Week 13: The Final Lecture

INFSCI 2560: Web Technologies and Standards

Slides will be posted after class.

Today's Agenda

1. Final Topics
2. Team Meeting Time

Exam 2

Next Monday, December 2

Exam 2 Study Guide Shared. (link)

Questions?

Final Project

- Use Glitch Teams to create a project team.
- Determine everyone's role/responsibility
- Create milestones
- Sit in the same room and code.
- **You should write ALL the code -** *no copy/paste from a tutorial or other source.*

Final Project Contest

- **Most beautiful:** Given to the most aesthetically pleasing final project
- **Best code:** Given to the final project with exceptionally well-written code
- **Best in show:** Given to the all-round most impressive final project

If your final project is selected, everyone on the team will receive 10 bonus points to their lowest project score.

Let’s address some questions you may have.

Why are we using Glitch?

- Allows us to focus on coding
- You don’t have to setup an environment, version control, etc.
- Glitch deploys the code for you.
- Great platform to learn and experiment with code.
- Built on top of Node.js and you can install/manage any npm packages
- You have access to logs and the console.
- Integrated with Git
    - Import/Export

There are so many changing technologies…

- How do I know which ones to use?
- How do I learn about new libraries?
- How do I stay up to date?
- Won't everything I learn be obsolete in 6 months?!?!

Q: How do I stay up to date??

A: This is the wrong question to ask.

Staying "up to date" is not that important.

Technology doesn't fundamentally change very often or very fast.

New tech: Helpful, not necessary

Most new web technologies makes your life easier **but is not necessary.**

Examples:

- `const` and `let`
- `async` / `await`
- CSS variables, etc

Everything* you want to do can already be done with the web technology available not just today, but 10 years ago.

*You know, within reason

Fundamentals don't change

Tech *doesn't* change that quickly

- Much of Facebook is still written in PHP
- Most of Google is written in Java and C++
- In practice, you will not (and should not) totally rewrite your codebase every year

- Tons of parallel problems, patterns, etc across tech

The real question to ask

How do I distinguish good web technology from bad web technology?

Also: **Many new libraries are bad.**

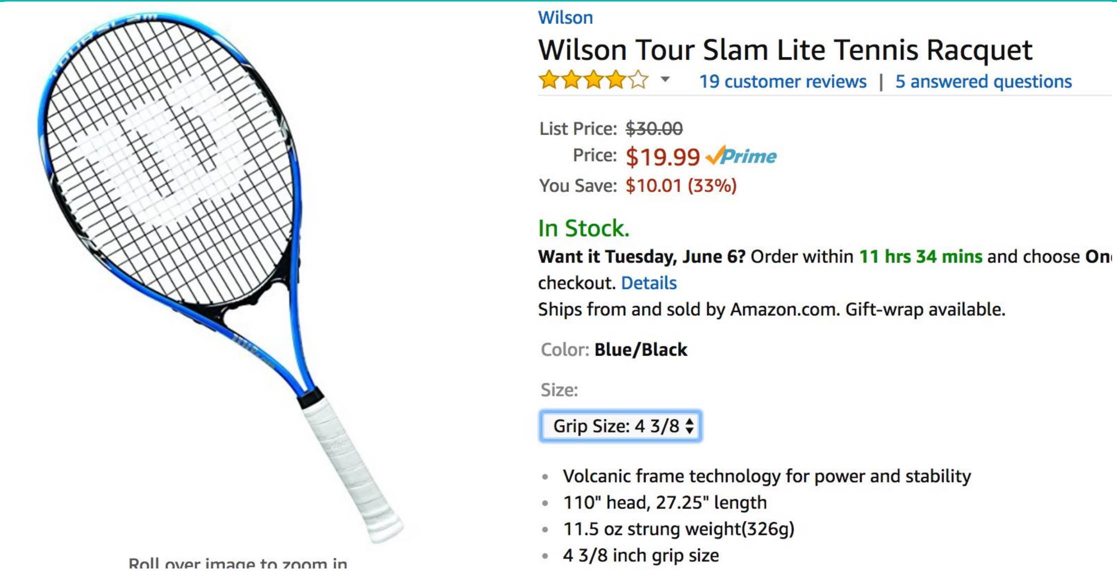
- Literally anyone can post a library on npm.
- Most libraries on npm are therefore garbage.
- Even popular libraries can be poorly written.

