AngularJS Notes-

Responsive website using angular \*\*very quick \*\*\*Directives

Request is only bringing information needed (JSON data)

Updates what we see.

A client-side Javascript framework for adding interactivity to HTML

How to trigger javascript? \*through directives so we add ng-controller!

\*function in js file is related to ng-controller

Modules- write pieces of our Angular application

Maintain, testable, readable code.

“var app= angular.module(‘store’, [ ]); //inside app.js file

What is it trying to solve: modern web development feel more fluid.

DOM- document object mode

Data and HTML, how to we connect them?

\*\*Model View Controller or MVVM- Angular is Model View whatever

Suppose we have model/ data (variable) then view (what people see HTML) and something else \*\*\*model and view are bound! Controllers or view models, angular doesn’t care. MV\*

HTML Aside:

Javascript Aside

Dependency injection:

\*giving a function an object. Rather than creating an object inside a function, you pass it to the function

* Ex: var Person = function(firstName, lastName){  
   this.firstName= firstName;

this.lastName= lastName;

}

Function logPerson(){

Var john = new Person(‘John’,’Doe’);

Console.log(john);}

logPerson();

\*\*this is long and complicated so we create a dependency injection:

\*Used by Angular JS (strong, stable structures)

* var Person = function(firstName, lastName){  
   this.firstName= firstName;

this.lastName= lastName;

}

function logPerson(person){

Console.log(person);}

Var john = new Person(‘John’,’Doe’);

logPerson(john);

1. The Scope Service:
   1. JavaScript Aside:
      1. Other Services. \*dependency injection concept!
   2. Another note is to understand:
      1. var myApp = angular.module('myApp', [\*\*\*list of modules (dependencies), ngMessages]);
   3. Arrays and Functions:
      1. Javascript arrays and how they work with function
         1. I can have an array seprated by things and can contain functions
   4. Single Page Apps
      1. Paragraphs of text. Anchor tag. #bookmark
2. Dependency Injection and Minification of js:
   1. Minification: shrinking the size of files for faster download.
      1. Normally add .min to name of file so file.min.js so we can tell the difference
      2. Make sure your order of calling strings and then functions parameters are correct!
3. Data Binding and Directives:
   1. Scope and Interpolation
      1. Since we understand how controller is connected to view, now the fun stuff
      2. Interpolation:
         1. Creating a string by combining strings and placeholders
            1. “My name is ” + name;
      3. Directives and Two Way Data Binding
         1. Directive – instruction to AngularJS to manipulate a piece of the DOM
            1. Prefers to use directives, powerful, flexible, easier to do, instruction : “Add a class”, “Create a new webpage”, ect.
            2. Two way data binding! No extra code because we used variable to bind view to model
            3. Remember twitter lesson!
            4. Model view whatever
      4. Event Loops:
         1. Document.getElementById means fetch from DOM
         2. Loops that listen for events:
            1. Keypress, click, mouseover, change
         3. AngularJS adds on angular context to loop.
            1. Adds watchers\*\* located in Digest Loop ask if something has changed, updates asap.
            2. When adding variables or functions
            3. Automatically keeping track
      5. XMLHTTPRequest Object:
         1. Can be in javascript to request data from server
         2. var rulesrequest = new XMLHTTPRequest(); \*outlookweb access
            1. rulesrequest.onreadystatechange = function() {

if(……){

$scope.rules = JSON.parse(rulesrequest.responseText);}}

Rulesrequest.open(“GET”, “http://......)

Rulesrequest.send();

* + - * 1. standard for web applications \*go out and get data
        2. jQuery has a wrapper since these can be complex aka AJAX
        3. Angular has it’s own wrapper
  1. External Data and $http \* service
     1. Get and send data to API
     2. Inject object….
     3. $http.get(‘/api’)
        1. .success(function) (result) what ever it got back from calling URL
        2. .error(function) (data, status code) {
           1. Console.log(data);}
  2. Angular Aside –
     1. Multiple controllers, Multiple views
        1. Within a separate controller, there are different views, these are unique per controller.
        2. Single page applications.
  3. Routing, Templates, Controllers
     1. $location , $log
     2. Routing: how we tell angular we’re using a service based on dependencies
        1. Var myApp = angular.module(module, [depedenices]);
           1. $routeProvider

.when(‘/’,{ templateUrl: ‘pages/main.htm’ controller: ‘mainController’} \*\*can do for multiple pages, controllers’’

* 1. Angular Singletons:
     1. Singleton- the one and only copy of an object.\*\*\*]
        1. AngularJs services are singletons \*routeParams, log, own service
        2. Angular chooses to have singletons
        3. $scope is an exception. “child scope”
        4. Still in same memory space on single page app \*power of Angular
  2. JS/Angular Side (Variable names, Normailzation)
     1. Normalize: to make consistent to a standard
        1. Text normalization
           1. Making different strings of text consistent to a standard
           2. Can’t use dashes in variable names in JS
           3. Html First-name in js firstname (camel case) \*normalized
           4. Learn-and-understand-angularkjs

Node-js:

Took javascript and run on pc!

Normally on browers

Took GoogleChrome’s V8 engine and run on pc. Listen to lots of things! \*ROR or PHP equivalent

\*people build utilities for pc. Life reload

\*Event driven, click, or form submission?

\*non-blocking, two-way connection

\*real time applications \*\*handle lots of connections simultaneously, Walmart or paypal

\*Challenges:

1. Doesn’t run on traditional apache server

\*Heroku is best node-js

\*requires a lot of installations (node, express, heroku, ect.)

\*add modules to expand capabilities

\*not for creating websites, but for handling server tasks

\*how to work with express.js