ANGULAR JS

- AngularJS is a **JavaScript framework**. It is a library written in JavaScript.
- It helps in developing SPA – **Single page applications**.
- (Request) All calls are **AJAX** --> no page reloading
- **JavaScript and JQuery** with Ajax is used to single page application. Without refreshing I can use this page.
- Building web applications with Angular JS - server work load very low and performance very high and response time is very quick processes.

Is AngularJS dependent on JQuery? No.

- It follows **MVC (Model View Controller)** pattern. It is open source, cross browser compliant and easy to maintain.
- It can be added to an HTML page with a <script> tag.
  <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></script>

Advantages of AngularJS?

- Allows us to create single page application
- follows MVC pattern
- predefined form validations
- supports animation
- open source
- supports two way data binding
- its code are unit testable

Disadvantages of AngularJS?

- JavaScript Dependent: If end user disables JavaScript, AngularJS will not work.
- Not Secured: It is JavaScript based framework so it is not safe to authenticate user through AngularJS only.

IDE's are currently used for the development of AngularJS?

Eclipse → JetBrains WebStorm

Features of AngularJS?

MVC Validations Modules Directives Templates Scope

Expressions Data Binding Filters Services Routing

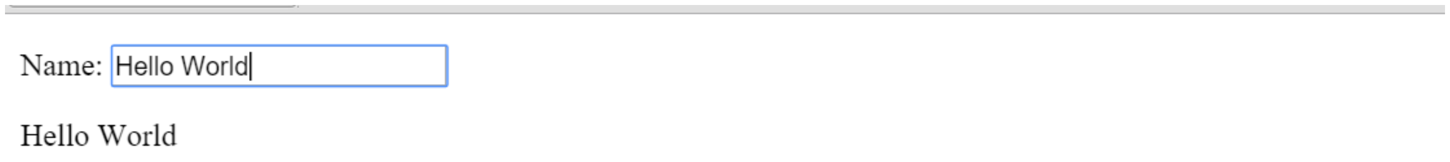
Dependency Injection Testing

- Angular JS **version-1 in 2011**. **Google** gave support to developers of Angular JS.
- **Innerhtml in html** --> changes from tommy to sarah in the web page --> document.getElementID("");

- AngularJS extends HTML attributes with **Directives**, and binds data to HTML with **Expressions**.

First Program:

```
<div ng-app="">
   <p>Name: <input type="text" ng-model="name"></p>
 {{name}}
</div>
```



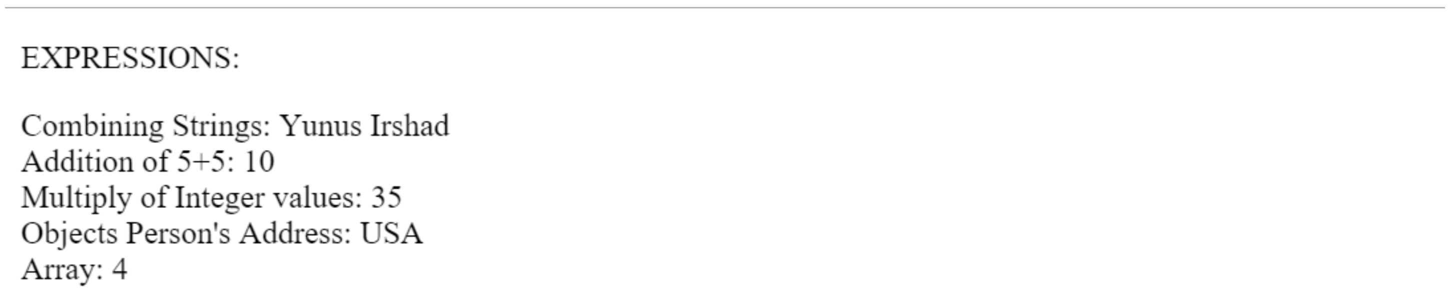
```
Name: Hello World

Hello World
```

AngularJS Expressions

AngularJS expressions are written inside double braces: **{{ expression }}**.

