CampNG Advanced

The plan

- Review Take-home solution
- Testing
- Directives, directives, directives
- Angular2 preview

We’ve got both kinds of testing!

Mocha

- BDD Style Unit testing framework
- Works in Node.js or browser
- Works with a variety of assertion styles

Chai

- supports expect assertions
- expect(something).to.equal(somethingElse)
- Language Chains: to, be, is, that, which, and, have, with
  - Basically noops

Chai matchers

- equal
- eql (deep)
- include
- a(type)
- true
- false
- not
- match
- any.keys, all.keys
- throw
- respondTo
- length

Setup and teardown

- beforeEach
- afterEach

karma

- Command line js test runner
- npm install -g karma-cli
- npm install chai
- karma start
- see karma.conf.js

Let’s see

Lab1

- checkout lab1_start
- Make calculator spec pass
- Fill in remaining specs
- Make them pass too

angular-mocks.js

- Extra goodness for testing angular
  - module - sets the module in a test case
  - inject - lets angular give you deps in your test case

$httpBackend

- From angular-mocks.js
- expectGET, expectPOST, etc
- whenGET, whenPOST, etc

CandidateService spec

Lab 2

- git checkout lab2_start
- add spec for CandidateService.getCandidate(id)
- Comment out getCandidate to see it fail properly

We have a problem

- Creating and editing candidate works
- But the list doesn’t always update :(
- I tried one solution, let’s look at another

Events

- Pub sub is built in
- Use methods on any $scope
- $rootScope can be injected anywhere

$broadcast/$emit

- $broadcast goes to this scope and down
- $emit goes to this scope and up
- both take an event name an args
- Easiest to do $rootScope.$broadcast

$on

- Subscribes to an event
- Pass name, event handler
- Event handler receives event, any args

Event example

protractor

- end to end testing for angular
- wrapper around web-driver

Installing protractor

- npm install -g protractor
- webdriver-manager update
- webdriver-manager start

protractor API

- browser
  - navigate using browser.get
- element
  - interact with elements on the page
  - click, sendKeys

protractor API

- locators - for finding elements
- by
  - css
  - model
  - buttonText
  - repeater

Protractor example

- lab3_start
- features/candidates_spec.js

Running protractor

- webdriver-manager start
- protractor config/spec-e2e.js

Lab 3

- checkout lab3_start
- Update the side list when the menu updates
- Use events
- Inject $rootScope in service
- Inject $scope in controller
- Make the spec pass
- Need to restart server.js to be sure :(

Creating filters

- module.filter(“name”, function(…) {…}
- returns a function which takes param(s) and returns filtered value

$sce

- $sce.trustAsHtml
- $sce.getTrustedHtml

hello filter and spec

markdown

- Simple DSL for expressing formatted text
- used by Github, many others
- markdown.toHtml()

Lab 4

- checkout lab4_start
- npm install
- Make a markdown filter
- Use markdown.toHTML
- Use $sce to make it trusted
- spec should pass

ng-bind-html

- Sets the innerHTML of it’s element
- Uses $sanitize unless it receives trusted HTML
- Use in place of {{}} if you want to output HTML

Lab 5

- Use it!
- Add a resume field to candidates
- Edit it with a text-area
- Display rendered output below using your filter
- Don’t forget to use ng-bind-html

Creating directives

- module.directive(“name”, function…
- function can get dependencies injected
- returns a directive definition object

Directive naming

- strip any data- or x- prefix
- convert :, -, or _ separated to camelCase
- resulting name is use to find directive

Directive definition obj

- template or templateUrl
- restrict
    - E for element
    - A for attribute (default)
    - C for class
    - M for comment
    - any combination thereof

Hello directives

Your turn

- Back to jsbin
- Make a directive that lets create a <hello-myname> element
- Have it print out a greeting

Directive scope

- Turns out inheriting scope is a bad idea
  - Leads to coupling
- Isolate scope to the rescue
- Scope property with a mapping object

Mapping scope

- The “interface” to your directive
- Uses attributes to set in properties on directive scope
- Prefixes specify how to map attribute to scope property
- Can be abbreviated to “@“, “=“, “&” if scope property and attribute are the same

@

- foo: “@bar”
- The foo property of scope holds the string value of the bar attribute

=

- foo: “=bar”
- creates a scope property foo
- bound to parent scope property specified by bar attribute

&

- evaluates an expression in parent scope
- foo: “&bar”
- foo property becomes a function
- bar is the expr to evaluate

Hello scopes

display-markdown

Lab 6

- Extract a markdown editor directive
- Directive should:
  - Pass in the resume via scope mapping
  - Edit markdown in a textarea
  - Display the rendered markdown below
- Use it in form.html

link function

- link: function(scope, element, attrs, …)
- Called after the resulting element is in the DOM
- Do DOM manipulation or event listening here
- The right place for any jQuery calls
- But you need to $(element)

link example

raty

- A jquery plugin to do star ratings
- http://wbotelhos.com/raty
- Let’s see a demo!

