2

Web Theory

Building Things for the Web

A flower does not think about competing with the flower next to it. It just blooms.

- Anonymous

Contents

How To Use This Document

Bits of text in red are links and should be clicked at every opportunity. Bits of text in `monospaced green` represent code. Most the other text is just text: you should probably read it.

Some sections are marked "Read-Only": these are sections that are intended to be read through in your own time and will not have a corresponding lecture.

Copying and pasting code from a PDF can mess up indentation. For this reason large blocks of code will usually have a View Code link above them. If you want to copy and paste the code you should follow the link and copy the file from GitHub.

Taking Notes

In earlier cohorts I experimented with giving out notes in an editable format. But I found that people would often unintentionally change the notes, which meant that the notes were then wrong. I've switched to using PDFs as they allow for the nicest formatting and are also immune from accidental changes.

Make sure you open the PDF in a PDF viewing app. If you open it in an app that converts it into some other format (e.g. Google Docs) you may well miss out on important formatting, which will make the notes harder to follow.

I make an effort to include all the necessary information in the notes, so you shouldn't need to take any additional notes. However, I know that this doesn't work for everyone. There are various tools that you can use to annotate PDFs:

- Preview (Mac)
- Edge (Windows)
- Hypothes.is
- Google Drive (*not* Google Docs)
- Dropbox

Do not use a word processor to take programming notes! (e.g. Google Docs, Word, Pages). Word processors have the nasty habit of converting double-quotes into "smart-quotes". These can be almost impossible to spot in a text-editor, but will completely break your code.

Chapter 1

General

When you're viewing a website on the internet you're asking for a file via the url. A *server* sends you this file in response to this request, which the *browser* then interprets or renders for you.
Programming for the server is generally referred to as *backend*, and programming for the browser or client is usually referred to as *frontend*.
Here are some languages you might hear of, this is not a definitive list.

Backend Languages

- Javascript
- PHP
- Python
- Ruby
- Java
- C / C# / C++
- .net

Frontend Languages

- HTML
- CSS
- Javascript

These define roles you will hear of in companies. Here is a list, again this is not definitive:

Job Roles

- UX
- Designer / UI
- Front End Developers (Front End Designers, UI Developers)
- Back End Developers
- System Administrators
- Product Owners
- Project Managers
- Clients
- Testers

Chapter 2

Tools & Techniques

2.1 Devtools

2.2 Your Best Friend

DevTools are a developer's best friend. Being able to inspect and debug code is a **neccessary** skill when it comes to web development.

To open DevTools you can either right click in the browser and click 'Inspect Element', or you can use a shortcut - CMD + OPTION + I (Chrome and Firefox), this will vary between browsers. You should ALWAYS have your DevTools open when coding.

2.2.1 Essential Tabs

- Elements - most important tab for this course
- Network
- Audits

2.2.2 Debugging

The Elements tab alongside the Inspector and Device testing tools, are essential when marking up and styling web pages. They will give you:

- A built-in text editor, with immediate feedback in the browser
- Insights into the CSS cascade

- More visual tooling when working with the Box Model, CSS Grid etc
- Multiple screen widths, orientation and constraints to test against

This is not an exhaustive list.

2.2.3 Auditing

The Network and Audit tabs evaluate your sites performance in terms of:

- Accessibility
- Performance
- SEO
- Best Practices

This can be useful when developing your site for the first time and when looking at optimisations.

2.3 Upload to a server

2.4 Testing your website

Chapter 3

Accessibility

3.1 Assistive Techonologies

3.2 Accessibility

Accessibility is about making something available for everyone. Not excluding or catering for a specific subset.

Who

- Sight - Hearing - Learning difficulties - Mobility difficulties - Colour Blindness - YOU (imagine if you broke your wrist)

Why

- Users == Money/Revenue - Readable code == happy developers (more productive == money!) - Legal requirement (Don't get sued == money!) http://www.lflegal.com/2017/06/winn-dixie/

How

- Write good code - use the right element for the right content - Check design - colours, fonts, interface - Test: Wave Accessibility Tool [http://wave.webaim.org/](http://wave.w

Consider

- Video captioning - Big enough buttons - Cluttered/busy/animated pages - Lots of text (20ish words per line) - Font size - Not just using colour as feedback - Colour contrast - Zoomable - Images of text

3.2.1 Aria attributes

In case you use elements that aren't for what they're suppose to be for
For when you want to give users more information about those elements & modules
https://www.w3.org/TR/html-aria/

Screen readers use aria

"'html ‹button aria-label="Close" onclick="myDialog.close()"›X‹/button› "'
Further reading

Chapter 4

Project Process

4.1 Requirements Gathering

Receiving a Brief

Ask the following questions

- Who are your users?
- What are they trying to do?
- With which technology?
- What are the business aims?

