# Angular JS in Details

Unit#3

Marwadi University

Department of Information Technology

Advanced Web Programming (3161611)

Tejas Chauhan

# Highlights

- Modules
- Directives
- Routes
- Angular JS Forms and Validations
- Data binding
- Creating single page website using Angular JS

- With this we will also cover: Routes and Services



# Modules

- AngularJS supports modular approach.
- Modules are used to separate logic such as services, controllers, application etc. from the code.
- We define modules in separate js files and name them as per the module.js file.
- Let's see some examples:
  - Application Module – used to initialize an application with controller(s).
  - Controller Module – used to define the controller.



# Application Module

- A file named mainApp.js that contains the following code –

```
var mainApp = angular.module("mainApp", []);
```

- We declare an application mainApp module using angular.module function and pass an empty array to it.
- This array generally contains dependent modules.



# Controller Module

- Add a controller to your application:

```
mainApp.controller("myCtrl", function($scope)
{
   $scope.firstName = "John";
   $scope.lastName = "Doe";
});
```

- We declare a controller myCtrl module using mainApp.controller function.

# Use Modules

- Application module using ng-app directive.
- Controller using ng-controller directive.
- Import the mainApp.js and myCtrl.js in the main HTML page.

```
<div ng-app = "mainApp" ng-controller="myCtrl">

   {{ firstName + " " + lastName }}

   <script src = "mainApp.js"></script>

   <script src = "myCtrl.js"></script>

</div>
```



# Scope

- The scope is the binding part between the HTML (view) and the JavaScript (controller).
- The scope is an object with the available properties and methods.
- The scope is available for both the view and the controller.
- If we consider an AngularJS application to consist of:
  - View, which is the HTML.
  - Model, which is the data available for the current view.
  - Controller, which is the JavaScript function that makes/changes/removes/controls the data.
- Then the scope is the Model.



# Root Scope

- All applications have a $rootScope which is the scope created on the HTML element that contains the ng-app directive.
- The rootScope is available in the entire application.
- If a variable has the same name in both the current scope and in the rootScope, the application uses the one in the current scope.

```
<script>
var app = angular.module('myApp', []);
app.run(function($rootScope) {
  $rootScope.color = 'blue';
});
app.controller('myCtrl', function($scope) {
  $scope.color = "red";
});
</script>
```



# Directives

- AngularJS directives are used to extend HTML. They are special attributes starting with ng-prefix.
  - **ng-app** – This directive starts an AngularJS Application.
  - **ng-model** – This directive defines the model that is variable to be used in AngularJS.
  - **ng-bind** – This directive will bind the innerHTML of the element to the specified model property.
  - **ng-init** – This directive initializes application data.
  - **ng-repeat** – This directive repeats HTML elements for each item in a collection.



# ng-app directive

- The ng-app directive starts an AngularJS Application.
- It defines the root element for application.
- It automatically initializes or bootstraps the application when the web page containing AngularJS Application is loaded.
- It is also used to load various AngularJS modules in AngularJS Application.

```
<html ng-app = "">
    ...
</html>
```



# ng-model directive

- The ng-model directive defines the model/variable to be used in AngularJS Application.

```
<div ng-app = "">

    ...

    <p>Enter your Name: <input type="text"
ng-model="name"></p>

</div>
```



# ng-bind directive

- The view has access to the model, and there are several ways of displaying model data in the view.
- The ng-bind directive will bind the innerHTML of the element to the specified model property.

```
<p ng-bind="firstname"></p>
```

- You can also use double braces {{ }} to display content from the model.

```
<p>First name: {{firstname}}</p>
```



# ng-init directive

- The ng-init directive initializes an AngularJS Application data.
- It is used to assign values to the variables.

```
<div ng-app="" ng-init="countries = [{
      locale:'en-US',
      name:'United States'
    }, {
      locale:'en-GB',
      name:'United Kingdom'
    }, {
      locale:'en-FR',
      name:'France'
    }]">
     ...
</div>
```



# ng-repeat directive

- The ng-repeat directive repeats HTML elements for each item in a collection.

```
<div ng-app = "">
    ...
    <p>List of Countries with locale:</p>
    <ol>
        <li ng-repeat = "country in
countries">
            {{ 'Country: ' + country.name +
', Locale: ' + country.locale }}
        </li>
    </ol>
</div>
```



# ng-click directive

- The ng-click directive is similar to onClick event.

```
<div ng-app = "">
    ...
    <button ng-click="clickFun()">Click
Me</button>
</div>
```

