HTML & CSS: Week 5

June 18 - July 23, 2014

Instructor: Devon Persing

This week

- Layout wrap-up
- Web fonts
- Pseudo-selectors and pseudo-elements
- Embedding content and media
- Javascript and related topics
- Other odds and ends

Layouts continued

Responsive Design designed by Nithin Viswanathan from the Noun Project

{ } `float` layout review

- The **`float`** property lets us take elements out of the main flow of the page and put them to one side (**`left`**) or the other (**`right`**)
- Floats can be "cleared" with the **`clear`** property
- Floated elements get **`display: block;`** applied to them by default

{ } `inline-block` layout

- **`inline-block`** was designed to display text elements like links
- We used it to make a horizontal menu on our pages by applying **`display: inline-block;`** to our menu's **`li`** elements
- You can use **`inline-block`** for whole containers to make column layouts too

{ } `inline-block` layout fix

- **`Inline-block`** was designed for text and it adds a bit of space after each element for readability
- When using it for layouts, you can give your containers a negative right margin:

```
.section {
    display: inline-block;
    margin-right: -4px;
}
```

{ } Using the `position` property

- The **`position`** property lets us arrange elements:
  - In relation to the flow (**`relative`**)
  - In a very specific place outside of the flow or within another **`relative`** element (**`absolute`**)
  - In relation to the browser window (**`fixed`**)
- How **position** is applied depends on to where the element is in the flow by default

{ } Tweaking the `position`

- We can dictate where elements go down to the pixel
- **`left`**, **`right`**, **`top`** and **`bottom`** + or - pixels between positioned elements and their containers

```
div {
    position: absolute;
    right: -10px;
    top: 30px;
}
```

{ } Using `position: fixed;`

- **`Position: fixed;`** is a way to make content "stick" to the browser window, regardless of where the user scrolls
- Commonly used to make headers, navigation, or footers that follow the page as it scrolls

{ } Responsive web design

- Allows layouts to **adjust to the size of a device or browser window**
- Uses **% of the parent container** instead of fixed pixel widths
- We can use **CSS media queries** to call different styles based on the size of a user's device or browser window, along **breakpoints**

{ } `@media` queries

- Designed to use different styles based on the way content is being displayed
- Previously most commonly used to style web pages for print

```
@media all and (max-width: 520px) {
   /* styles for smaller devices */
}
```

{ } Media queries example

```
/* basic widths for larger browser window/screen */
main { width: 100%; }
.photo { width: 33.333333%; }

/* styles for smaller browser window/screen override
previous widths */
@media all and (max-width: 520px) {
   main, .photo {
      width: 100%;
      /* change other styles at different browser sizes!
      */
      background: #ccc;
      font-size: 1em;
   }
}
```

{ } `display: table;`

- Have a grid-like responsive layout?
- **`display: table`** works like a table container
- **`display: table-row`** works like a table row
- **`display: table-cell`** works like a **`<td>`** or **`<th>`**

{ } Best practices for responsive

- Floated layouts are the easiest to make "fully" responsive
- Absolute and fixed position layouts can break down on smaller screens
- **There are no perfect breakpoints**
- Change your layout when it starts to break or look broken!

Web fonts

{ } Freedom from Arial!

- Web fonts let us style sites with **fonts that users may not have** on their own device
- Web font services **licence fonts for online use** specifically
- Files are either:
  - hosted by a service
  - served with your pages

A note about licensing

- **Not all fonts can be used online**, even if you own their rights for print, they're in Adobe products, etc.
- Fonts with online licensing will come with **documentation saying so**
- **Exception:** If you own the rights to use a font with software, you can use it to make images that are published online

{ } Some web font options

- **Google Fonts** is free and hosted
- **TypeKit** (owned by Adobe) is hosted and subscription based or bundled with Creative Cloud
- **FontSquirrel** is free and not hosted
- **FontDeck** is subscription based and not hosted
- *And some others!*

CSS pseudo-classes and pseudo-elements

{ }

{ } Conditional `Pseudo-classes`

- **Pseudo-classes** are added to a selector to add conditional styles to an element
- Most often used to style **states** of **`<a>`** elements and form elements

```
a:link { /* the default state of a link */ }
a:visited { /* a link that's been clicked */ }
a:hover { /* a link that has a mouse hover */ }
a:focus { /* a link that has keyboard focus */ }
a:active { /* a link that is being clicked */ }
```

{ } `:hover` versus `:focus`

- **`:hover`** is for a link or other interactive element that has a **mouse hover**
- **`:focus`** is for a link or other interactive element that has **keyboard focus**
- Browsers have their own default **`:focus`** styles for **accessibility**

```
a:hover, a:focus {
/* it's good practice to style them together! */
}
```

{ } `:hover` for other elements

- **`:hover`** can be used to style hover states for some non-interactive elements to create a more dynamic experience

```
tr { /* a table row with one background... */
   background: #9f6;
}

tr:hover { /* ...could have another on hover */
   background: #f60;
}
```

{ } Some nifty pseudo-elements

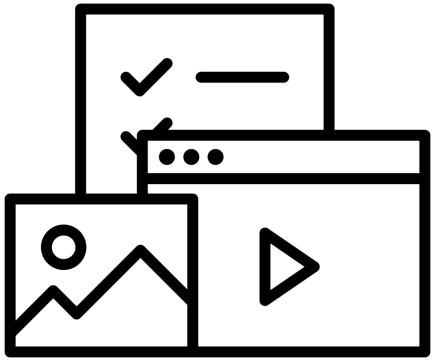
- **`:first-letter`** styles the first letter of a block of text
- **`:first-child`** and **`:last-child`** style the first and last children of a parent
- **`:nth-child()`** can be used to style even or odd children, or do some math to style every 5th, etc.
- **`:before`** and **`:after`** can be used to add style-only pseudo-content to elements

