JMS 302: HACKING THE MEDIA

CLASS 3: 8/26/2014

TODAY'S TASKS

- Web/Program Planning
- HTML Structure & Text
- Starting Ruby

Facebook group
Note on GitHub and notes.

REVIEW

- GitHub
- Your First Programs

Note: When adding directories, you have to manually add the directory via the command line. The GitHub app will not automatically recognize new directories.

Let's say you create a new directory (class_3) in exercises. Now you need to add it to GitHub. This will be a frequent task and also contains some of the command line commands that you should just know cold.

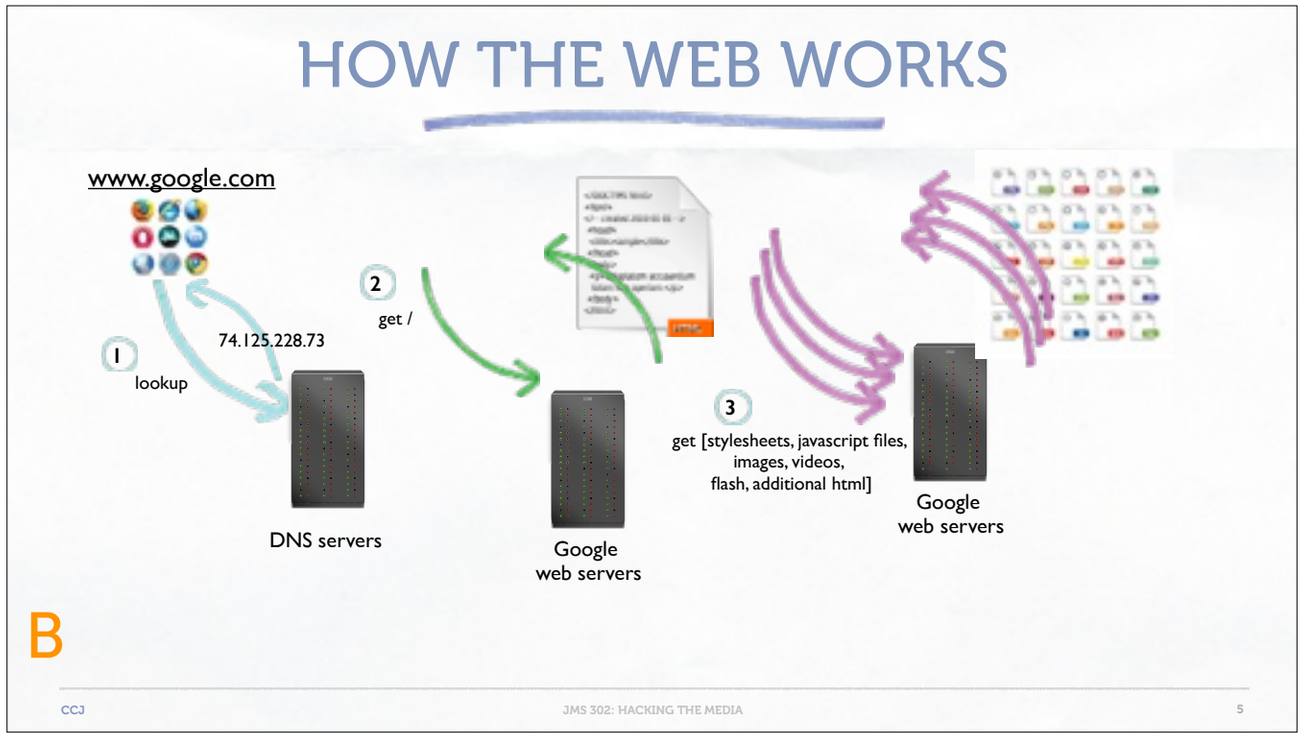
Basic idea: In terminal, find the directory (folder). You need to be in the new directory or the directory containing the new directory. Then you can add it to GitHub using the git command.
[Note: When I say type a command, assume you type a return or enter after the command. That makes it execute.]