Choosing good tools

Either:

- You have enough knowledge to be able to decide whether a tool or technology is beneficial

Choosing good tools



Roll over image to zoom in

Wilson

Wilson Tour Slam Lite Tennis Racquet

19 customer reviews | 5 answered questions

List Price: ~~$30.00~~

Price: $19.99 ✓Prime

You Save: $10.01 (33%)

In Stock.

Want it Tuesday, June 6? Order within **11 hrs 34 mins** and choose **On**
checkout. Details

Ships from and sold by Amazon.com. Gift-wrap available.

Color: **Blue/Black**

Size:

Grip Size: 4 3/8

- Volcanic frame technology for power and stability
- 110" head, 27.25" length
- 11.5 oz strung weight(326g)
- 4 3/8 inch grip size

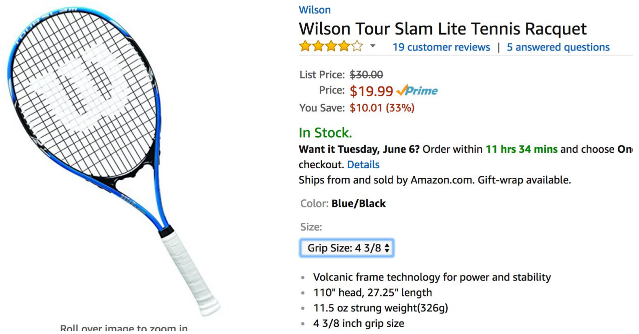
Or:

- You don't have enough knowledge to tell the difference
- Therefore it doesn't really matter
- And you should choose the simplest / cheapest thing that other people say is good

Choosing good tools

If you keep getting better at tennis, someday you'll look back at your first racquet and think

- "OMG how was I using this terrible racquet" or,
- "Lol I had a $300 racquet and had no idea how to use it", or
- "Huh, that cheap one was actually pretty good"



Roll over image to zoom in

Wilson

Wilson Tour Slam Lite Tennis Racquet

★★★★☆ 19 customer reviews | 5 answered questions

List Price: ~~$30.00~~

Price: $19.99 ✓Prime

You Save: $10.01 (33%)

In Stock.

Want it Tuesday, June 6? Order within 11 hrs 34 mins and choose On checkout. Details

Ships from and sold by Amazon.com. Gift-wrap available.

Color: **Blue/Black**

Size:

Grip Size: 4 3/8

- Volcanic frame technology for power and stability
- 110" head, 27.25" length
- 11.5 oz strung weight(326g)
- 4 3/8 inch grip size

But the ability to choose good tools takes expertise and experience that you don't have as a beginner.

Choosing good tools

And sometimes there's just a bit of personal preference

General advice

Focus on becoming a good engineer.

- Learn how to build good software in any language, frontend, backend, web, iOS, Android, data pipelines, anything.

Work as a full-time software engineer for N years **with other (good) people.**

- Even after 1 year working full-time, your engineering skills will improve immensely

This is how you will develop and hone your own technical judgement.

General advice

Don't be afraid or intimidated by new technology.

When you confront a new web thing, like a library or framework, one of two things will happen:

1. You will be excited by it, and you will want to use it.
2. You will not be excited by it, and you can safely ignore it.

Simpler is always better.

- ALWAYS delete code if you can
- ALWAYS remove a library if you can
- ALWAYS remove a framework if you can

Topics you really-really-really ought to know if you want to be a full-stack developer

Testing Testing Testing Testing
Testing Testing Testing Testing
Testing Testing Testing Testing
Testing Testing Testing Testing
Testing Testing Testing Testing

Missed topic: Robustness

The code we wrote in INFSCI 2560 is **extremely fragile**:

- No tests
- No type checking
- No backups for databases
- Etc.