Lab 7

- Make a starRating directive
- Use a link function
- Star images are available at /images
  - $(element).raty({path: “/images”})
- Add a rating property on edit candidate form
- Just get as far as plugin appearing

$scope.$watch

- I want angular to tell me when stuff changes
- Two args
  - An expression to watch
  - A function to execute on change
    - receives newval, oldval as params

$watch example

Lab 8

- Add a rating scope mapping
- watch the rating mapping using scope.
  $watch
- $(element).raty(“score”, newValue)
- feel free to add a temporary input bound to rating property

scope.$apply()

- Hey angular, something happened!
- Kicks off dirty checking, AKA the $digest cycle
- Useful to coordinate with external APIs
- Takes a function as optional arg
- $digest cycle happens after function executes

$apply example

Lab 9

- Pass a click function into raty
- In click function, call scope.$apply
- In function passed to apply, update rating property on scope
- Raty directive should work!

directive controllers

- controller
- controllerAs
- bindToController

controller example

require

- Attribute on directive specification object
- Allows directives to collaborate
- The controller of another directive gets passed as additional arguments to link

Require syntax

- “otherDirective” should be located on this element
- “^parentDirective” search for directive on parent elements
- “?maybeDirective” pass null if maybeDirective isn’t found instead of throwing error

require example

ngModelController

- Use require: “ngModel” to get handed an ngModelController
- Exposes form component API
- Manages model and view values
- Allows for building custom form controls that first class “angular form citizens”

ngModelController

- $render
  - set this to function which gets passed the view value to render.
- $setViewValue
  - Call this to update the model controllers view value
- $viewValue accesses it

ngModel example

Lab 10

- Turn raty into a “real” angular form control
- Require ngModel and get passed an ngModelController in link
- Set a $render function on ngModel controller to call raty
    - Use controller.$viewValue to access
- Call $setViewValue from click handler to update ng-model

Transclusion

- The inclusion of one thing in something else
- transclude: true
- Allows directive to wrap arbitrary angular template content
- Use ng-transclude to specify where the original body content goes

Transclusion example

Lab 11

- Build a dollar input group directive
- Wrap a normal input with twitter bootstrap input group formatting
- Use it to enter a candidates expected salary
- Use an element (restrict: “E”)

component

- New in 1.5
- Simpler API for some directives
- module.component(name, definition)
- takes a definition object rather than a function

component differences

- defaults to element
- bindings instead of scope
  - bindToController is true
- no link function

Component example

Lab 12

- Make dollar input a component

validation directives

- require ngModel
- $validators object on modelController
- Each property of $validatators is a validator function
  - key is validator name
  - function passed viewValue, modelValue
  - returns true or false

validator example

Lab 13

- Write an ssn directive
- Should validate ssn with regex and add an ssn error in case of failure
- Markup is there already in form.html
- /^\d{3}-?\d{2}-?\d{4}$/

Fin

Nested routes

- Name with “parentState.childState”
- url is relative to parent state
- Add a ui-view to parent template

Example me!

Lab 15

- Candidates now have comments
- List comments on the show candidate view
- Click on a comment title to view details
- Use nested routes make it happen!

Angular2

All about the web components

It’s ~~totally~~ different!

- Web components
- ES6/Typescript
- Routing
- Injection system
- bindings

It’s not done yet!

- Some of what I will show you is already obsolete!
- Even more of it will become obsolete soon!
- Yay! Let’s get started!

Angular2

- git clone:
- https://github.com/gaslight/angular2_labs

```
npm install -g http-server
```

SystemJS

App component

ES6 Modules

Typescript

bootstrap

Woot! We can haz angular2

You try!

- git checkout start
- Make your app element <name>-app
- Where <name> is your name
- Have it say something!

Components

- The class is the “controller”
- It is in scope in the template
- Expressions are still {{}}

Let’s see!

You try!

Nested components

Super weird for loop syntax

You try!

attribute bindings

Make a sub subcomponent

Event bindings

Add a button!