1. Open iTerm
2. Type `pwd` and see that you're in your home director. Type `ls` and see the `JMS\ 302` folder
3. `cd` into the directory (don't forget filename completion using your tab key)
4. `cd` into `exercises`
5. If you haven't made the directory, type `mkdir class_3`
6. Two options (Tim Toady: TIMTOWTDI):
    1. From within `exercises`, type `git add class_3`
    2. `cd class_3` and then (after hitting enter) `git add .` (the period is crucial here)

Review your first programs.
Note, missing DOCTYPE.

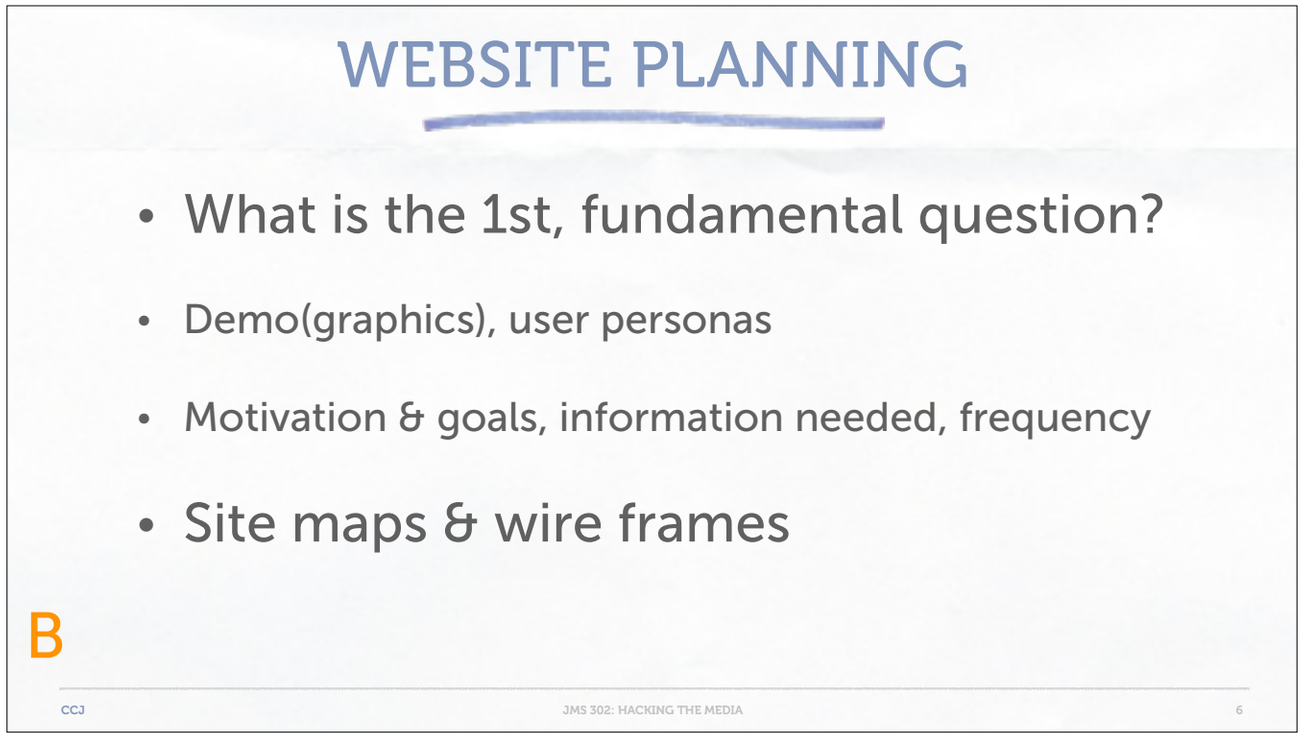
POP QUIZ

- What is the 1st question one should ask when planning a website? (B)
  - Who is the site for?
- How can you create visual contrast? (A)
  - proximity, white space, color, borders, etc.
- Name the parts of the following element: (C)
  - peach
    - element, opening tag, attribute name, attribute value, closing tag
- Name the tags that do the following (these are the key tags you should know) (C)
  - [Describe verbally]
    - body, head, title, headings, paragraphs, line breaks
    - strong (vs b), emphasis (vs i), horizontal rule, block quote
- script is a series of ____ (B)
  - [instructions]
- How does JS make a page more interactive (B)
  - access & modify content
  - program rules & react to events



Details in lecture. See also HTML text, pages 9 & 10.

Images from http://codeplanet.io/wp-content/uploads/2013/08/browsers.jpg, http://en.wikipedia.org/wiki/HTML and http://www.vectorstock.com/royalty-free-vector/computer-files-icons-vector-1082325.



Who is the site for?
May seem obvious, but many if not most start with what the creator (employee) wants.

See HTML text, chapter 18.

Example wire frames: https://medium.com/@citizenblr/balsamiq-101-wireframe-quickly-effectively-9231bb126d01

The next few slides will have some summarizing of the reading. I probably won't do this much summarizing, particularly of basic conceptual material. I will focus on the difficult content, as well as highlighting the things you really need to know to succeed.