# directives

- There are number of directives in AngularJS:
  - ngApp, ngBind, ngBindHtml, ngBindTemplate, ngBlur, ngChange, ngChecked, ngClass, ngClassEven, ngClassOdd, ngClick, ngCloak,ngController, ngCopy, ngCut, ngDblclick, ngDisabled, ngFocus, ngForm, ngHide, ngHref, ngIf, ngInit , ngKeydown, ngKeypress, ngKeyup, ngList, ngMaxlength, ngMinlength, ngModel, ngModelOptions, ngMousedown, ngMouseenter, ngMouseleave, ngMousemove, ngMouseover, ngMouseup, ngOn, ngOpen, ngOptions, ngPaste, ngPattern, ngReadonly , ngRef, ngRepeat, ngRequired, ngSelected, ngShow, ngSrc, ngStyle, ngSubmit, ngSwitch, ngValue, etc…



# Create New Directives

- In addition to all the built-in AngularJS directives, you can create your own directives.
- New directives are created by using the .directive function.
- To invoke the new directive, make an HTML element with the same tag name/Attribute/Class/Comment as the new directive.
- When naming a directive, you must use a camel case name, myExampleDirective, but when invoking it, you must use - separated name, my-example-directive.



# Create New Directives

```
<body ng-app="myApp">
<my-example-directive></my-example-direct
ive>

<script>
var app = angular.module("myApp", []);
app.directive("myExampleDirective",
function() {
   return { template : "<h1>Say hello to a
   newly created directive!</h1>" };
});
</script>

</body>
```



# Invoke the new directive

- To invoke the new directive, make an HTML element with the same tag name/Attribute/Class/Comment as the new directive.
  - <my-example-directive></my-example-directive>
  - <div my-example-directive>
  - 
  - <!-- directive: my-example-directive -->



# Invoke the new directive

- You can restrict your directives to only be invoked by some of the methods.
- By adding a restrict property with template.
- Values of restrict property:
  - E for Element name
  - A for Attribute
  - C for Class
- Default the value is EA, meaning that both Element names and attribute names can invoke the directive.



# Filters

- AngularJS provides filters to transform data:
  - uppercase: Format a string to upper case.
  - lowercase: Format a string to lower case.
  - number: Format a number to a string.
  - currency: Format a number to a currency format.
  - date: Format a date to a specified format.
  - filter: Select a subset of items from an array.
  - json: Format an object to a JSON string.
  - limitTo: Limits an array/string, into a specified number of elements/characters.
  - orderBy: Orders an array by an expression.



# Filters: Examples

```
<p>The name is {{ firstName | uppercase
}}</p>

<p>Price: {{ date | currency : symbol :
fractionSize}}</p>

<p>{{ bDate | date : format : timezone}}</p>
<li ng-repeat="x in names |
orderBy:'country'">

<li ng-repeat="x in names | filter :
expression : comparator">
```



# Services

- A service is a function, or object, that is available for, and limited to, your AngularJS application.
- You can make your own service, or use one of the many built-in services.
- AngularJS has about 30 built-in services.
- Examples: $window, $location, $http, $timeout, $interval, $route, etc…
- In order to use the service in the controller, it must be defined as a dependency - need to be passed in to the controller as an argument.



# $location service

- AngularJS constantly supervises your application, and for it to handle changes and events properly, AngularJS prefers that you use the $location service instead of the window.location object.
- Use the $location service in a controller:

```
var app = angular.module('myApp', []);
app.controller('customersCtrl',
function($scope, $location) {
    $scope.myUrl = $location.absUrl();
});
```



# $timeout Service

- The $timeout service is AngularJS' version of the window.setTimeout function.
- Display a new message after two seconds:

```
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope,
$timeout) {
  $scope.myHeader = "Hello World!";
  $timeout(function () {
    $scope.myHeader = "How are you
today?";
  }, 2000);
});
```



# $interval Service

- The $interval service is AngularJS' version of the window.setInterval function.
- Display the time every second:

```
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope,
$interval) {
  $scope.theTime = new
Date().toLocaleTimeString();
  $interval(function () {
    $scope.theTime = new
Date().toLocaleTimeString();
  }, 1000);
});
```



# $http Service

- The $http service is one of the most common used services in AngularJS applications. The service makes a request to the server, and lets your application handle the response.
- Use the $http service to request data from the server:

```
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope,
$http) {
  $http.get("welcome.htm").then(function
(response) {
    $scope.myWelcome = response.data;
  });
});
```



# $http Service

- The .get method is a shortcut method of the $http service.
- There are several shortcut methods:
  - .delete()
  - .get()
  - .head()
  - .jsonp()
  - .patch()
  - .post()
  - .put()



# $http Service

- To handle errors, add one more functions to the .then method:

```
app.controller('myCtrl', function($scope,
$http) {
  $http({
    method : "GET",
    url : "welcome.htm"
  }).then(function(response) {
    // First function handles success
    $scope.content = response.data;
  }, function(response) {
    // Second function handles error
    $scope.content = "Something went wrong";
  });
});
```



# $http Service

- The response from the server is an object with these properties:
  - **.config** the object used to generate the request.
  - **.data** a string, or an object, carrying the response from the server.
  - **.headers** a function to use to get header information.
  - **.status** a number defining the HTTP status.
  - **.statusText** a string defining the HTTP status.



