight:after { content: "lbs"; color: #bbb; }	Used with content to insert content after the specified element.

last-childExample

```
tr:nth-last-child(2){
  color: green;
}
```

```
td:last-child{
  font-weight: bolder;
}
```

Item	Price	Quantity	Total
Coffee mug	$10.00	5	**$50.00**
Polo shirt	$20.00	5	**$100.00**
Red stapler	$9.00	4	**$36.00**
		Subtotal	**$186.00**
		Shipping	**$12.00**
		Total Due	**$198.00**

```
tr:last-child{
  font-weight: bolder;
}
```

```
tr:nth-last-child(-n+3) td{
  text-align: right;
}
```

```
tr:last-child td:last-child{
  font-size:24px;
}
```

IE again

IE 8 and earlier does not support css selectors

So another workaround selectivizr.js

http://selectivizr.com/

Use with jQuery or other library

```
<script type="text/javascript" src="[JS library]"></script>
<!--[if (gte IE 6)&(lte IE 8)]>
  <script type="text/javascript" src="selectivizr.js"></script>
  <noscript><link rel="stylesheet" href="[fallback css]" /></noscript>
<![endif]-->
```

Media Queries

Select CSS based on media features

width
height
device-width
device-height
orientation
aspect-ratio
device-aspect-ratio
color
color-index
monochrome
resolution
scan
grid

Media in link

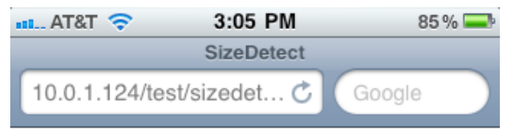
```
<!DOCTYPE html>
<html lang="en">
<head>
      <meta charset="utf-8" />
      <title>SizeDetect</title>
      <link rel="Stylesheet" href="ipad.css" type="text/css"
    media="screen and (max-device-width: 768px)"/>
  <link rel="Stylesheet" href="phone.css" type="text/css"
          media="screen and (max-device-width: 320px) "/>
</head>
<body>
<p>Red = phone, Blue = iPad, Black = desktop</p>
</body>
</html>
```

ipad.css

```
p { color: blue; font-size: 3em; }
```

phone.css

```
p { color: red }
```



Red = phone,

max-device-width gives an upper bound on the actual device-width. Any device with a smaller width will match so the iphone matches max-device-width: 768px. The last style sheet that matches overrides properties set in previous one

Device-width

```
<link rel="Stylesheet" href="ipad.css" type="text/css"
    media="screen and (device-width: 768px)"/>
```

device has exact width 768px

```
<link rel="Stylesheet" href="ipad.css" type="text/css"
    media="screen and (min-device-width: 700px)
    and (max-device-width: 768px"/>
```

device has width between 700 and 768px

px = CSS pixels not physical pixels

In CSS file

example.css

```
@media (device-width: 768px) { p {color: blue; font-size: 3em; }
}

@media (device-width: 320px) { p {color: red; }
}
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <title>SizeDetect</title>
    <link rel="Stylesheet" href="example.css" type="text/css"/>
</head>
<body>
<p>Red = phone, Blue = iPad, Black = desktop</p>
</body>
</html>
```

How to detect iPhone 4

In CSS

@media (-webkit-min-device-pixel-ratio: 2) { p {color:yellow}}

iPhone 4 has a high resolution screen. People want to identify it to use better images, etc.

Orientation

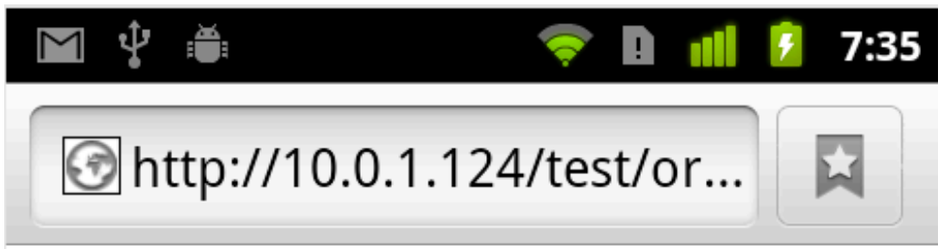
```
<!DOCTYPE html><html lang="en">
<head>
	<meta charset="utf-8" />
	<title>Orientation</title>
	<link rel="Stylesheet" href="orientation.css" type="text/css"/>
</head>
<body>
<p>Red = landscape, Blue = portrait</p>
</body>
</html>
```

