CS 572 Modern Web Applications

Najeeb Najeeb, PhD (najeeb@miu.edu)

V1.1.0

JavaScript Full Stack Development

- MongoDB
  - NoSQL database (document store)
  - Stores JSON documents
- Express
  - JavaScript web framework
  - On top of Node
- Angular
  - JavaScript UI framework
  - Single Page Applications
- Node
  - JavaScript server-side platform
  - Single threaded, fast and scalable

Roadmap and Outcomes

- Node.js: write asynchronous (non-blocking) code. Understand node platform to start a project.
- Express: setup express and get requests and send back responses. REST API.
- MongoDB: what NoSQL DB looks like. Full API interacting with DB.
- AngularJS: Investigate AngularJS and architect it. A single page application.
- MEAN application: Learn by example. We will create a MEAN Games application.

Introduction to AngularJS

AngularJS

jsbin.com

Forgiving expr

Concatenation

Data Binding

Scope

Use AngularJS for the first time. Go to www.jsbin.com
Add library "Angular 1.4.0 Stable"

Add directive

```
<html ng-app>


<body>
1 + 2 = {{ 1+2 }}
</body>
```

AngularJS

jsbin.com

Forgiving expr

Concatenation

Data Binding

Scope

Error will be silently failing.

```
<body>
{{ console.log(something.doesNotExist);  }}
</body>
```

AngularJS

jsbin.com

Forgiving expr

Concatenation

Data Binding

Scope

JS operations can be performed, like string concatenation

```
<body>
{{ "Hello "+ "world!" }}
</body>
```

AngularJS

jsbin.com

Forgiving expr

Concatenation

Data Binding

Scope

Data Binding, use ng-model directive and assign a variable. Two-way data binding, any update to the model updates the view and any update to the view updates the model.

```
<body>
<p> Hello {{ user }}! </p>
<input type="text" name="" id="" ng-model="user" />
</body>
```

AngularJS

jsbin.com

Forgiving expr

Concatentation

Data Binding

Scope

Scope of ng-app depends on where it is defined. The scope of your angular application depends on the scope of ng-app.

```
<html>
...
<body>
<p> Hello {{ user }}! </p>
<div ng-app>
</div>
<input type="text" name="" id="" ng-model="user" />
</body>
</html>
```

Built-in Directives

Directives

- ng-app
- ng-model
- ng-init
- ng-click
- ng-if
- ng-hide
- ng-class
- ng-repeat
- ng-options
- ng-cloak

Directives

ng-init
ng-click
ng-if
ng-show ng-hide
ng-class
ng-repeat
ng-options
ng-cloak

Directive to initialize variables (string, numbers, bool, array or object) do not assign values to variables using ng-init

```
<div ng-init="name= 'Jack'">
   {{ name }}
</div>
```

{{ }} is a shortcut for the directive ng-bind

```
<div ng-init="name= 'Jack'">
   <p ng-bind="name"></p>
</div>
```

Directives

ng-init

ng-click

ng-if

ng-show ng-hide

ng-class

ng-repeat

ng-options

ng-cloak

Directive to execute a method or an expression

```
<div ng-init="number= 0">
   <button ng-click="number= number + 1">+1</button>
   <button ng-click="number= number - 1">-1</button>
   <p> {{ number }} </p>
</div>
```

Directives

ng-init
ng-click
ng-if
ng-show ng-hide
ng-class
ng-repeat
ng-options
ng-cloak

Directive to execute a method or an expression

```
Check the checkbox to see the paragraph
<input type="checkbox" ng-model="showParagraph">
<p ng-if="showParagraph"> The paragraph. </p>
```

We can replace with ng-show and ng-hide

```
<p ng-show="showParagraph"> The paragraph. </p>
```

ng-if removes the element from the DOM tree, ng-show applies CSS display none.

Directives

ng-init
ng-click
ng-if
ng-show ng-hide
ng-class
ng-repeat
ng-options
ng-cloak

```
<div ng-init="number = 19">
   <input type=" text" ng-model="guess">
   <p ng-hide= "guess != number">Correct</p>
   <p ng-show= "guess != number">Incorrect</p>
</div>
```

Directives

ng-init
ng-click
ng-if
ng-show ng-hide
ng-class
ng-repeat
ng-options
ng-cloak

Modify the CSS class dynamically based on conditions

```
<style>
   .red {border-color: red;}
   .green {border-color: green;}
</style>
</head>
<body>
<div ng-init="number = 19">
   <input type=" text" ng-model="guess" ng-class="{red:
guess != number, green: guess == number}">
</div>
```