Pay attention to the visual hierarchy and grouping on these sites. What stands out? What elements are similar to each other (and why might that be)? How are size, color and style used? Notice the simplicity—the organization and prioritization that had to go into this design.
- https://www.humin.com/#/product
- http://www.zendesk.com/beautifully-simple

That doesn't work for all sites. News sites are particularly tough, but these do a good job within their constraints.
- http://nytimes.com/ (compare navigation to the book's recommendations)
- http://guardian.co.uk/

But long-form content can be very elegantly presented (more readily).
- http://nautil.us/
- http://bittersoutherner.com/

HTML

B
- Structure, not layout. Why?

C
- Elements & tags, attributes
  - body, head, title, headings, paragraphs, line breaks
  - strong (vs b), emphasis (vs i), horizontal rule, block quote
  - view source, developer tools

See HTML text, chapters 1 & 2

- HTML for structure, CSS for layout.
  - Use CSS to describe how things look. HTML can do almost everything, but it's a bad idea.
    - Consistency, ease of updating, efficient file sizes, can split (human) responsibilities
    - General concept: separation of concerns (specialization)
  - General-purpose CMSs are really bad at this separation.
- Elements usually have opening and closing tags.
  - Sometimes you have self-closing tags (`<br />`).
  - Not closing tags is a common source of visual problems (but most browsers are very forgiving on closing self-closing tags).
  - **The convention for this class is to lowercase tag and attribute names and use quotation marks around attribute values.**

HTML HIERARCHY

MAKE THIS EVIDENT IN YOUR CODE

YOUR CODE EDITOR WILL HELP

```
<html>
 <body>
  <h1>This is the Main Heading</h1>
  <p>And we have a list below:</p>
  <ul>
   <li>list item #1</li>
   <li>list item #2</li>
  </ul>
 </body>
</html>
```


This is aesthetically pleasing and shows the few people who will look at your code that you care about your work. More practically, it helps you see the structure of your page and quickly find elements (particularly if you've left something unclosed). If you are not using a code editor that automatically closes and indents code for you, you will definitely come to appreciate this habit.