MUST: Server-side Testing

If you write production server code, **you must write tests.**

Q: What are tests?

- A test is a type of software that verifies the code you wrote works
- Tests help you:
  - Verify everything works before you launch your product
  - Catch regressions as you modify code

Types of testing

Functional Testing :

Unit Testing*

Integration Testing*

System Testing

Sanity Testing

Smoke Testing

Interface Testing

Regression Testing

Beta/Acceptance Testing

Non-Functional Testing:

Performance Testing

Load Testing

Stress Testing

Volume Testing

Security Testing

Compatibility Testing

Install Testing

Recovery Testing

Reliability Testing

Usability Testing

Compliance Testing

Localization Testing

MUST: Server-side Testing

You should probably write tests for all your code, but server-side testing is especially important

Check out:

- MochaJS: A popular JavaScript test framework that works on frontend and backend (NodeJS) code
- Jest: Facebook's JS test framework
- Chai: Helper library to write easier-to-read tests
  - Used with Mocha, Jest, etc

Warning: Setting up tests for the first time always sucks.

Older browser support

Older browsers?

In INFSCI 2560, we used JavaScript features that worked on the latest version of each major browser.

But sometimes you need to support older browsers.

What do you do?

- Don't use the new stuff until it's ready? But when will that be?
- Write multiple versions of your code? But that's time-consuming and annoying
- Write polyfill fallback code? Also super annoying

BabelJS

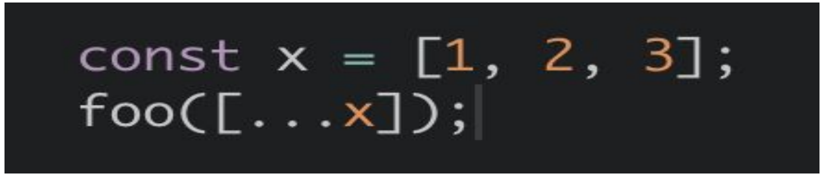
Solution:**BabelJS**

- Babel is a JavaScript compiler for the latest features of EcmaScript, including ES6+
  - If the browser supports ES6 natively, babel does nothing
  - If the browser does not support ES6 natively, babel provides a polyfill

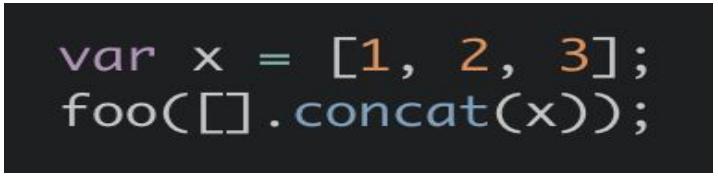
Use BabelJS so that you can:

- Write code with the latest features in JavaScript
- Support older browsers without having to rewrite anything

Compiling with Babel



```
const x = [1, 2, 3];
foo([...x]);
```



```
var x = [1, 2, 3];
foo([].concat(x));
```

ES6 code

JavaScript that works on older browser

Use Babel!

Type checking

Missed topic: Type checking

JavaScript is loosely typed, meaning you do not declare the data types of variables.

- Sometimes loose typing is a great thing, e.g. when you are starting a project from scratch, prototyping, etc.

- But loose typing gets to be a pain as your code base grows.

Type checking

There are ways to essentially add **type checking** to JS:

- TypeScript: A programming language that is a superset of JavaScript. Write TypeScript code and **transpile** it to raw JavaScript.

- Flow: A static type checker for JavaScript. Write annotated JavaScript code and **transpile** it to raw JavaScript.

- Closure Compiler: An early bundler, code minimizer, and static type checker for JavaScript. Type definitions are done in comments and doesn't require transpiling.

