Week 1

HISTORY:

* 1990-2000 – web 1.0 (internet used for sharing documents)
* 2000-2010 – web 2.0 (web apps became dynamic)
* 2010- present – web 3.0 (better version of web 2.0)
* Bush was head of science in WWII

Dirty Secret of SPA’s

* The Internet never intended for pages to be interactive. It was intended for displaying documents. NOT GUIS

Web Model:

* Client holds browswer and SPA (static files in single page app). Client connects to Server via HTTP(S). Server contains web server and app server. Web and app servers are connected. App server connects to database server which is not in server. Serverside javasript is connected to app server.

Benefits of an SPA

* Good performance
* Don’t have to you download an app everytime you want to visit a website
  + This saves space, improves security, and improves browsing speed.
* Single file is loaded on startup (static files)
* It has server-side asynchronous processing

SPA benefits over a Traditional app

* SPAs divide client side vs server side for storing data
* Don’t need to download the app to use it
* SPA’s aren’t really apps, they’re meant to give the illusions they are
* SPA = HTML + javascript + CSS

Internet vs World Wide Web

* WWW is a structure built on top of the internet. The internet is interconnected computers that make up every location.

Week 2

Elements of World Wide Web

* HTTP or Hypertext transfer protocol
* HTML – Hypertext markup language
* URL – Unifrom resource locator

What does a browser engine do

* Transforms HTML docs and other stuff into an interactive visual representation
* Chome uses Blink which is written in C++

Why Javascript is “Kluge” or seen as bad

* Javascript trades performance for dynaminism
* Misleading confusing name
* Semicolon insertion if you forget
* Many things that should be errors are allowed.
* Early impleminations were quite buggy.
* Real time so it has less performance
* You can use protoypes in ways you shouldn’t be able to

Requirements for Object Oriented Language

* Encapsulation
* Polymorphism
* Inheritance

Promises

* Promises let you proccess asynchronously and when the results get there deteremine if success or if failure
* const promise = createAudioFileAsync(audioSettings);
* promise.then(successCallback, failureCallback);
* createAudioFileAsync(audioSettings).then(successCallback, failureCallback);

Closures

* Functions within functions have access to out-scoped function’s variables. You can save closures into a variable and call it with the variable.

Strict mode

* Stops silent errors in JavaScript to throw errors
* Good for really debugging your code
* Lets you code with increased performance

Week 3

Benefits of NodeJS:

* Single threaded
* Fast and synchronous
* It demuxes server requests into an event loop
* Designed for internet type requests
* Be careful not to block the main thread, the asynchronous part proccess buffer in pieces

When to use Node.js

* I/O Bound Apps
* Data streaming
* Single Page Apps
* Json api apps.
* Anytime you want fast asynchnous app.
* NOT FOR CPU INTENSIVE PROGRAMS

Common Stacks:

* LAMP – Linux, apatchi, MySQL, PHP
* MEAN – MongoDB, ExpressJS, AngularJS, Node.JS)
* MERU
* RUBY ON RAILS
* Flask – pthon based

Package.json is used to define properties of a module

Layers:

* Client layer
* Server layer
* Business layer
* Data layer

Middeware

* Respond to Http request

Static files

* HTML, CSS< Javascript

Routing