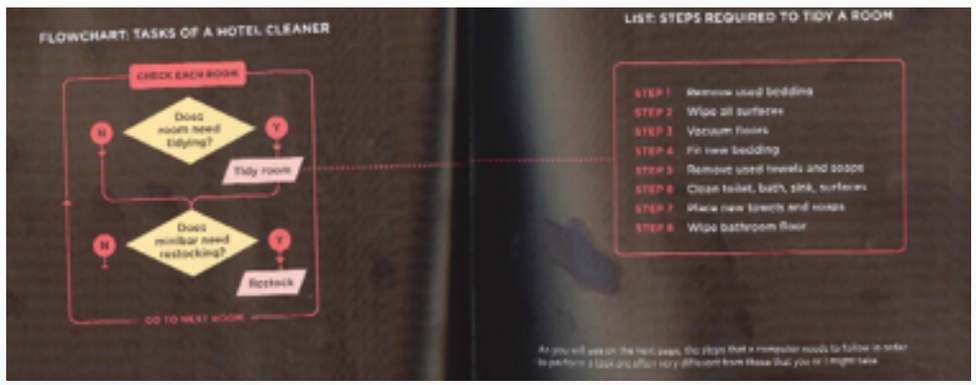
JAVASCRIPT

- Makes pages interactive
  - access & modify content
  - program rules & react to events
- Scripts
  - define, design, code
  - flow charts
- Language
  - vocab (keywords) and syntax (plus a word about logic errors)

A

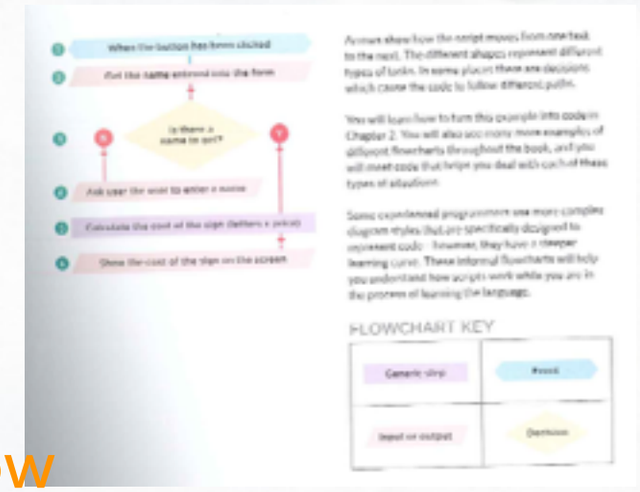
FLOWCHARTS

- www.gliffy.com



A

FLOWCHARTS



FYI for now

OBJECTS

- We'll cover in-depth later
- properties & methods
- events
- Document Object Model
- Progressive enhancement

FYI until later

MORE INFO/PRACTICE

- User Experience

  http://www.uxapprentice.com/

- Official HTML spec:

  http://www.w3.org/TR/html5/

- Useful reference & guides for HTML:

  https://developer.mozilla.org/en-US/docs/Web/HTML

- Code Academy for HTML:

  http://www.codecademy.com/tracks/web

FYI

Notice the browser compatibility section of the Mozilla reference.

ASSIGNMENT

- Site map & homepage wireframe
- MyBalsamiq.com
- GitHub submission

1. Create a trial account at MyBalsamiq.com. Later, I'll invite you to join group projects, which will be free.
2. Create your site map and a home page.
3. Download the images and BMML files for both, name appropriately, place in your week_1 folder and commit to GitHub.
4. Create HTML for the homepage using the main tags you've learned so far. The HTML doesn't have to match your mock-up (at this point), and it can have placeholder (garbage) content. Put the HTML in your week_1 folder and commit.

More info
You can build this as a sight for the unblight conference or as a resource for combating blight (housing data, code enforcement, neighborhood revitalization). Later, you'll get specific guidance (user personas, motivation & goals, etc.).

Unblight conference info
- http://unblight.org
- http://sunlightfoundation.com/blog/2014/07/14/register-for-unblight-a-community-unconference-on-housing-data-august-14-15/
- http://www.macon.com/2014/08/17/3253368/unblight-conference-marshals-attention.html
- http://www.macon.com/2014/08/17/3254913/conference-explores-multi-layered.html

A couple of sites from attendees:
http://localdata.com/
http://596acres.org/