Directives

ng-init
ng-click
ng-if
ng-show ng-hide
ng-class
ng-repeat
ng-options
ng-cloak

Modify the CSS class dynamically based on conditions

```
<style>
   .red {color: red;}
   .green {color: green;}
</style>
</head>
<body>
<div ng-init="numbers = [0,1,2,3,4,5,6,7,8]">
   <ul>
     <li ng-repeat="number in numbers" ng-class="{red:
$even, green:$odd}">{{ number }}</li>
   </ul>
</div>
```

Directives

ng-init
ng-click
ng-if
ng-show ng-hide
ng-class
ng-repeat
ng-options
ng-cloak

Repeat by values or index

```
<style>
    .red {color: red;}
    .green {color: green;}
</style>
</head>
<body>
<div ng-init="names  = ['Jack', 'John', 'Jack']">
    <ul>
      <li ng-repeat="name in names" ng-class="{red: $even,
green:$odd}">{{ name }}</li>
    </ul>
</div>
```

Repeat using index

```
      <li ng-repeat="name in names track by $index" ng-class="{red:
$even, green:$odd}">{{ name }}</li>
```

Directives

ng-init
ng-click
ng-if
ng-show ng-hide
ng-class
ng-repeat
ng-options
ng-cloak

Repeat over objects

```
<style>
   .red {color: red;}
   .green {color: green;}
</style>
</head>
<body>
<div ng-init="names = [{fistName: 'Jack', lastName: 'Smith'},
{fistName: 'John', lastName: 'Simson'}]">
   <ul>
     <li ng-repeat="name in names" ng-class="{red: $even,
green:$odd}">{{ name.lastName }}, {{ name.firstName}}</li>
   </ul>
</div>
```

Directives

ng-init
ng-click
ng-if
ng-show ng-hide
ng-class
ng-repeat
ng-options
ng-cloak

Do not populate options using ng-repeat, use ng-options

```
</head>
<body>
 <div ng-init="students = [{name: 'Jack', course: 'MPP', gpa: 3.0}, {name: 'John', course: 'MWA', gpa: 2.5}, {name:'Jill', course: 'SWE', gpa: 3.3}, {name: 'Jim', course: 'MWA', gpa: 2.8}]">
   <select name="" id="" ng-model="student" ng-options="student.name for student in students"></select>
    <p>You have selected: {{student:name}} ({{student.gpa}})
</p>
 </div>
```

Grouping

```
<select name="" id="" ng-model="student" ng-options= "student.name group by student.course for student in students"></select>
```

Directives

ng-init
ng-click
ng-if
ng-show ng-hide
ng-class
ng-repeat
ng-options
ng-cloak

Performance and user experience

```
<head>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.0
/angular.min.js"></script>
…
</head>
<body>
…
```

This will result in a delay due to downloading of resource.

displaying some {{…}} while the script is being downloaded.

Directives

ng-init
ng-click
ng-if
ng-show ng-hide
ng-class
ng-repeat
ng-options
ng-cloak

Better performance not so good user expereince

```
...
<body>
...
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.0/angular.min.js"></script>
</body>
```

This will result displaying some {{...}} while the application is loading, then they will be populated.

Directives

ng-init
ng-click
ng-if
ng-show ng-hide
ng-class
ng-repeat
ng-options
ng-cloak

Better performance and user experience use ng-cloak

```
...
 <style>
  .ng-cloak, [ng-cloak], [ng\:cloak] {
    display: none !important;
  }
 </style>
</head>
<body>
 <div ng-cloak
...
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/
1.4.0/angular.min.js"></script>
</body>
```

Until AngularJS has not finished the bootstrapping process it will not display anything, after it is done it will display.

Built-in Filters

Some Built-in Filters

- Currency
- Number
- String
- Date
- Limit
- Order
- Filter

Filters

Currency
Number
Strings
Date
Limit
OrderBy
Filter

Display money values using currency filter.

```
<div ng-init="total= 123.45">
  <p>{{ total | currency}}</p>
<div>
```

Different currency?

```
  <p>{{ total | currency:"£"}}</p>
  <p>{{ total | currency:"¥"}}</p>
  <p>{{ total | currency:"€"}}</p>
```

Filters

Currency

Number

Strings

Date

Limit

OrderBy

Filter

Display a format for decimal digits to display.

```
<div ng-init="interest= 123.456789">
  <p>{{ interest | number: 4}}</p>
<div>
```

Negative numbers?

```
  <p>{{ -interest | number:4}}</p>
```

Filters

Currency

Number

Strings

Date

Limit

OrderBy

Filter

Display a format for decimal digits to display.

```
<div ng-init="title= 'Maharishi International University'">
  <p>{{ title | uppercase}}</p>
<div>
```