AngularJS expressions binds data to HTML the same way as the **ng-bind** directive.

```
<div ng-app="" ng-init="a=5;b=7;Fname='Yunus';Lname='Irshad';Person={name:'New
Name',Address:'USA'};points=[1,2,3,4,5]">
   <p>EXPRESSIONS: </p>
   Combining Strings: {{ Fname + " " + Lname }}<br>
   Addition of 5+5: {{5+5}}<br>
   Multiply of Integer values: {{ a*b }}<br>
   Objects Person's Address: {{Person.Address}}<br>
   Array: {{points[3]}}
</div>

<div ng-app="" ng-init="a=5;b=7;Fname='Yunus';Lname='Irshad';Person={name:'New
Name',Address:'USA'}">
   <p>EXPRESSIONS: </p>
   Objects Person's Address: <span ng-bind="Person.name" /><br>
   Addition of Integer values using ng-bind: <span ng-bind="a+b" /><br>
   Combining Strings using ng-bind: <span ng-bind="Fname+' '+Lname"/><br>
   Array: <span ng-bind="points[3]" />
</div>
```



```
EXPRESSIONS:

Combining Strings: Yunus Irshad
Addition of 5+5: 10
Multiply of Integer values: 35
Objects Person's Address: USA
Array: 4
```

AngularJS vs JavaScript

- Like JavaScript expressions, AngularJS expressions can contain literals, operators, and variables.
- Unlike JavaScript expressions, AngularJS expressions can be written inside HTML.
- AngularJS expressions do not support conditionals, loops, and exceptions, while JavaScript expressions do.
- AngularJS expressions support filters, while JavaScript expressions do not.

AngularJS Directives

AngularJS directives are extended HTML attributes with the prefix **ng-**.

The **ng-app** directive initializes an AngularJS application. The **ng-app** directive will **auto-bootstrap** (automatically initialize) the application when a web page is loaded.

The **ng-init** directive initializes application data. Normally, you will not use ng-init. You will use a controller or module instead.

The **ng-model** directive binds the value of HTML controls (input, select, textarea) to application data. Provide type validation and status for application data

Data Binding

Data binding in AngularJS synchronizes AngularJS expressions with AngularJS data.

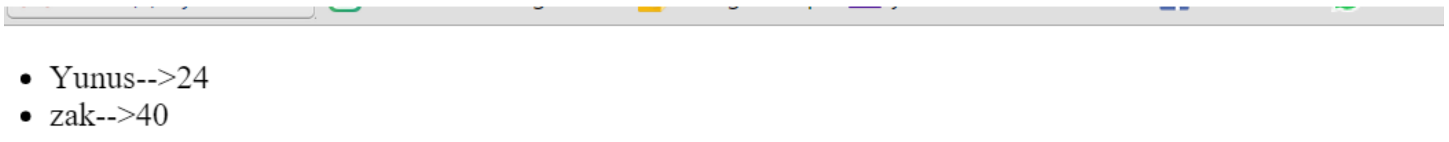
{{ firstName }} is synchronized with **ng-model="firstName"**.

Repeating HTML Elements

The **ng-repeat** directive repeats an HTML element: **ng-repeat** directive **clones HTML elements** once for each item in a collection (in an array).

```
<div ng-app="" ng-init="pointsarray=['Yunus','Zak','Irshad']">
<p>
  <ui>
    <li ng-repeat="points in pointsarray">            // for (int x: array)
      {{points}}
    </li>
  </ui>
</p>
</div>
<div ng-app="" ng-init="pointsarray=[{name:'Yunus',age:'24'},{name:'zak',age:'40'}]">
  <p>                                       // array objects
    <ui>
      <li ng-repeat="points in pointsarray">
        {{points.name+'-->'+points.age}}
      </li>
    </ui>
  </p>
</div>
```

AngularJS is perfect for database CRUD (Create Read Update Delete) applications.
Just imagine if these objects were records from a database.



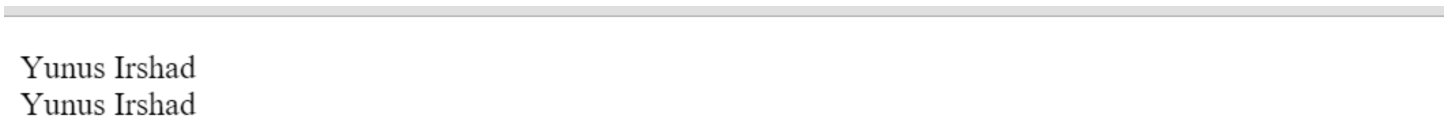
- Yunus-->24
- zak-->40

AngularJS Controllers

AngularJS controllers **control the data** of AngularJS applications.

