Angular JS has directives. With the prefix ng.

They are used as an attribute, element, comment, parameter, and a css element.

Whatever comes between ng-app is where all your angularjs directives should be placed.

AngualrJS can be seen as either MVC or MVVM.

This is due to the controller in angular not really doing anything,

Everything from model is mapped to the view directly in a way through the scope. And the scope is considered the VM (View-Model).

The view logic, model, and data logic should be separate with web frameworks.

Angular has a concept of a **watch** that synchornizes everything with the scope.

The watch will slow the web app drastically when there are too much watches running at the same time. So large enterprise apps aren’t built with AngularJS.

One time binding will disable the watch in the background.

When using expressions, it is better to use ng-bind to evaluate then {{ expression }}

Scope is the glue between Controller and View.

Data Binding:

One –way Data Binding: Model to View communication (Only one way changes)

Two way Data Binding: View to Model, and Model to view changes.

One –time Data Binding: updates info for only the first time so that watches get disabled

IT is good practice to use one time binding every time and only use others when values don’t need keep being updated.

Each logic group is considered its own app. So that way the whole web apps bindings are not loaded.

Instead each logic. So AngularJS uses spa (single page application). And each spa has only one module. Other dependent modules can be added as secondary modules.

Data binding is being done by the watch. It matches the values in the scope with the model and the view.

Directives:

Extends html

Lets you create your own elements.

Used as attributes and as tags.

We can create services to handle common ajax calls.

When an angular app runs there is the config phase and then the run phase.

So if you call the module.config it will run first, and then the module.run will run second.

DAY 2

Bower installs packages that are pushed to production. Things that are important to angular.

NPM is a superset of Bower, but it is used to install things that are development tools. They are not pushed to production.

Day 3

The angular compiler will read your index first, then find all the directives used, then find their definitions.

There is also a process called normalization. It removes the data- in front of attributes and also removes – from attributes to make them camelCase.

**April 24**

Templates in Directive object modules can be in template or templateURL. Where template url refereces an id with an ending of html.

Scope:

Child is controller, and parent is directive.

There is controller scope and directive scope.

If scope is true, the directive can access the controller, but the controller cant access the directives variables. (New variables created using the directives scope.)

If scope is false (default), the controller can access the directive.

Isolate never inherits anything from parents, and parent cant use child.

Isolate scope is in curly braces for the scope parameter in the Directive object model.

Values can still be passed through the attributes of that directive.

Isolate is useful because it lets you use that directive in other controllers and pass the object in the attributes of that directive. It makes it de coupled from the current controller.

Defined by {} as the scope in the template.

Can pass objects/variables through parameters

April 25, 2017

Link functions: defined in the directive object model

All the functions have 4 parameters, scope, element, attributes, controller , and transclude. Compile functions can’t have a scope parameter. THE ORDER IS IMPORTANT.

Order of operation

Compile, controller, pre-link, and post-link

By default, link is executed in post link.

If you are a defining a compile function, the link functions must be written within (unless it returns this.link), while if you don’t have a compile function, it can be defined on its own as an attribute in the directive object model.

If you are executing the same directive nested in another , the order we get is,

<div log=’parent’>

<div

Compile is always executing first, for every directive. Then the parents controller and pre-link, ten the childs controller, pre-link, and postlink. Then lastly the parents post-link is executied.

Therefore the order moves to the child after the parents pre-link.

So final order, Compile for every directive, then the parents controller and prelink, the all the childs stuff, then the parents pre-link.

<https://jsfiddle.net/Ammudhan1436/avymd4p5/2/>

Scope can not be used in the compile phase because it is not yet defined.

The element parameter passed in the compile function is the source element. If we want to ng-repeat on a directive, the compile function is run once, while post and pre link will run multiple times.

Ex. If we ng-repeat a directive 100 times, compile runs for the first element and then link gets repeated all 100 times. The first element is called the source element.

April 26, 2017

Looking at fido site.

CMS calls are being done on a header directive and footer directive.

They use ajax calls to gain info on the header and footer views of the site. There is also a default value

May 16, 2017

Routing:

routeProvider vs stateProvider

routeProvider cant display 2 pages in its ng-view, while stateProvider can.

stateProvider:

lets you change the urls configuration much easier compared to routeProvider.

May 17, 2017

https://docs.angularjs.org/guide/providers

Angular Recipies:

Service is considerd a angular recipe along with:

Factory, value, provider, constant.

They are all considered Providers of different variations.

There are 2 injectors. (Look up injectors)

In config phase, you append Provider at the end, in the run phase you use the name of the provider directly.

Again, there are 2 phases. Config phase and run phase. All Providers can be accesed differently in the config phase then the run phase. Variables are not available during the config phase.

The only 2 that can be accesed in the config phase directly, are constant and providers.

Constant values can be changed only in the config phase.

In providers, a mandatory function is this.$get = function();

If you separate config and run phase. Many per running logic needs to be run in the config phase and not the run phase. Example: if things need to be