# Custom Service

- There are two ways to create a service:
  - Factory
  - Service



# Using Factory Method

- In this method, we first define a factory and then assign method to it.

```
var mainApp = angular.module("mainApp", []);
mainApp.factory('MathService', function() {
   var factory = {};

   factory.multiply = function(a, b) {
      return a * b
   }

   return factory;
});
```



# Using Service Method

- In this method, we define a service and then assign method to it. We also inject an already available service to it.

```
mainApp.service('CalcService',
function(MathService) {
    this.square = function(a) {
        return MathService.multiply(a,a);
    }
});
```



# Forms

- AngularJS enriches form filling and validation.
- We can use ng-click/ng-submit event to handle the button/form and use $dirty and $invalid flags to do the validation in a seamless way.
- Use novalidate with a form declaration to disable any browser-specific validation.
- The form controls make heavy use of AngularJS events.



# Forms

- Input controls:
  - input elements
  - select elements
  - button elements
  - textarea elements
- Data-Binding
  - using the ng-model directive



# Form Validations

- AngularJS monitors the state of the form and input fields, and lets you notify the user about the current state.
- AngularJS also holds information about whether they have been touched, or modified, or not.
- You can use standard HTML5 attributes to validate input, or you can make your own validation functions.



# HTML DOM - Directives

- The following directives are used to bind application data to the attributes of HTML DOM elements –
  - **ng-disabled:** disables a given control.
  - **ng-show:** shows a given control.
  - **ng-hide:** hides a given control.
  - **ng-click:** represents a AngularJS click event.

# Form Validations

- The following can be used to check status:
  - **$untouched:** The field has not been touched yet
  - **$touched:** The field has been touched
  - **$pristine:** The field/form has not been modified yet
  - **$dirty:** The field/form has been modified
  - **$invalid:** The field/form content is not valid
  - **$valid:** The field/form content is valid
  - **$submitted:** The form is submitted



# CSS Classes

- AngularJS adds CSS classes to forms and input fields depending on their states.
  - **ng-untouched:** The field has not been touched yet
  - **ng-touched:** The field has been touched
  - **ng-pristine:** The field has not been modified yet
  - **ng-dirty:** The field has been modified
  - **ng-valid:** The field content is valid
  - **ng-invalid:** The field content is not valid
  - **ng-valid-key:** One key for each validation. Example: ng-valid-required, useful when there are more than one thing that must be validated
  - **ng-invalid-key:** Example: ng-invalid-required



# Form Validations

- Example:

```
<form name="subscribeForm" novalidate>
    <input type="email" name="subscribeEmail"
ng-model="subscribeEmail" required>
    <button type="submit" ng-disabled=
"subscribeForm.subscribeEmail.$invalid">Subscribe
</button>
    <div ng-show=
"subscribeForm.subscribeEmail.$touched &&
subscribeForm.subscribeEmail.$invalid"
style="color:red">
      <span ng-show=
"subscribeForm.subscribeEmail.$error.required">Email
id is required.</span>
      <span ng-show=
"subscribeForm.subscribeEmail.$error.email">Please
enter valid email id.</span>
    </div>
</form>
```



# Basics of routes and navigation

- When you want your user to navigate to different pages in your application, you send different http requests to your server.
- Routing refers to determining how your application responds to a client request.
- Routing is the mechanism by which requests (as specified by a URL and HTTP method) are routed to the code that handles them.
- To put simply, in the Router you determine what should happen when a user visits a certain page.



# Routing in AngularJS

- If you want to navigate to different pages in your application, but you also want the application to be a SPA (Single Page Application), with no page reloading, you can use the ngRoute module.
- This routing module acts based on the url.
- When a user requests a specific url, the routing engine captures that url and renders the view based on the defined routing rules.
- The ngRoute module routes your application to different pages without reloading the entire application.



# Routing in AngularJS

- To make your applications ready for routing, you must include the AngularJS Route module:

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

- Then you must add the ngRoute as a dependency in the application module:

```
var app = angular.module("myApp",
["ngRoute"]);
```

- Now your application has access to the route module, which provides the $routeProvider.