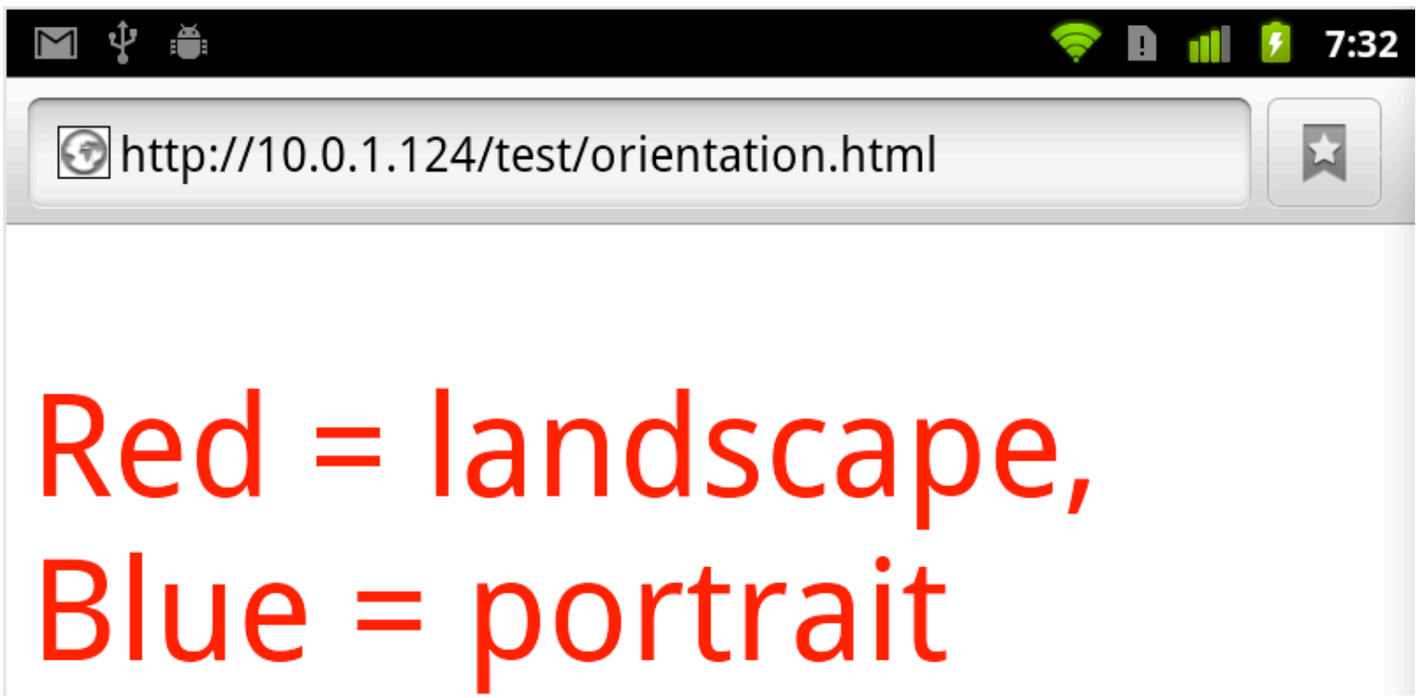
orientation.css

```
@media (orientation: landscape) { p {color: red; font-size: 3em; }
}

@media (orientation: portrait) { p {color: blue; }
}
```

Android


7:35
http://10.0.1.124/test/or...
Red = landscape, Blue = portrait


7:32
http://10.0.1.124/test/orientation.html
Red = landscape,
Blue = portrait

Loading CSS based on user agent

```
<!DOCTYPE html><html lang="en">
<head><meta charset="utf-8" /><title>SizePlatform</title>
    <script lang="javascript">
    var loadCSS = function(file) {
        var link = document.createElement('link');
        link.href = file;
        link.rel = 'stylesheet';
        link.type = 'text/css';
        document.getElementsByTagName('head')[0].appendChild(link); };
    </script>
</head>
<body onload="
    if (navigator.userAgent.match(/iPhone/i)) { loadCSS('iphone.css');
  } else if  (navigator.userAgent.match(/iPad/i)) { loadCSS('ipad.css')
    } else if  (navigator.userAgent.match(/Android/i)) { loadCSS('android.css')
    }">
<p>Red = phone, Blue = iPad, Android = Yellow</p>
</body>
</html>
```