TypeScript (2012)

- TypeScript is a **programming language** by Microsoft
- It is a superset of JavaScript that includes static typing.
- Browsers can only execute JavaScript, so you must **transpile** TypeScript to JavaScript

TypeScript

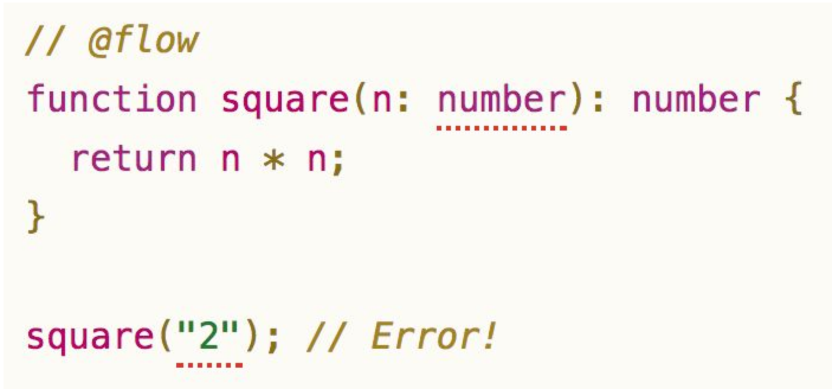
```
function Greeter(greeting: string) {
    this.greeting = greeting;
}
```

JavaScript

```
function Greeter(greeting) {
    this.greeting = greeting;
}
```

Flow (2014)

- Flow is a static type checker by Facebook.
- It is not a full programming language, but it involves adding a combination of non-standard annotations and comments to your JavaScript.
- Browsers can only execute JavaScript, so you must **transpile** Flow-annotated code to JavaScript

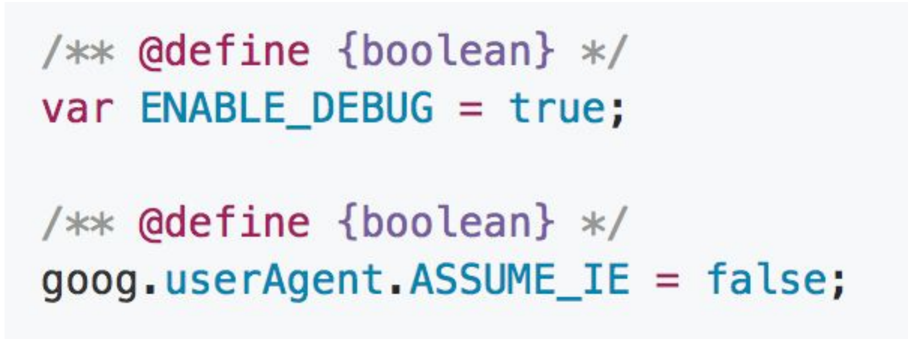


```
// @flow
function square(n: number): number {
  return n * n;
}

square("2"); // Error!
```

Closure compiler (2009)

- Closure Compiler is a command-line tool by Google
- Transforms valid JavaScript into more efficient valid JavaScript.
- Type information (closure annotations) is specified in comments



```
/** @define {boolean} */
var ENABLE_DEBUG = true;

/** @define {boolean} */
goog.userAgent.ASSUME_IE = false;
```

Web Security

The Security Mindset

“Security requires a particular mindset. Security professionals—at least the good ones—see the world differently. They can’t walk into a store without noticing how they might shoplift. They can’t use a computer without wondering about the security vulnerabilities. They can’t vote without trying to figure out how to vote twice. They just can’t help it.”

“The Security Mindset” by Bruce Schneier, at https://www.schneier.com/blog/archives/2008/03/the_security_mi_1.html

Keep your code as bug-free as possible

- Testing is SUPER important.
- Seek code reviews.
- Don’t reinvent the wheel.
- Enforce good JavaScript with JSHint
- Check/validate any input you get from users (e.g. query params)
- Audit code of any of your dependencies
- Keep your dependencies up to date
- Check against the Node Security Project

Reading

Expect at least 2 questions from this reading on the exam.