The **ng-controller** directive defines the application controller.

```
<div ng-app="myApp" ng-controller="myCtrl">
   <p>{{Fname + " " + Lname}}<br>
   {{fullname()}}              // method controller
   </p>
</div>
<script src="ExternalController.js"></script>
<script>
   var app = angular.module('myApp', []);
   app.controller('myCtrl',function($scope)
   {
      $scope.Fname="Yunus";
      $scope.Lname="Irshad";
      $scope.fullname = function()
      {
         return $scope.Fname+ " "+$scope.Lname;
      }
   });
</script>
```



```
Yunus Irshad
Yunus Irshad
```

In AngularJS, $scope is the application object (the owner of application variables and functions).

Controller Methods

A controller can also have methods (variables as functions):

Controllers In External Files

In larger applications, it is common to store controllers in external files.

```
var app = angular.module('myApp', []);
app.controller('myCtrl',function($scope)
{
$scope.names = [{name:'Yunus', age:'24'},{name:'Irshad', age:'28'}];
});
```

- Yunus 24
- Irshad 28

AngularJS Filters

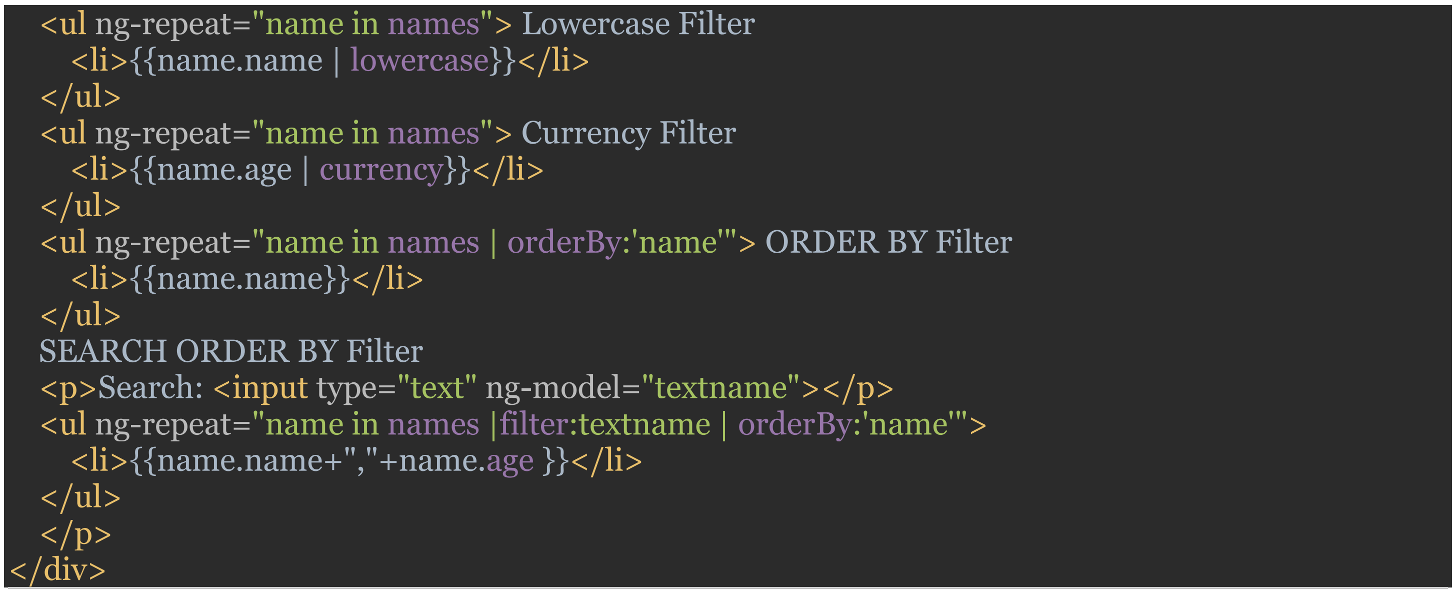
AngularJS filters can be used to transform data:

Filter	**Description**
currency	Format a number to a currency format.
filter	Select a subset of items from an array. A filter can be added to a directive with a pipe character (|) and a filter.
lowercase	Format a string to lower case.
orderBy	Orders an array by an expression.
uppercase	Format a string to upper case.