Lower case?

```
  <p>{{ title | lowercase}}</p>
```

Filters

Currency

Number

Strings

Date

Limit

OrderBy

Filter

Display a format for decimal digits to display.

```
<div ng-init="firstDayOfCourse= 1626706800000">
  <p>{{ firstDayOfCourse | date}}</p>
<div>
```

Date and Time?

```
  <p>{{ firstDayOfCourse | date: "short"}}</p>
  <p>{{ firstDayOfCourse | date: "medium"}}</p>
```

Fine ontrol?

```
  <p>{{ firstDayOfCourse | date: "MMM-dd-yyyy
(hh:mm:ss:(sss)"}}</p>
  <p>{{ firstDayOfCourse | date: "MM-dd-
yy (hh:mm:ss:(sss)"}}</p>
```

Filters

Currency

Number

Strings

Date

Limit

OrderBy

Filter

Limit items returned from Array or Object.

```
<div ng-init="numbers= [0,1,2,3,4,5]">
  <p>{{ numbers | limitTo: 4}}</p>
<div>
```

Get last numbers instead of first?

```
  <p>{{ numbers | limitTo: -2}}</p>
```

Filters

Currency

Number

Strings

Date

Limit

OrderBy

Filter

Limit items returned from Array or Object.

```
<div ng-init="students = [{name: 'Jack', course: 'MPP', gpa: 3.0}, {name: 'John', course: 'MWA', gpa: 2.5}, {name:'Jill', course: 'SWE', gpa: 3.3}, {name: 'Jim', course: 'MWA', gpa: 2.8}]">
  <ul>
    <li ng-repeat="student in students | orderBy: 'gpa'">{{student.name}} with gpa {{student.gpa}} taking {{student.course}}.</li>
  </ul>
<div>
```

Reverse order?

```
    <li ng-repeat="student in students | orderBy: '-gpa'">
```

Nested ordering?

```
    <li ng-repeat="student in students | orderBy: ['course', '-gpa']">
```

Filters

Currency

Number

Strings

Date

Limit

OrderBy

Filter

Search through the data set.

```
<div ng-init="students  = [{name:  'Jack', course:  'MPP', gpa:  3.0},
{name:  'John', course:  'MWA', gpa:  2.5}, {name:'Jill', course:  'SWE',
gpa:  3.3}, {name:  'Jim', course:  'MWA', gpa:  2.8}]">
  <ul>
    <li ng-repeat="student  in students | filter:  'jack'">{{student.name}}
with gpa {{student.gpa}} taking {{student.course}}.</li>
  </ul>
<div>
```

Implement a dynamic search instead of a static one?

```
    <input type="text"  ng-model="searchText">
    <li ng-repeat="student  in students | filter:  searchText>
```

Only search courses?

```
    <input type="text"  ng-model="searchText.course">
```

Search all fields?

```
    <input type="text"  ng-model="searchText.$">
```

Controllers

Controllers

Controller

As Syntax

My First Controller, create an Angular module (myFirstApp)

```
<html ng-app="myFirstApp">
...
<script>
  angular.module("myFirstApp",
[]).controller("MyFirstController", MyFistController);
  function MyFirstController($scope) {
    $scope.name=  "Jack";
  }
</script>
...
<body>
  <div ng-controller="MyFirstController">
    Hello, {{name}}!
  </div>
</body>
```

Controllers

Controller

As Syntax

Use Arrays in controller.

```
<html ng-app="myFirstApp">
…
<script>
 angular.module("myFirstApp", []).controller("MyFirstController",
MyFistController);
 function MyFirstController($scope) {
  $scope.students= [{name: 'Jack', course: 'MPP', gpa: 3.0}, {name:
'John', course: 'MWA', gpa: 2.5}, {name:'Jill', course: 'SWE', gpa: 3.3},
{name: 'Jim', course: 'MWA', gpa: 2.8}];
 }
</script>
…
<body>
 <div ng-controller="MyFirstController">
  <ul>
   <li ng-repeat="student in students">{{student.name}} with gpa
{{student.gpa}} taking {{student.course}}.</li>
  </ul>
 </div>
</body>
```

Controllers

Controller

As Syntax

Use functions in $ scope.

```
<html ng-app="myFirstApp">
…
<script>
 angular.module("myFirstApp", []).controller("MyFirstController", MyFistController);
 function MyFirstController($scope) {
  $scope.number= 0;
  $scope.increment= function(value) {
   $scope.number= $scope.number+value;
  }
  $scope.decrement= function(value) {
   $scope.number= $scope.number-value;
  }
 }
</script>
…
<body>
 <div ng-controller="MyFirstController">
  {{number}}
 <p>
  <button ng-click="increment(5)">+5</button>
  <button ng-click="decrement(5)">-5</button>
 </p>
 </div>
</body>
```