From that we can determine

- What technologies are we going to use?
- Users need
- Does client need to be able to support it in-house?
- What other systems does it need to work with?

Assumptions

Remember you are not your user. We have to make assumptions, but we make sure we test those assumptions and modify accordingly

4.2 Sitemapping

Site maps used during the planning of a Web site by its designers.
Human-visible listings, typically hierarchical, of the pages on a site.
cite: wikipedia

Sitemapping tools

- Create a list in a wordprocessor
- https://www.gloomaps.com/
- https://bubbl.us/

4.3 Scamping

A quick sketch with a pen and paper
As you're discussing the site, just sketch it out
You can throw it out and start again easily
Think about:

- Content hierarchy on the page
- Layout + UI
- User journeys: How to get around
- Responsive behavior

Scamps & Scamping

The term *scamping* comes from films. Storyboards go through sketch out versions before the final storyboard is produced. This process is called scamping.

4.4 Wireframing

A wirefrmae is a representation of what is there and how it works.
Crucially, a way of thinking about the content and behaviour **before visual design is considered.**
That's why wireframes will look **bland and grey.**
We want to **focus on the function** of the site before we start thinking about style.
It will be given to:

- clients.
- designers.
- developers & sys admins.

Educate the client

Clients can sometimes be unfamiliar with digital processes. When sending a wireframe link to a client you should include an explanation of what a wireframe is for, what feedback is useful, and what isn't. Clients will circulate a wireframe link within the organisation, often without your helpful explanation of what a wireframe is for so add a note to the wireframe saying what it is.

Wireframe Software

- Balsamiq: https://balsamiq.com/products/
- Moqups: https://moqups.com/
- UXPin: https://www.uxpin.com/
- AdobeXD: https://www.adobe.com/uk/products/xd.html
- https://www.creativebloq.com/wireframes/top-wireframing-tools-11121302
- Any design software too

Prototypes

Wireframing is good but it doesn't give you interactivity. You can build a prototype to show that however. CSS Frameworks (see css.pdf) can be useful for this.

4.5 Design Handover

What do you want to ask the designer

- Artwork file
- Responsive designs
- Fonts

- Other assets (images, icons) can export from artwork yourself
- Media (videos, audio)
- Animations, interactivity (hovers)

What to expect

Lots of designs

- Home
- Standard page: About
- Contact page
- Article list page: news
- Individual article page

Different program files

- Photoshop (PSD)
- InDesign (INDD)
- Possibly Illustrator
- Sketch (not Adobe, mac only)
- Figma

Flat files ok...

- **PDFs** Good, vector-based, usually not lossy compression, often can copy text.
- **PNGs** Lossless.
- **JPGs** Lossy compression, difficult to get clean image from. Potentially different 'colourspace'.

4.6 Design Breakdown

4.6.1 Figma

Pages & Layers

Make sure sizes seem ok, have responsive designs

Tools & Menus at top

Use select tool

Select a layer

See information on right. NB check sub-layers. Things are grouped.

Export panel & different image types

Select layer and see bottom right

4.6.2 Different image types

Raster (Bitmap)

- **png** Has transparency, good compression, can be big
- **jpg** No transparency but smaller than png
- **tiff** No place on the web really
- **gif** Transparency, animation, small file size if icon based, incredibly big file size if animated.
- **webp** Google's new image format for the web. Incredibly good quality for compression. Not supported everywhere (check your picture element)

Vector based

- **svg** Scalable (and actually code!)

Blueprint

A blueprint is your own sketch of the site. It should include any information you might need in coding, such as fonts, colours and spacings

Glossary

- **Server**: A computer accessible via a network (in web world, the internet), which holds and serves files when requested.
- **Client**: The device accessing and viewing the world wide web.
- **Browser**: A piece of software which allows the user to view web pages.
- **Frontend**: The word commonly used to describe programming for the client side or browser.
- **Backend**: The word commonly used to describe programming for (or on) the server.

Colophon

Created using TEX

Fonts

- Feijoa by Klim Type Foundry
- Avenir Next by Adrian Frutiger, Akira Kobayashi & Akaki Razmadze
- Fira Mono by Carrois Apostrophe

Written by Ruth John

∞

December 1, 2020