User agent

```
<!DOCTYPE html><html lang="en">
<head>
    <meta charset="utf-8" />
    <title>User Agent</title>
</head>
<body onload="alert(navigator.userAgent)">
<p></p>
</body>
</html>
```

One Css file, Different CSS classes

```
.ipad p { color: blue; font-size: 3em; }
.iphone p { color: red;  }
.android p { color: yellow;  }
```

One Css file, Different CSS classes

```
<!DOCTYPE html><html lang="en">
<head>
    <meta charset="utf-8" />
    <title>CSSClass</title>
    <link rel="Stylesheet" href="joint.css" type="text/css"/>
    <script type="text/javascript">
        var cssclass = "desktop";
        if (navigator.userAgent.match(/iPhone/i)) {
            cssclass="iphone";
      } else if  (navigator.userAgent.match(/iPad/i)) {
            cssclass="ipad";
        } else if  (navigator.userAgent.match(/Android/i)) {
            cssclass="android";
        }
        document.documentElement.className += ' ' + cssclass;</script>
</head>
<body>
<p>Red = phone, Blue = iPad, Android = Yellow</p>
</body></html>
```

Android User Agent

Android supports many screen sizes

100+ Android tablets announced in January

Screen size may be more important than Android user agent

Loading different pages in JavaScript

```
if((navigator.userAgent.match(/iPhone/i)) ||
        (navigator.userAgent.match(/Android/i)) ||
        (navigator.userAgent.match(/iPod/i)))
    {
    location.replace("http://sitename.com/m/");
  }
```

A poor way do it as requires two round trips to server

http://stackoverflow.com/questions/2254558/android-emulator-browser-detecting

Apache

in .htaccess

```
#redirect mobile browsers
RewriteCond %{HTTP_USER_AGENT} ^.*iPhone.*$
RewriteRule ^(.*)$ http://mobile.yourdomain.com [R=301]
RewriteCond %{HTTP_USER_AGENT} ^.*BlackBerry.*$
RewriteRule ^(.*)$ http://mobile.yourdomain.com [R=301]
RewriteCond %{HTTP_USER_AGENT} ^.*Palm.*$
RewriteRule ^(.*)$ http://mobile.yourdomain.com [R=301]
```

This way there is only one round-trip to server

source: http://dev-tips.com/featured/redirect-iphone-blackberry-palm-requests-with-htaccess

Web Accessibility

Web Accessibility

http://www.w3.org/WAI/

Designing Web sites that are flexible to meet different user needs, preferences, and situations

May not be able to see, hear, move

May not have or be able to use a keyboard or mouse

Provide equivalent alternatives to auditory and visual content.

```
<img src="foo" alt="this is am image of a Foo Bear" border="0" align="middle">
```

Use alt tag in img, input, applet

Screen readers read the alt tag

Don't rely on color alone

Ensure that text and graphics are understandable when viewed without color

Use markup & style sheets and do so properly

Control presentation with style sheets rather than
with presentation elements and attributes

Don't use h1 tag just to change font

Create tables that transform gracefully

For data tables

Use TD to identify data cells and TH to identify headers

For data tables that have two or more logical levels of row or column headers

Use THEAD, TFOOT, and TBODY to group rows

Use COL and COLGROUP to group columns

Ensure that pages featuring new technologies transform gracefully

Organize documents so they may be read without style sheets

Ensure user control of time-sensitive content changes

Ensure that moving, blinking, scrolling, or auto-updating objects or pages may be paused or stopped

People with photosensitive epilepsy can have seizures triggered by flickering or flashing

Sample Accessibility Checker

http://achecker.ca/checker/index.php

For a list of tools see

http://www.w3.org/WAI/ER/tools/complete

HTML5 & Accessibility

Roles

Used by screen readers

```
<header id="page_header" role="banner">
<h1>AwesomeCo Blog!</h1>
</header>
```

Widget Roles

alert
alertdialog
button
checkbox
dialog
gridcell
link
log
marquee
menuitem
menuitemcheckbox
menuitemradio
option
progressbar
radio
scrollbar
slider
spinbutton
status
tab
tabpanel
textbox
timer
tooltip
treeitem
combobox
grid
listbox
menu
menubar
radiogroup
tablist
tree
treegrid

Document & Landmark Roles

Document Structure Roles

article
columnheader
definition
directory
document
group
heading
img
list
listitem
math
note
presentation
region
row
rowheader
separator
toolbar

Landmark Roles

application
banner
complementary
contentinfo
form
main
navigation
search