COMP6080

🍓 Other - Outline

Lecture 1.1

Author(s): Hayden Smith

(Download as PDF)

🤷‍♀️ Why COMP6080?

Web is exploding. Everyone wants web-based front-end developers.

- Stack Overflow Developer Survey 2023: JavaScript has consistently been the most commonly used programming language for over 10 years. In 2023, 65% of professional developers reported using JavaScript.
- GitHub Octoverse 2023: JavaScript is the most active language on GitHub, with millions of repositories being built with it. It accounts for a significant portion of pull requests and commits on the platform.

🤷‍♀️ Why COMP6080?

Web is exploding. Everyone wants web-based front-end developers.

- 97.6% of all websites use JavaScript as their client-side programming language (according to W3Techs, 2023). This makes JavaScript an essential technology for building dynamic and interactive web pages.
- React.js (built on JavaScript) is the most popular front-end framework, used by over 42% of developers according to the Stack Overflow Developer Survey in 2023.

🤷‍♀️ Why COMP6080?

Within 10 weeks I want to make you all confident in building your own simple web apps, to build them quickly, and to build it to look how you want it to look and behave.

But first let's answer two questions:

1. What is a front-end developer?
2. Why is web exploding?

🎨 What Is A Front-End Developer?

Someone who writes code that is executed client-side, typically part of a end-user facing product.

In most cases, they write code for web browsers. This is a course about helping you make web browsers do things.

They largely use HTML to structure pages, CSS to style them, and Javascript to make them dynamic.

They build things for *people* to use.

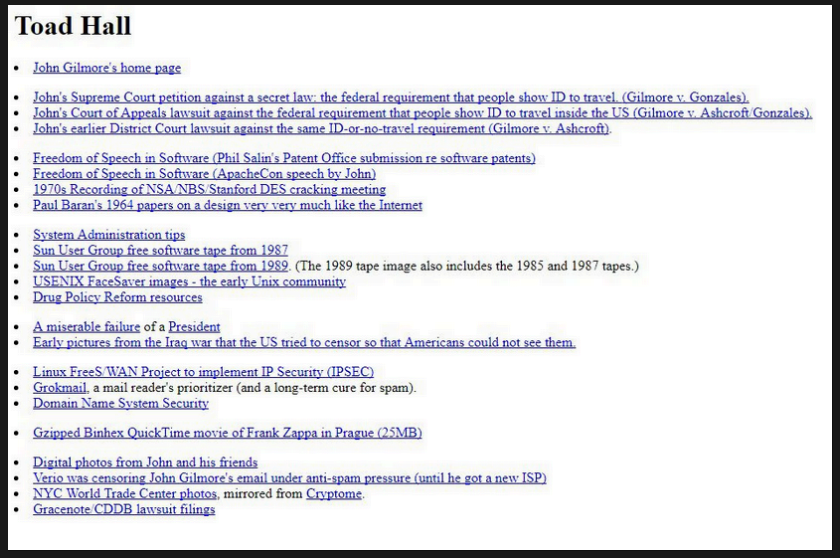
💥 Why Is Web Exploding?

And how did we get to this point?

 The Progress In Web

1️⃣ HTML Static Pages

Beginning of websites



Toad Hall

- John Gilmore's home page

- John's Supreme Court petition against a secret law: the federal requirement that people show ID to travel. (Gilmore v. Gonzales).
- John's Court of Appeals lawsuit against the federal requirement that people show ID to travel inside the US (Gilmore v. Ashcroft/Gonzales).
- John's earlier District Court lawsuit against the same ID-or-no-travel requirement (Gilmore v. Ashcroft).

- Freedom of Speech in Software (Phil Salin's Patent Office submission re software patents)
- Freedom of Speech in Software (ApacheCon speech by John)
- 1970s Recording of NSA/NBS/Stanford DES cracking meeting
- Paul Baran's 1964 papers on a design very very much like the Internet

- System Administration tips
- Sun User Group free software tape from 1987
- Sun User Group free software tape from 1989. (The 1989 tape image also includes the 1985 and 1987 tapes.)
- USENIX FaceSaver images - the early Unix community
- Drug Policy Reform resources

- A miserable failure of a President
- Early pictures from the Iraq war that the US tried to censor so that Americans could not see them.

