Internet Programming

Practice #11

AngularJS - Lab 2

Dept. of Software and Information Systems Engineering

by Shir Frumerman Base on Hasidi Netanel Slides

Today

- Some more built-in directives
- Routing
- Forms

Built in Directives

- AngularJS built-in directives list and examples :

https://www.w3schools.com/angular/angular_ref_directives.asp

Built in Directives

- ng-show / ng-hide

  Used to show or hide a given element based on expressions, booleans and functions

  Examples on : scotch.io/tutorials/how-to-use-ngshow-and-nghide

- ng-click

  Used to fire a method or expression when element is clicked.

Built in Directives

- ng-options

  The ng-options directive fills a <select> element with <options>.

  The ng-options directive uses an array to fill the dropdown list.

```
<select ng-model="selectedName" ng-options="item for item in names"></select>
```

Routing

- ngRoute - handle routing
- In single page app url is called *hashbang* URL
- Traditional URL: http://www.myApp.com/first/page
- Hashbang URL: http://www.myApp.com/#/first/page
- When the hash fragment changes, the JavaScript responds and loads only the relevant data and HTML – faster app

Routing

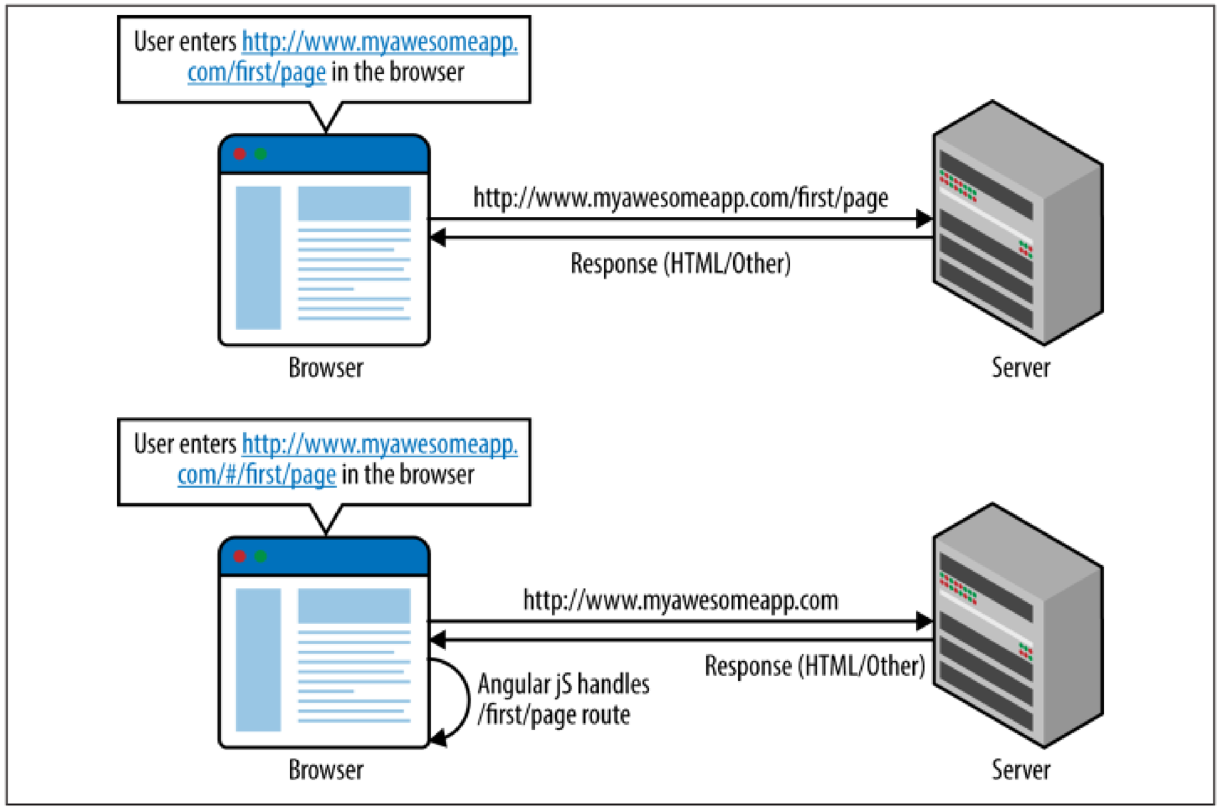

User enters http://www.myawesomeapp.com/first/page in the browser
http://www.myawesomeapp.com/first/page
Response (HTML/Other)
Browser
Server
User enters http://www.myawesomeapp.com/#/first/page in the browser
http://www.myawesomeapp.com
Response (HTML/Other)
Angular jS handles /first/page route
Browser
Server

