CampNG Advanced

Intros

The plan

- ES6 Crash Course
- Webpack
- Mocha/Chai
- Directives, directives, directives
- UI Router

The unplanned

-
-

NG6

- Starter repo using Angular(1), ES6, and web pack
- Thin gulp wrapper

Webpack

- module bundler
- manages dependencies
- rich ecosystem loaders and plugins

Let’s git started

- http://github.com/gaslight/campng-advanced-es6
- Copy clone URL
- git clone <url>

Next steps

- git checkout start
- npm install
- npm install -g gulp
- gulp
- http://localhost:3000

ES6 Modules

- Reference external libraries
- Precise control of namespacing
- Module = file
- Loaders are a separate spec :(

Export

- Multiple named exports
- One default export
- allows for renaming

Import

- Named
- Default
- Allows for renaming

Quick tour

Example time

Crash ES2015 course

- classes
- arrow functions
- let

Babel.js

Mocha

- Go to unit test framework for Angular
- Rspec/BDD style

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
- karma start
- see karma.conf.js

Let’s see

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

Let’s see some generated specs

$httpBackend

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

server.js

CandidateService spec

Lab 1

- git checkout lab1_start
- add spec for CandidateService.get(id)
- Make it pass

sinon

- spies, stubs, and mocks
- sinon.stub()
    - returns a function
    - we can ask how it was invoked
    - specify a return value

sinon example

sinon-stub-promise

- adds returnsPromise to stubs
- call resolves() or rejects() to specify behavior
- makes test synchronous

controller spec

UI Router

- A replacement for ngRouter
- Allows for nested and multiple views

Differences from ngRoute

- ngRouter => ui.router
- $routeParams => $stateParams
- ng-view => ui-view

$stateProvider

- very close to $routeProvider API
- when => state
- state(stateName, options)
- url goes in options

ui-sref

- directive to build links
- pass a state name
  - ui-sref=“state”
- with params
  - ui-sref=“state({id: 1})”

Let’s see!

Lab 2

- See spec for ShowController
- Make it pass
- Be sure to give it a $stateParams with an id
- Add a show template to display candidate’s name

Creating candidates

Lab 3

- Editing candidates
- spec and impl. for CandidateService#update
- spec and impl for EditController
- Edit Candidate link on show

We have a problem

- Creating and editing candidate works
- But the list doesn’t update :(

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

protractor API

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

protractor API

- locators - for finding elements
- by
  - id
  - css
  - binding
  - model

Protractor example

- lab3_start
- spec-e2e/candidates_spec.js

Running protractor

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

Installing protractor

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

Lab 4

- checkout lab4_start
- Update the side list when the menu updates
- Use events
- Make the spec pass
- Need to restart server.js to see it fail :(

Creating filters

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

$sce

- $sce.trustAsHtml
- $sce.getTrustedHtml

checkout lab5_start

hello filter and spec

markdown

Lab 5

- checkout lab5_start
- npm install
- Make a markdown filter
- Write a spec for it
- Use markdown.toHTML
- Use $sce to make it trusted

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

- 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

directive controllers

- controller
- controllerAs
- bindToController

Components

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 6

- Make a markdown editor directive
- gulp component —name markDown
- Directive should:
  - Pass in the markdown via scope mapping
  - Edit markdown in a textarea
  - Display the rendered markdown below
- Add it to edit

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 raty directive
- Use a link function
- Star images are available at /images
  - {path: “/images”}
- Just get as far as plugin appearing

$scope.$watch

- 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
- call $().raty(“score”, newValue)
- Use an input to check your work

scope.$apply()

- Triggers a $digest() cycle
  - This is where dirty checking happens
  - Normally not needed
  - Except to coordinate with external APIs
- Takes a function as arg

$apply example

Lab 9

- Pass a click function into raty
- Use scope.$apply to update the rating property and have angular know about it
- Raty directive should work!

require

- Allows directives to collaborate
- Specifies a directive this directive needs
  - “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
- required directives have their controller passed to requiring directive

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

Lab 10

- Turn raty into a “real” angular form control
- Require ngModel and get passed an ngModelController
- User $render to update raty
- Use $setViewValue to update ng-model

$formatters

- pipeline (array) of functions
- Converts model values to view value
- Each function gets passed model value, returns view value

$parsers

- pipeline (array) of functions
- Converts view value to model value
- Each function gets passed view value, returns model value

parse/format example

Lab 11

- Let’s make raty convert from percentage
- Model value is 0 to 100
- Stars are 0 to 5

Lab 12

- Edit is looking good, but what about new?
- Make a candidate-form component
- Use it on new and edit

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 13

- Build a dollar input group directive
- Wrap a normal input with twitter bootstrap input group formatting
- Use it to enter a candidates expected salary

validation directives

- require ngModel
- $setValidity(validationErrorKey, isValid)
- called during $parsers function

Lab 14

- Write an ssn directive
- Should validate ssn with regex and add an ssn error in case of failure
- Use it on edit and display appropriate error message

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!