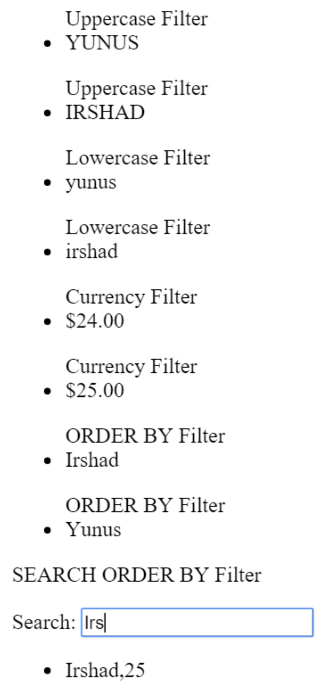
Filtering Input

An input filter can be added to a directive with a pipe character (|) and filter followed by a colon and a model name.

```
<div ng-app="myApp" ng-controller="myCtrl">
  <p>
    <ul ng-repeat="name in names"> Uppercase Filter
      <li>{{name.name | uppercase}}</li>
    </ul>
```



```
  <ul ng-repeat="name in names"> Lowercase Filter
    <li>{{name.name | lowercase}}</li>
  </ul>
  <ul ng-repeat="name in names"> Currency Filter
    <li>{{name.age | currency}}</li>
  </ul>
  <ul ng-repeat="name in names | orderBy:'name'"> ORDER BY Filter
    <li>{{name.name}}</li>
  </ul>
  SEARCH ORDER BY Filter
  <p>Search: <input type="text" ng-model="textname"></p>
  <ul ng-repeat="name in names |filter:textname | orderBy:'name'">
    <li>{{name.name+","+name.age }}</li>
  </ul>
  </p>
</div>
```



Uppercase Filter
- YUNUS

Uppercase Filter
- IRSHAD

Lowercase Filter
- yunus

Lowercase Filter
- irshad

Currency Filter
- $24.00

Currency Filter
- $25.00

ORDER BY Filter
- Irshad

ORDER BY Filter
- Yunus

SEARCH ORDER BY Filter

Search: Irs

- Irshad,25

AngularJS AJAX - $http

$http is an AngularJS service for reading data from web remote servers.

$http.get(url) is the function to use for reading server data.

http://www.w3schools.com/angular/customers.php

```
{
"records": [
 {
  "Name" : "Alfreds Futterkiste",
  "City" : "Berlin",
  "Country" : "Germany"
```

```
},
{
```

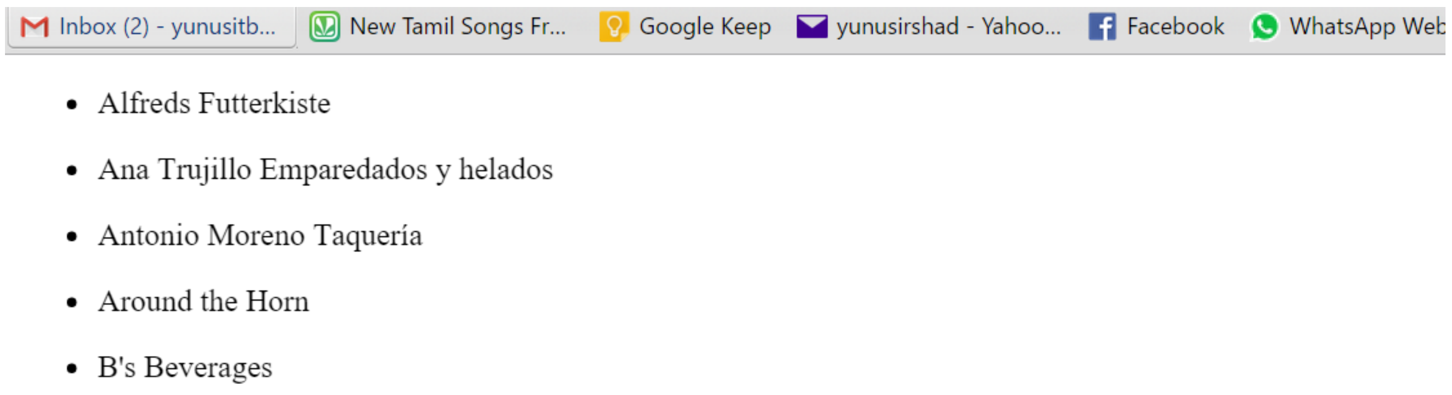
$http is an **XMLHttpRequest object** for requesting external data.