- Linux FreeS/WAN Project to implement IP Security (IPSEC)
- Grokmail, a mail reader's prioritizer (and a long-term cure for spam).
- Domain Name System Security

- Gzipped Binhex QuickTime movie of Frank Zappa in Prague (25MB)

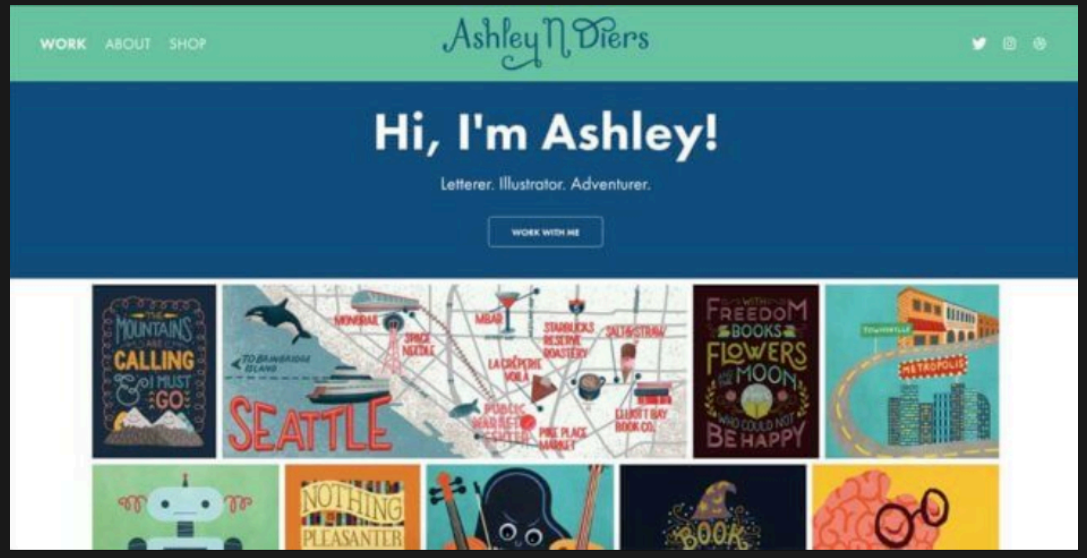
- Digital photos from John and his friends
- Verio was censoring John Gilmore's email under anti-spam pressure (until he got a new ISP)
- NYC World Trade Center photos, mirrored from Cryptome.
- Gracenote/CDDB lawsuit filings