- Read chapter 10 from Express.js in Action
  - Fulltext online through ULS (Pitt Login Required)
- Read this article from MDN on the First Steps in Web Security

Topics you might find handy

Misc web topics

A few other topics that might be useful for you:

<canvas>

- Allows you to draw graphics in a <canvas> tag
- Uses more traditional, lower level graphics commands
- 3d support with WebGL
- Simple demo; complex demo
- Canonical examples: Games; complex visualizations

WebSockets / Socket.io

- Used for server -> client messages
- Canonical examples: Chat client; gaming; anything that has live updating

Publishing tools

Publishing static web pages

Domain name registration:

- Reserves a custom URL: myawesomesite.com
- But doesn't usually include web hosting; all you own is the name.

Web hosting:

- Provides a location on the internet to upload files
- Usually with some crummy URL, like http://bucket.s3-website-us-west-2.amazonaws.com/

Domain name registration and web hosting are sometimes provided by the same company, but not always.

Publishing static web pages

You can register your own domain name through many companies:

- Google Domains: Only domain name registration
- Amazon S3: Only web hosting
- Dreamhost: Domain name **and** web hosting options
- GoDaddy: Domain name **and** web hosting options

Domain name registration is usually ~$12/year

Web hosting is usually ~$10/month

- Amazon S3 is **significantly** cheaper (virtually free for low-traffic websites) but more complicated to set up

Publishing server-side code

If you want to host both a frontend and a backend, you need a web host that allows you to configure a server.

There are an immense number of options, with different levels of configuration. Here are some:

- Heroku: Super easy to use, but offers less control. Also a lot more expensive.
- AWS: Cheap, lots of options, but more complicated
- Google Cloud: Basically the Target brand of AWS: Cheaper than AWS; as complex as AWS; fewer products than AWS

Tonya’s Perspective

jQuery: Don't use

jQuery was built in 11 years ago when the web was in a much worse state

But now most of jQuery's features have native JS equivalents

- `document.querySelector`
- `classList`
- ES6 classes
- CSS animations
- etc.

jQuery: Don't use

jQuery also provides cross-browser compatibility, but you should prefer babel for that.

Suggestion:

- Only use jQuery if you're forced to, i.e. if you're working in a code base that already uses jQuery and you can't change it.

Bootstrap: Don't use

Bootstrap is a *really heavyweight*, not-very-flexible set of default CSS styles and JavaScript components

Bootstrap is nice for what the name implies: bootstrapping a pretty, generic-looking website

However, Bootstrap is often used as a crutch by people who don't want to learn CSS.

Bootstrap: Don't use

Suggestion:

- Use Bootstrap if you want your page to look like every other site.
- Otherwise, avoid Bootstrap:
  - It is really hard to do anything that's not this
  - It is *really* hard to debug
- Learn and use raw CSS:
  - Use CSS flexbox
  - Use CSS grid
- Hire a designer to make your website look nice

Recap

DON'T-dos:

- **Don't use jQuery**
- **Don't use Bootstrap (consider using CSS grid)**
- Don't unnecessarily complicate your tech stack
- Don't be afraid of new libraries/tools/frameworks.
  - If they are good, they make your life easier, not harder!

BASARAT Follow
That TypeScript Guy http://www.ba
Apr 27, 2016 · 3 min read

TypeScript won

I love all the people (great develo
mentioned in this post 🌹. That s

Forget Angular & Ember, React Has Already Won the Client-Side War

Is golang the future?

Is Golang dead?

Q: Which library/tool/language/platform is going to **win**?????

Is jQuery Still Relevant?

Is Java Dead? No! Here's Why…

R.I.P. Ruby on Rails. Thanks for everything.

Published on January 13, 2016

▲ Ask HN: Is Python dying?

Is Django already a dying technology?

A: Wrong question.

CS is not a competitive sport.

Not everything is a dominance hierarchy.

Not everything is a dominance hierarchy.

Not everything is a dominance hierarchy.

Not everything is a dominance hierarchy.

Not everything is a dominance hierarchy.