Controllers

Controller As Syntax

More than one controller, using controller as syntax.

```
<html ng-app="myFirstApp">
…
<script>
  angular.module("myFirstApp", []).controller("MyFirstController", MyFistController)
.controller("MySecondController", MySecondController);
  function MyFirstController(){
    const vm= this;
    this.name= "Jack";
  }
  function MySecondController($scope){
    const vm= this;
    this.name= "John";
  }
</script>
…
<body>
  <div ng-controller="MyFirstController as MyJackCtrl">
    <div ng-controller="MySecondController as MyJohnCtrl">
       Hello {{MyJackCtrl.name}}!
       Hello {{MyJohnCtrl.name}}!
    </div>
  </div>
</body>
```

Modules

Modules

HTML Page

Module

Controller

Create file index.html

```
<!DOCTYPE html>
<html ng-app="myProperApp">
<head>
<meta charset="utf-8">
<meta name="viewport"  content="width=device-width">
<title></title>
</head>
<body>
  <div ng-controller="MyProperController  as MyCtrl">
    <p>Hello, {{MyCtrl.name}}!</p>
  </div>
<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.0/angular.
min.js"></script>
<script src="app.js"></script>
<script src="controller.js"></script>
</body>
</html>
```

Modules
HTML Page
Module
Controller

Create file app.js

```
angular.module("myProperApp", []);
```

Modules
HTML Page
Module
Controller

Create file controller.js

```
angular.module("myProperApp").controller("MyProperCo
ntroller", MyProperController);
function MyProperController() {
  const vm= this;
  vm.name= "Jack";
}
```

Single Page Application (SPA)

SPA

template

templateURL

controller

Bad URL

Modify file index.html

```
...
<body>
  <div ng-view>
  </div>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.0/angular.min.js"
></script>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.0/angular-
route.min.js"></script>
...
</body>
```

Add dependency, modify file app.js

```
angular.module("myTemplateApp", ['ngRoute"]).config(config);
function config($routeProvider){
  $routeProvider.when("/", {
    template: "<h1>This is the home page.</h1>"
  }).when("/about", {
    template: "<h1>This is the about page.</h1>"
  });
}
```

SPA

template

templateURL

controller

Bad URL

Add dependency, modify file app.js

```
angular.module("myTemplateApp",
['ngRoute"]).config(config);
function config($routeProvider) {
  $routeProvider.when("/", {
    templateUrl: "template/main.html"
  }).when("/about", {
    templateUrl: "template/about.html"
  });
}
```

Create template/main.html

```
<h1>This is the home page.</h1>
```

Create template/about.html

```
<h1>This is the about page.</h1>
```

Setup Server to Serve

- Due to security the previous page redirection will not work.
- How to setup a webserver:
- Using Python
  - python -m SimpleHTTPServer 8181
  - python –m http.server 8181
  - python3 –m http.server 8181

SPA

template

templateURL

controller

Bad URL

Add controller, modify file app.js

```
angular.module.("myControllerApp", ['ngRoute']).config(config);
function config($routeProvider) {
  $routeProvider.when("/", {
    templateUrl: "template/main.html",
    controller: "MainController",
    controllerAs: "mainCtrl"
  }).when("/about", {
    templateUrl: "template/about.html",
    controller: "AboutController",
    controllerAs: "aboutCtrl"
  });
}
```

Create controller file mainController.js

```
angular.module("myControllerApp").controller("MainController",  MainController);
function MainController() {
  const vm= this;
  vm.name= "Jack";
}
```

Create controller file aboutController.js

```
angular.module("myControllerApp").controller("AboutController",  AboutController);
function MainController() {
  const vm= this;
  vm.about= "This is my bio";
}
```

SPA

template

templateURL

controller

Bad URL

Add controllers to your HTML file, modify index.html

```
…
<script src="mainController.js></script>
<script src="aboutController.js></script>
…
```

Update main.html, no need to use controller directive

```
<H1> Hello, {{ mainCtrl.name }} </H1>
```

Update about.html

```
<p> {{ aboutCtrl.bio }} </p>
```

SPA

template

templateURL

controller

Bad URL

If a request for a non-exsisting page is made we can handle that using otherwise. Modify app.js

```
…
}).when(…
}).otherwise({
  redirectTo: "/"
});
…
```

We can also handle error with a "Page not Found" 404
In this case we need to add a controller and possibly a template for that.