HTML released 1993

🏃‍♀️ The Progress In Web

2️⃣ Basic CSS / Javascript

Now things don't like terrible

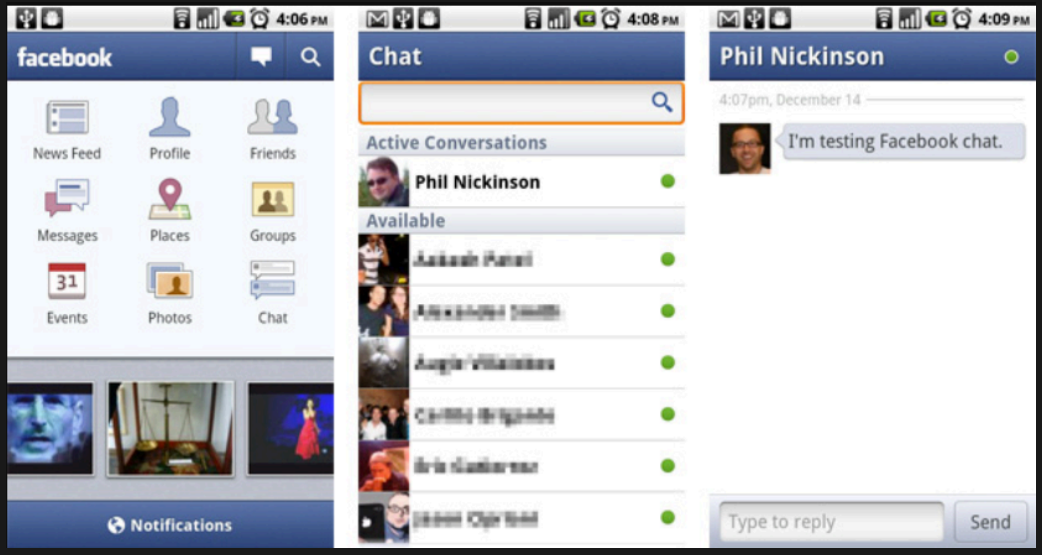


JS released 1995, CSS released 1996

 The Progress In Web

3️⃣ AJAX and background requests

A whole new class of websites, without the need to refresh



AJAX appeared 2000~

🚶‍♀️ The Progress In Web

4️⃣ NodeJS & Typescript

Type checking and common codebases - the beginning of heavy JS infrastructure

NodeJS released 2009, TS released 2012

🚶‍♀️ The Progress In Web

5️⃣ Hybrid Apps, Electron

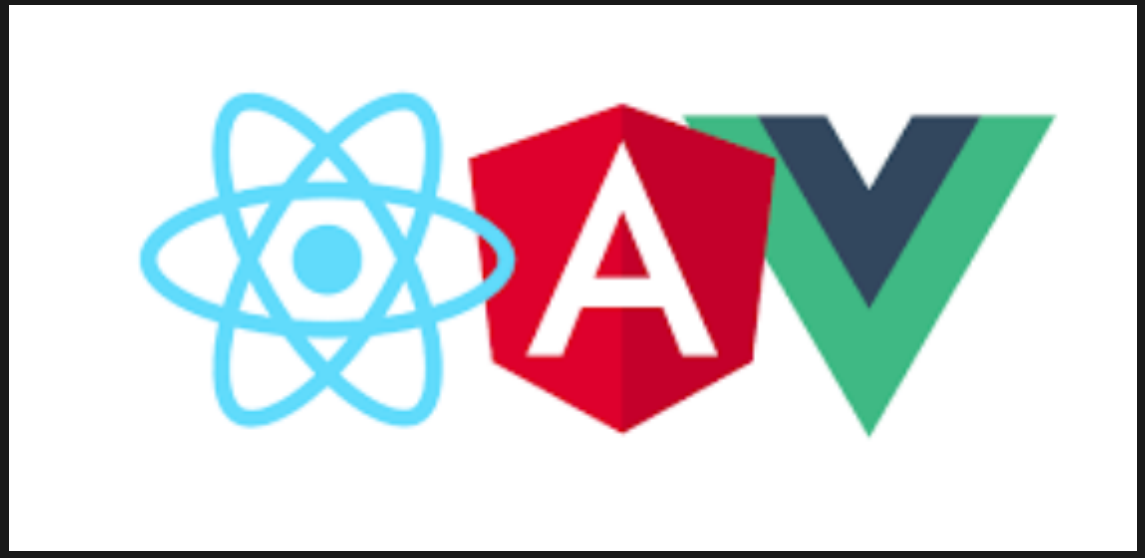
Type checking and common codebases - the beginning of heavy JS infrastructure

Electron, Apache Cordova

🏃‍♀️ The Progress In Web

6️⃣ Declarative Frameworks

Building of complex applications rapidly and in a scalable way. A web approach finally optimised for apps, rather than pages.



AngularJS released 2010, ReactJS released 2013, VueJS released 2014

The Progress In Web

In the vast majority of cases, your native desktop applications, native mobile applications, and web-apps or websites, can be built on a web-based, javascript based stack.

🧠 Assumed Knowledge

- Basics of GIT
- Basics of HTTP & Web Browsers
- High level understanding of scripting languages (either Javascript or Python)

This course has no other assumed knowledge. Experienced web developers will likely find this course slow.

If you lack this assumed knowledge, we have lectures to provide you all the help.

😍 Content We Cover

- HTML
- CSS
- Javascript (Vanilla)
- Javascript (Async)
- UI/UX/Accessibility
- Web Framework (ReactJS)

🥹 Content We Don't Cover

- Javascript classes
- Global state managers
- Advanced CSS (canvas, animation)
- CSS Compilers (.less, .scss)

📗 Summarised Learning Outcomes

- Fundamentals of Javascript to design, construct, test, debug code
- Understanding HTML, CSS, DOM, to construct web pages
- Modern Javascript & CSS frameworks, to build componentised apps
- Understanding of asynchronous programming in the context of JS
- Basic knowledge of front-end security
- Awareness of UI/UX design, including accessibility

🌻 Approach To Teaching

Firstly, this is a challenging level 6 course. It will be hard to perform in the HD bracket without hard work or natural intellect.