# Routing in AngularJS

- With the $routeProvider you can define what page to display when a user clicks a link.
- $routeProvider is defined as a function under config of mainApp module using key as '$routeProvider'.
- $routeProvider.when defines a URL "/addStudent", which is mapped to "addStudent.htm".
- "otherwise" is used to set the default view.
- "controller" is used to set the corresponding controller for the view.



# Routing in AngularJS

```
app.config(function($routeProvider) {
  $routeProvider
  .when("/", {
    templateUrl : "home.html"
  })
  .when("/student/:id", {
    templateUrl : "student.html",
    controller : "studentController"
  })
  .otherwise({
    template : "<h1>404</h1><p>Page not
found.</p>"
  });
});
```



# Routing in AngularJS

- Your application needs a container to put the content provided by the routing.
- This container is the ng-view directive.
  - <div ng-view> OR
  - <ng-view></ng-view> OR
  - 
- Applications can only have one ng-view directive, and this will be the placeholder for all views provided by the route.



# Creating single page website using Angular JS

- Let's create an 'AddressBook'.



# My Address Book

Filter: Name City State

## Honorable Angela I...

aivell0@blogtalkradio.com
6269099421

## Ms Dorri Dalling

ddalling1@bbb.org
3143442309

## Ms Fabien Semmens

fsemmens2@tamu.edu
3053386842

## Ms Tucky Joutapaitis

tjoutapaitis3@ftc.gov
8643336478

## Dr Gloria Tanman

gtanman4@redcross.org
2107958791

## Rev Candi Austwick

caustwick5@ed.gov
4125503125

## Dr Lauren Brahmer

lbrahmer6@oaic.gov.au
3108315418

## Dr Stevana Barrand

sbarrand7@bloglines.com
7163231225

## Rev Daryl Pegden

dpegden8@friendfeed.com
5714409559

## Honorable Cate Co...

ccoddrington9@kickstarter.c...
2518506680

## Mrs Zarla Allcorn

zallcorna@freewebs.com
2016782508

## Ms Boote Ciepluch

bciepluchb@taobao.com
4028178893

## Rev Auberon Tasseler

atasselerc@msu.edu
9491138834

## Ms Gigi Claige

gclaiged@soup.io
5166844957

## Dr Dare Anstiss

danstisse@wunderground.com
2063598114

# My Address Book

Filter: Name City State

## Contact List

| Sr.No. | Person | City | State | Country | Action |
|---|---|---|---|---|---|
| 1 | Dr Dare Anstiss | Seattle | Washington | United States | |
| 2 | Ms Gigi Claige | Hicksville | New York | United States | |
| 3 | Rev Auberon Tasseler | Irvine | California | United States | |
| 4 | Ms Boote Ciepluch | Omaha | Nebraska | United States | |
| 5 | Mrs Zarla Allcorn | Paterson | New Jersey | United States | |
| 6 | Honorable Cate Coddrington | Mobile | Alabama | United States | |
| 7 | Rev Daryl Pegden | Falls Church | Virginia | United States | |
| 8 | Dr Stevana Barrand | Buffalo | New York | United States | |
| 9 | Dr Lauren Brahmer | Inglewood | California | United States | |
| 10 | Rev Candi Austwick | Pittsburgh | Pennsylvania | United States | |
| 11 | Dr Gloria Tanman | San Antonio | Texas | United States | |
| 12 | Ms Tucky Joutapaitis | Spartanburg | South Carolina | United States | |

# Contact Details

## Dr Dare Anstiss

**Email**

danstisse@wunderground.com

| | |
|---|---|
| **Mobile1** | 2063598114 |
| **Mobile2** | 4047112557 |

## Other Details

Edit Delete

**Address:**

5829 Jay Park
0 Orin Terrace
City : Seattle
State : Washington
Postal/Zip code : 98109
Country : United States

**Note**

Swedish

AddressBook Contacts Add New

# Edit Contact

## Update Employee Details

Title: Dr

*First Name: Dare

*Last Name: Anstiss

*Mobile1: 2063598114

Mobile2: 4047112557

*Email: danstisse@wunderground.com

*Address Line 1: 5829 Jay Park

*State: Washington

Address Line 2: 0 Orin Terrace

Postal/Zip Code: 98109

*City: Seattle

*Country: United States

Save Discard

AddressBook Contacts Add New

# Add Contact

## Add Employee Details

Title
Mr/M

*First Name:
First Name

*Last Name:
Last Name

*Mobile1:
9898989898

Mobile2:
9898989898

*Email:
test@test.com

*Address Line 1:

*State:

Address Line 2:

Postal/Zip Code:

*City:
City

*Country:

Save

Discard

# Review

- Modules
- Directives
- Routes
- Services
- Angular JS Forms and Validations
- Data binding
- Creating single page website using Angular JS



# Thank You.