CSS selectors are evolving

- **Pseudo-classes**, **pseudo-elements**, **combinators**, and **attribute selectors** create extremely targeted ways to style content that degrade gracefully in older browsers
- To learn more of these techniques: **http://www.quirksmode.org/css/selectors/**

Pocket designed by Michela Tannoia from the Noun Project

Embeddable content



Media designed by Garrett Knoll from the Noun Project

<> Embedded content and media

- Embedded content is what it sounds like: content, usually media, that is embedded in our HTML page
- We already know one embeddable element: the **`<img>`** tag
- Probably the next most common type of embedded content is the **`<iframe>`**

<> `<iframe>` implementation

- Used to load content from **another HTML document** into an HTML page
- iframes have a **`src`** attribute
- Commonly used to:
    - Embed media (like YouTube videos)
    - Add **social widgets** (like the Facebook Like button)
    - Load 3rd party ads on a page

<> Good practice for `iframes`

- **Include fallback HTML** in case the iframe fails to load
- **Specify the iframe's dimensions** with CSS or HTML attributes

```
<iframe src="page.html" width="200"
  height="400">
If you can see this, your browser doesn't
  support iframes. <a href="page.html"
  >Here's a direct link to the content.</a>
</iframe>
```

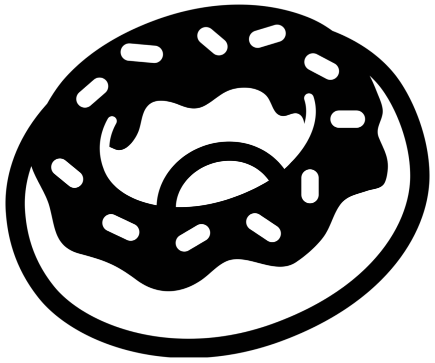
<> An example YouTube iframe

- There's very little reason to make your own iframes to include in your own pages since...you can just make your own content
- Let's drop a YouTube iframe into a page and look under the hood.

<> `<video>` and `<audio>`

- HTML5 introduced **`<video>`** and **`<audio>`** embeddable elements (and others)
- Adds **default playback controls** that can be managed with Javascript
- Can fall back to Flash media
- The current trend for "background" videos? Those are HTML5 videos!

Extra goodies



Donut designed by Jacob Halton from the Noun Project

Related topics

- Javascript
- CSS3 and CSS4
- Libraries and frameworks
- Mobile-first thinking
- Accessibility
- Version control

Javascript

- The third pillar of the web along with HTML and CSS
- Embedded into an HTML document with the **`<script>`** tag
- Allows for additional interactivity and data manipulation that isn't possible with HTML and CSS alone

Javascript use examples

- Hiding, showing, moving, etc., content based on user actions
- Displaying controls for HTML5 media
- Drawing content on the screen based on data (ex.: Chart.js)
- Collecting data about the type of browser, device, and internet connection a user has

Javascript libraries

- A set of **pre-made scripts**
- A platform for **common user interface patterns**
- Designed to work out of the box
- Designed to work with plugins and other libraries to provide extra functionality
- Probably the most common is **jQuery**

Javascript and CSS frameworks

- A set of **pre-made scripts and styles** for quickly prototyping or iterating on projects
- **Heavily tested** and **prevents having to roll your own** Javascript and styles to complete a common task
- Probably the most common is **Twitter Bootstrap**

CSS3 and CSS4

- **CSS3+4** techniques add extra **refinements**, **depth**, **transitions**, **animations**, **rotations**, and **typography**
- Frequently **combined with Javascript**
- Range from simple (rounded corners) to **full-blown interactive experiences** previously only possible with Flash or Javascript (ex: Animate.css)

Mobile and tablet-first

- Means thinking about scaling up using **progressive enhancement**
- **Defining the base experience that can work on a smartphone** and add enhancements to tablets, then laptops and desktops
- Only add bells and whistles **when a system can more easily support them**

Why think mobile-first?

- 20% of worldwide web usage is on mobile devices[1]
- Mobile usage for everything besides talking on the phone has tripled since 2011[2]
- 63% of adults in the US use their phones to use the internet[3]

1 Browser stats for Q4 [2013]
2 US Time Spent on Mobile to Overtake Desktop
3 PEW Internet: Mobile

Web accessibility (a11y)

- **Web accessibility** is about providing support for people in four major use cases:
  - Blindness and low vision or color-blindness
  - Deafness
  - Issues with motor skills
  - Cognitive/learning disabilities
- HTML, CSS, and Javascript can be written to support each use case

Developing for a11y

- **Logical content order** and **semantic elements**
- Media **alternatives** (ex.: **`alt`** attributes)
- **Keyboard** focus (**`:focus`**) and interactions
- Sufficient **color contrast**
- W3C's **[WCAG 2.0 guidelines]**

Version control for code

- **Version control** is a method of storing versions of files in a **repository**
- Helps **prevent** accidental deletions, additions, mistakes, and errors in live code for you
- **Tracks and manages conflicts** between files
- Common systems are **git** and **svn**

Version control integration

- Version control lets us (more) safely share code between developers and collaborate on projects
- Can be integrated into systems for deploying code onto live sites
- For example, the code for our class site is stored online in GitHub, and the site is served with GitHub Pages

What else?

Before we go...

1. Visit www.svcseattle.com/evaluation/
2. Choose **"HTML and CSS - Level 1 (Persing) (Summer 14)"** from the dropdown
3. Fill out the evaluation
4. Please be honest and constructively critical!

Thank you!

It was really fun.