Secondly, learning front-end development will be a bit different from other languages you've learned:

- Front-end is a 'breadth' topic, which is unlike many other languages
- A lot less time is spent solving algorithmic or computational problems, and a lot more time is spent just trying to fine tune your visuals or find the right method/library/approach to get the expected behaviour
- You will still feel like you have a lot to learn by the end of the course

This will make assignments feel very large. If you are uncomfortable with medium-high workloads, or uncomfortable with the perception of very high workloads, please reconsider this course. **This course is not for everyone**

💻 Course Site

Our course site is a custom content management system called Eckles. Eckles was written by Hayden in January 2022.

It is a ReactJS application with a lightweight NodeJS server. The source code is public for all all students, and we encourage you to contribute.

🔨 Why Was This Built?

- COMP6080 is trying to pioneer a new style of learning in CSE, and all previous software was not considered appropriate for this style
- We want to give you a real example to reference throughout the course and something we can all have fun with together.

🥰 The Team

We have a team of staff from UNSW who support you through your teaching. We also have a helping hand with some pre-recorded lectures from Canva.

👀 Assessment Structure

There is no direct assessment associated with Lectures or Tutorials

Item	Due	Weighting
Assignments	Due weeks 3, 4, 7, 10	80%
Exam	Exam Period	20%

📝 Important Notes About The Course

- This is a "hard" course for some - it's a breadth course (unique for many of you)
- Assignment due dates are as late as possible (includes no late penalty)
- We consider Vanilla JS a critical part of teaching - a course just in ReactJS wouldn't do you justice.
- Pre-recorded industry lectures are always being reviewed
- We can't teach everything! So we have to cut some things, and sometimes have bonus lectures.
- We don't have labs, because this is a level 6 course.

👨🏽‍🏫 Teaching Methods

🟢 Lectures

- Avg 2 hours live lectures (ideally no new content, all demos, don't expect pre-prepared perfection)
- Avg 2 hours of pre-recorded per week (tightly broken up)
- Lectures are categorised into levels of importance for you.

🟡 Tutorials

- You can attend any tutorial you want
- You can attend no tutorials if you wish
- All tutorials have been pre-recorded to learn at your own pace
- Tutorials are also your self-learning exercises
- Please respect your tutor probably needs to leave at the end

 Teaching Methods

🔵 Help Sessions

- Chance to talk to tutors and get help on matters to do with tutorial exercises and assignments
- Help sessions will contain 1-5 tutors who will be split between assisting students with questions, and marking labs off
- Please pay attention to how many tutors are in a given help session - we do our best to predict demand and adjust but please attend help sessions being prepared to wait

🟣 Assignments - In-Depth Skills

- Ass1: Static HTML/CSS
- Ass2: Intro to DOM
- Ass3: Building an app with HTML/CSS/VanillaJS
- Ass4: Building and testing an app with ReactJS (pair)

🔥 Assignments are now due 4 days later this term 🔥

💖 Testimonials

Assignments are really fun and give so much freedom to implement features how you want to.

💖 Testimonials

The course workload is slightly above average, I spent about 50~ hours on assignment 3, and just over 60 on assignment 4 working solo. Considering there's no other assessable content throughout the term, and a 20% weighted exam, I'd say the course workload is in an good state for a level 6 course.

💖 Testimonials

Got to take so much out of this course, definitely worth the blood sweat and tears to come out feeling like I've got an applicable skill

 Quick Q&A!

We're going to do a quick Q&A based off of previous comments students have left at the end of the course - so that you don't have to continue with the course if it's not what you're looking for.

 Quick Q&A!

Q. Can we spend less time on HTML/CSS? 1.5 weeks is too much

A. No! It's 1) An important and fundamental thing to understand, and 2) Helps ease beginners (most people) in

 Quick Q&A!

Q. Tutorials are too short

A. No issue, you can attend as many tutorials as you like.

 Quick Q&A!