Services

Services

http

routeParams

Factory

Add service to MainController, modify file mainController.js

```
function MainController($http) {
  const vm= this;
  $http.get("https://official-joke-
api.appspot.com/jokes/ten")  .then(function(response) {
    vm.jokes= response.data;
  });
}
```

Modify template/main.html

```
<ul>
  <li ng-repeat="joke in
mainCtrl.jokes">{{joke.type}}<BR/>
  {{joke.setup}} : {{joke:punchline}}</li>
</ul>
```

Services

http

routeParams

Factory

Read route parameters, modify file app.js

```
…
}).when("joke/:jokeType",{
templateUrl: "template/joke.html",
controller: "JokeController",
controllerAs: "jokeCtrl"});
}
```

Add jokeController.js

```
angular.module("myControllerApp").controller("JokeController",JokeController);
function JokeController($http,$routeParams){
  const vm= this;
  const jokeType= $routeParams.jokeType;
  $http.get("https://official-joke-
api.appspot.com/jokes/"+jokeType+"/random").then(function(response){
    vm.joke= response.data;
  });
}
```

Add template/joke.html

```
{{jokeCntrl.joke.type}}<BR/>
{{jokeCntrl.joke.setup}}: {jokeCntrl.{joke:punchline}}
```

Update index.html

```
<script scr="jokeController.js"></script>
```

Architecture

- Routes
  - app.js
- Create a folder for each part of the application (main, joke, about, …) folder contents
  - Templates file (main.html, joke.html, …)
  - Controller file (main-controller.js, joke-controller.js, …)

Service

http

routeParams

Factory

Create dataFactory/dataFactory.js

```
angular.module("myJokeApp").factory("JokeFactory",   JokeFactory);
function JokeFactory($http) {
  return {
    getTenJokes: getTenJokes,
    getOneJoke: getOneJoke
  };
  function getTenJokes(){
    return $http.get("https://official-joke-api.appspot.com/jokes/ten").then(complete).catch(failed);
  }
  function getOneJoke(jokeType){
    return $http.get("https://official-joke-
api.appspot.com/jokes/"+jokeType+"/random").then(complete).catch(failed);
  }
  function complete(response){
    return response.data;
  }
  function failed(error) {
    return error.statusText;
  }

}
```

Update index.html to read the factory script

```
<script src="dataFactory/dataFactory.js"></script>
```

Service
http
routeParams
Factory

Update controllers to use the Factor, update main-controll.js

```
function MainController(JokeFactory) {
  const vm= this;
  JokeFactory.getTenJokes().then(function(response) {
    vm.jokes= response;
  });
}
```

Update joke-controller.js

```
function JokeController($routeParams, JokeFactory) {
  const vm= this;
  const jokeType= $routeParams.jokeType;
  JokeFactory.getOneJoke(jokeType).then(function(response)
{
    vm.joke= response[0];
  });
}
```

Custom Filters

Filter

Number Filter

String Filter

Add the filter filters/numberPostfix.js

```
angular.module("myJokeApp").filter("order", numberOrder);
function numberOrder() {
  return function(number) {
    if(number && !isNaN(number)) {
      const digit= number%10;
      let suffix= "";
      switch(digit) {
        case 1:
          suffix= "st";
          break;
        case 2:
          suffix= "nd";
          break;
        case 3:
          suffix= "rd";
          break;
        default:
          suffix= "th";
          break;
        }
      return number+suffix;
    }
    return number;
  }
}
```

Update index.html

```
<script src="filters/numberPostfix.js></script>
```

Update main/main.html

```
{{joke.id | order}}
```

Filter

Number Filter

String Filter

Add the filter filters/vowelsRemover.js

```
angular.module("myJokeApp").filter("vowels", vowelRemover);
function vowelRemover() {
  return function(string, vowel) {
    if(string && (vowel=="a" || vowel=="e" || vowel=="i" || vowel=="o" || vowel=="u") {
      let newString= "";
      for(let char of string) {
        c= char.toLocaleLowerCase();
        if (c == vowel) {
          continue;
        }
        newString+= char;
      }
      return newString;
    }
    return string;
  }
}
```

Update index.html

```
<script src="filters/vowelsRemover.js></script>
```

Update main/main.html

```
{{joke.type | vowels:"e"}}
```

Main Points

- AngularJS is the UI part of a MEAN application. It enables building flexible Single Page Applications (SPA).
- AngularJS enforces an MVC architecture. AngularJS enforces proper software engineering practices, separation of concern.
- AngularJS has a set of built-in directive to speed up the development of web applications. At the same time, you may write your own custom directives and filters.