Figure 10-1. Flow of normal URLs versus hash URLs

Taken from : AngularJS: Up And Running by Shyam Seshadri and Brad Green

Routing - set up

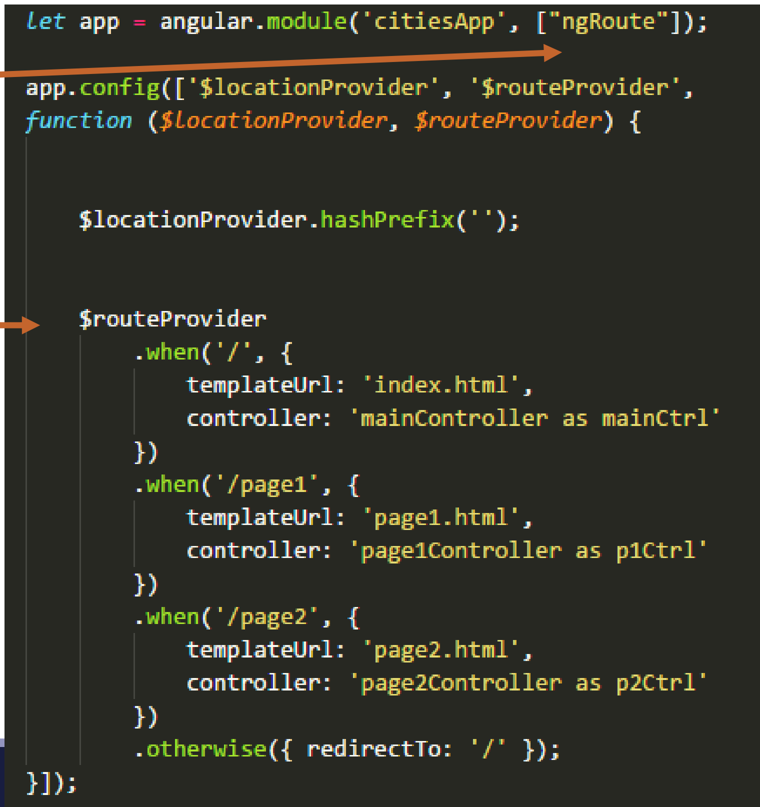
- AngularJS routing is not part of the core library
  Include in index.html :

```
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular-route.js"></script>
```

Include the module as a dependency of our main AngularJS app module :

Define our routes in the config section using the $routeProvider service

- Define for each view it’s controller



```
let app = angular.module('citiesApp', ["ngRoute"]);

app.config(['$locationProvider', '$routeProvider',
function ($locationProvider, $routeProvider) {


    $locationProvider.hashPrefix('');


    $routeProvider
        .when('/', {
            templateUrl: 'index.html',
            controller: 'mainController as mainCtrl'
        })
        .when('/page1', {
            templateUrl: 'page1.html',
            controller: 'page1Controller as p1Ctrl'
        })
        .when('/page2', {
            templateUrl: 'page2.html',
            controller: 'page2Controller as p2Ctrl'
        })
        .otherwise({ redirectTo: '/' });
}]);
```

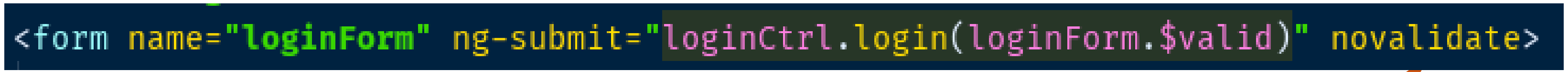
Routing - ng-view

- ng – view : Mark which section of the page AngularJS should change when the route changes
- AngularJS application that uses ngRoute, there can be one and only one ng-view directive for that application
- Only the content inside the ng-view tags will be changed

```
<div ng-view></div>
```

Working with Forms

- heavily leverage the ng-model directive to get our data into and out of the form
- We use the '**ng-submit**' directive to trigger the submission functionality.
- Handle forms states and validate easily



```
<form name="loginForm" ng-submit="loginCtrl.login(loginForm.$valid)" novalidate>
```

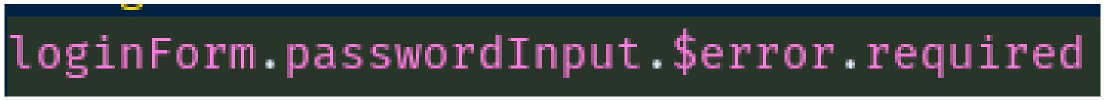
This will prevent the default HTML5 validations (since we'll be doing that ourselves)

Form States

- AngularJS creates a FormController that holds the current state of the form as well as some helper methods
- Access the FormController for a form using the form's name
- Each of the states (except $error) are Booleans and can be used to conditionally hide, show, disable, or enable HTML elements in the UI

Form States

- $valid: True if a form item is valid.
- $invalid: The opposite of the above.
- $pristine: True if the form/input **has not** been used yet.
- $dirty: True if the form/input has been used.
- $touched: True if an input has lost focus.
- $error: Expression that checks if an input has an error.