Q. Can we have 3 assignments instead of 4?

A. The number of assignments is arbitrary, we could combine assignment 1 and 2 and make it 3 but it would change nothing.

 Quick Q&A!

Q. Can we have typescript incorporated?

A. Whilst the course isn't too big, it's still quite big. The solution to this is to create a second course where we can cover things like that - but that's a long term solution.

 Quick Q&A!

Q. Can we learn Vue and Angular?

A. Whilst the course isn't too big, it's still quite big. The solution to this is to create a second course where we can cover things like that - but that's a long term solution.

 Quick Q&A!

Q. Can we have marked labs every week?

A. We have tried this; students overwhelmingly told us they did not like it. If you want extra feedback, complete the exercises, and share it with a tutor during a help session.

 Quick Q&A!

Q. Can we remove some of the poorer quality Canva lectures?

A. We are slowly removing older lectures that have poorer recording quality. Thanks for your patience!

 Quick Q&A!

Q. Can we have less lectures? There are too many!

A. Core lectures are pretty minimal. If you only watch mandatory lectures you can ignore the others. Just because optional ones are there does not mean you need to watch them.

 Quick Q&A!

Q. Introduce a late penalty! No late penalties are too stressful!

A. Assignment deadlines are currently 1 hour before marking has to start. The only way to introduce a late penalty is to bring the deadline forward. If you'd like your deadline brought forward with a late penalty added email me and we can sort it out!

 Quick Q&A!

Q. Assignments are released too early

A. I know that having things released can be stressful. Because of short terms on 1 occasion an assignment is released before the other is due. If this structure is stressful please reconsider the course.

Quick Q&A!

Q. Penalties for significant violations are too high

A. Certain programming practices are clearly and strictly banned in assignment 3 and 4. Penalties apply for ignoring these. They are significant. If you're someone who wants to not turn up to lectures, not read the spec, and find penalties stressful, please reconsider the course.

🤖 AI Usage Policy

The final exam does not allow for the usage of LLMs.

Besides that, you are welcome to use them as assistants (not code generation tools) throughout the course.

Please remember: AI is killing graduate roles, don't become collateral.

🙏🏾 Getting Help

If you need help with something, go here:

- EdStem (sidebar)
  - Note Hayden does not directly monitor
- Help Sessions
- cs6080@cse.unsw.edu.au

☃️ Getting Setup

⌨️ Running Your Code

- In this course you'll need to use: NodeJS, NPM, Web Browser, HTML, CSS, ReactJS, and more.
- All of these tools can be easily run on your local machine (recommended) for OSX, Linux, or Windows.
- If you aren't comfortable with local installs, you can complete the course using gitlab (we have installed all relevant programs there).

💾 Gitlab

- gitlab is the online tool we're using to manage our git repositories.
- We will provide a demo of it's usage.
- There are git resources on Eckles.

📚 Resources

- Web is a very well-resourced set of tools on the internet. Self-guided research will generally be adequate.
- The biggest issue will be finding the appropriate resources.
- Lectures may include resources on slides, and we also have a course-wide resources page here.

🎭 Style Guide

Information about style guides for particular languages can be found on various parts of our resources pages.

Feedback

Throughout term, you can leave anonymous feedback by clicking on the link in the Eckles sidebar "Feedback"

🙏 Content Requests Board

We have added a content requests board that is accessible via the sidebar.

Common Work Time

We've booked a room 4pm-6pm on Monday's (right before the lecture) that students are free to join and meet / hang out with other students! Location will be shared in the weekly notices.

💝 Getting Along

- Understand the expectations around student conduct.
- Create an inclusive learning environment.
- Let's all treat each other with respect and understanding.

🚀 Related Courses

- COMP3511: Human Computer Interaction
- COMP4511: User Interface Design and Construction
- COMP6443: Web Application Security & Testing

⚒️ A Taste: "Making Web Browsers Do Things"

- Most things we do in this course are various combinations of:
  - **HTML**: Page structure
  - **CSS**: Page aesthetics
  - **Javascript**: Page dynamics
- Other topics we discuss are not about technology, but rather how to use it effectively:
  - Accessibility, product design, UI/UX, testing
- (Optional) Let's do a short demo to explore what is coming up in the first weeks

🦻 Feedback

Or go to the form here.