- JavaScript libraries are not at war.
- Multiple things can be good.
- Learning **any good library** is valuable, even if it's not in its absolute height of popularity.
  - A great way to improve your software engineering skills: Studying other people's designs

Better questions

- Does this library solve the problems that I care about?
- Is this library production-ready?
    - Does it have prominent clients?
    - Does it work at scale?
    - Has it worked out most of its bugs?
- Is this library under active development?
    - Does it *need* work?
- How easy is it to find documentation/StackOverflow results for this library?
    - Does it *need* documentation/help pages?

Final advice

Staying up to date

With all the caveats aside:

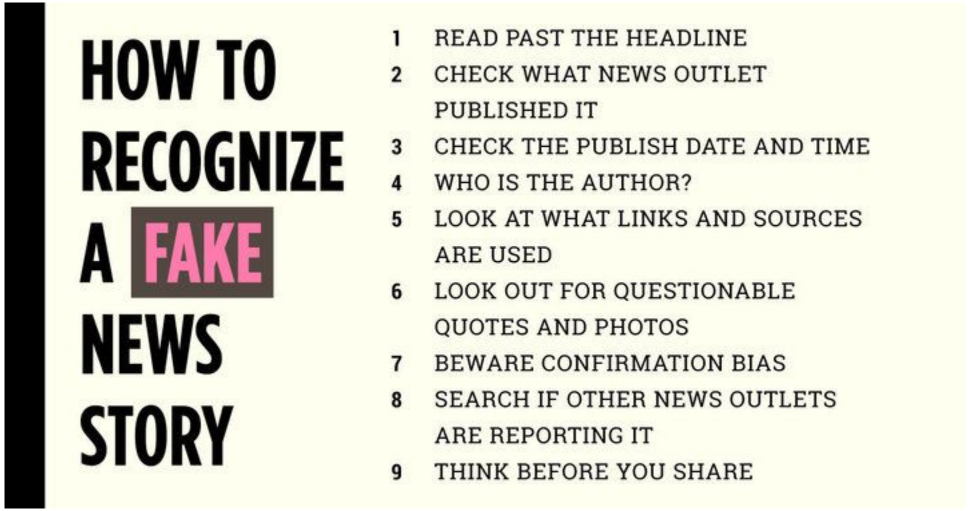
Q: "How do you stay up to date on web stuff?"

Staying up to date

With all the caveats aside:

Q: "How do you stay up to date on web stuff?"

A: Read the internet! But tread carefully:



HOW TO RECOGNIZE A FAKE NEWS STORY

1. READ PAST THE HEADLINE
2. CHECK WHAT NEWS OUTLET PUBLISHED IT
3. CHECK THE PUBLISH DATE AND TIME
4. WHO IS THE AUTHOR?
5. LOOK AT WHAT LINKS AND SOURCES ARE USED
6. LOOK OUT FOR QUESTIONABLE QUOTES AND PHOTOS
7. BEWARE CONFIRMATION BIAS
8. SEARCH IF OTHER NEWS OUTLETS ARE REPORTING IT
9. THINK BEFORE YOU SHARE

Garbage piles

Do not trust:

- Comment sections of Reddit
- Comment sections of Hacker News
- Comment sections of any website
- Medium articles by randos

In my experience, these are far too often full of posturing, gross misinformation, terrible opinions based on little-to-no facts, etc.

Hit-and-miss

Usually works, but sometimes poor style / not best practice

- StackOverflow answers
- W3C schools
- Programming YouTube videos

Better opinions than most, but sometimes still trash

- Quora answers

Good web resources

Reliable websites

- Google Web Fundamentals
- Mozilla hacks and Mozilla Developer Network

Official documentation:

- HTML WHATWG spec / HTML W3C spec
- EcmaScript status / spec

Write code

The only way to get better at web programming is to write lots and lots of code.

- Become a software engineer
- Work with software engineers who are better than you
- Write simple side projects to learn new tech
    - **Suggestion:** Choose a project you know you could finish in 1 day - 1 week

You can do it!

Thank you!

I hope you enjoyed this class!