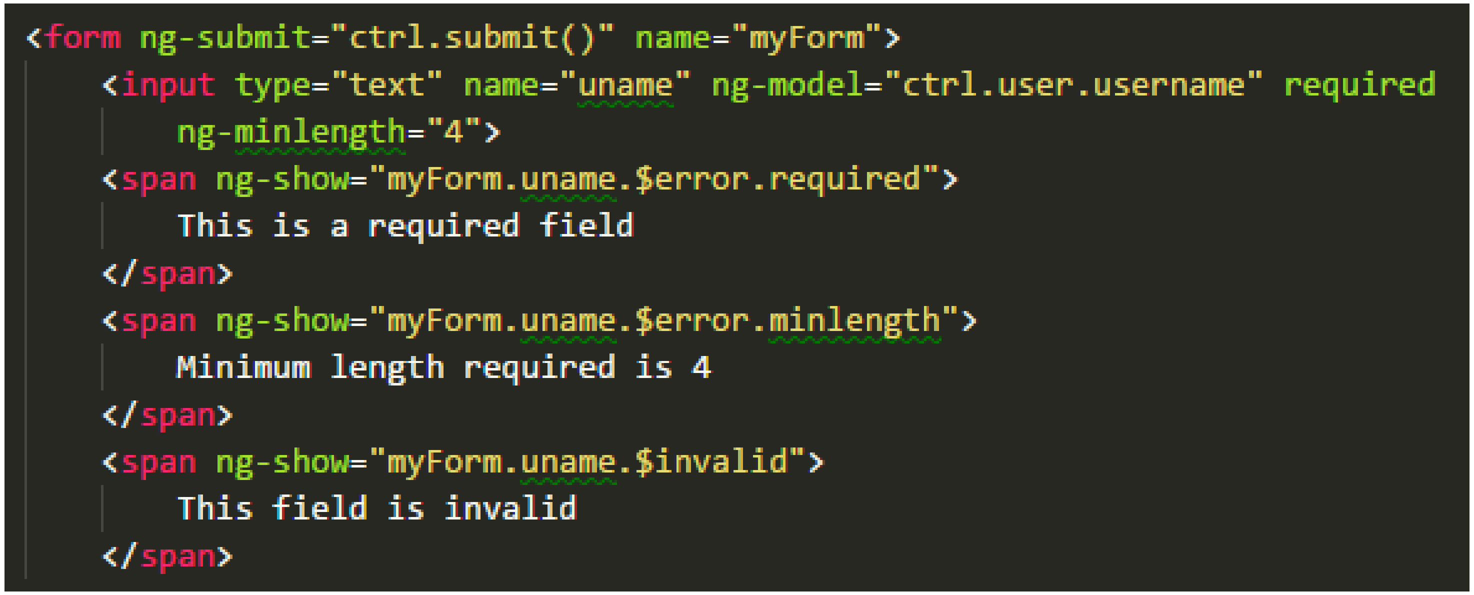


```
loginForm.passwordInput.$error.required
```

Form Directives

- ng-required: required by expression
- ng-disabled: disable the submit button when our form is not valid.
- ng-minlength: controls the min length of the input.
- ng-maxlength: controls the max length of the input.
- ng-pattern: regular expression validation.

Form - Displaying Error Messages



```
<form ng-submit="ctrl.submit()" name="myForm">
    <input type="text" name="uname" ng-model="ctrl.user.username" required
        ng-minlength="4">
    <span ng-show="myForm.uname.$error.required">
        This is a required field
    </span>
    <span ng-show="myForm.uname.$error.minlength">
        Minimum length required is 4
    </span>
    <span ng-show="myForm.uname.$invalid">
        This field is invalid
    </span>
```

Lab Excersice

1. Open “routing“ file in Examples folder in VSCode
2. Add a new button in the navigation bar to Point of interests page
3. Add the sol form EX1 to the routing directory (component/POI/)
4. Clean the redundant code, edit all the files in order to combine the new files to the existed ones
5. Add a new button in the navigation bar to : Register
6. Create a form with validation according to our register form of the project
7. Add register controller that log to console the form details
8. Set the routing to the form view and controller

Any Questions?

And I’ll come shortly..