$http.get() reads **JSON data** from http://www.w3schools.com/angular/customers.php.

If **success**, the controller creates a property (**names**) in the scope, with JSON data from the server.

```
var app = angular.module('myApp', []);
app.controller('myCtrl',function($scope, $http) {
   $http.get("http://www.w3schools.com/angular/customers.php").success(function (response) {

      $scope.names = response.records;
   });

});
```

```
<div ng-app="myApp" ng-controller="myCtrl">
   <p>
      <ul ng-repeat="name in names">
         <li>{{name.Name}}</li>
      </ul>
   </p>
</div>
```



- Alfreds Futterkiste
- Ana Trujillo Emparedados y helados
- Antonio Moreno Taquería
- Around the Horn
- B's Beverages

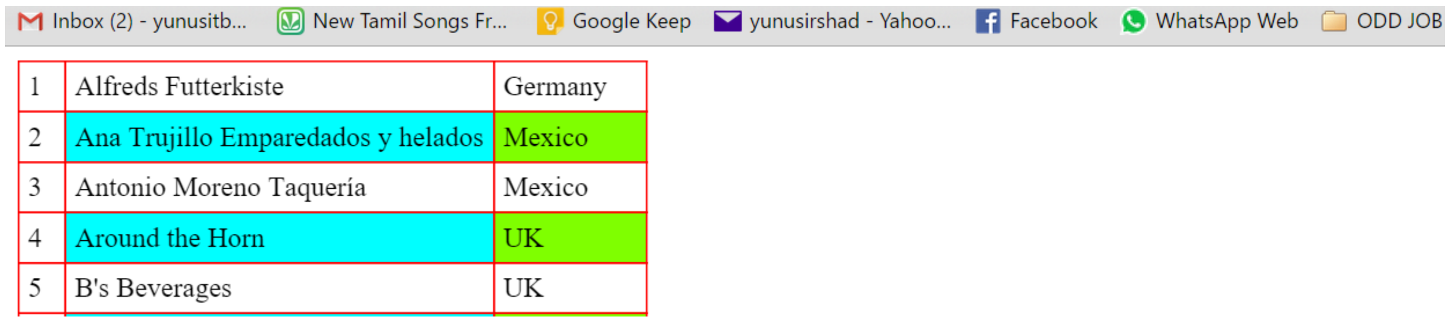
Displaying Data in a Table

- **Using $even and $odd in Table**
- **Display the Table Index ($index)**

To display the table index, add a <td> with **$index**:

```
<style>
   table, th , td {
      border: 1px solid red;
      border-collapse: collapse;
      padding: 5px;
```

```
  }
</style>
<body>
<div ng-app="myApp" ng-controller="myCtrl">

  <table>
    <tr ng-repeat="name in names">
      <td>{{$index+1}}</td>  <!--Displaying Table Index -->
      <td ng-if="$odd" style="background-color: aqua">{{name.Name}}</td>
      <td ng-if="$even">{{name.Name}}</td>
      <td ng-if="$odd" style="background-color: chartreuse">{{name.Country}}</td>
      <td ng-if="$even">{{name.Country}}</td>
    </tr>
  </table>
</div>
```



1	Alfreds Futterkiste	Germany
2	Ana Trujillo Emparedados y helados	Mexico
3	Antonio Moreno Taquería	Mexico
4	Around the Horn	UK
5	B's Beverages	UK

AngularJS SQL

- Fetching Data From a Tomcat Server Running Java restful – JSON – MySQL server.

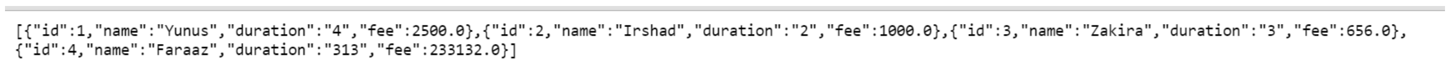
```
public String courses()
        {
                String courses = null;
                ArrayList<Course> courseList = new ArrayList<Course>();
                try
                {
                        courseList = new AccessManager().getCourses();

                        Gson gson = new Gson();

                        courses = gson.toJson(courseList);
```

- Export it as JSON File

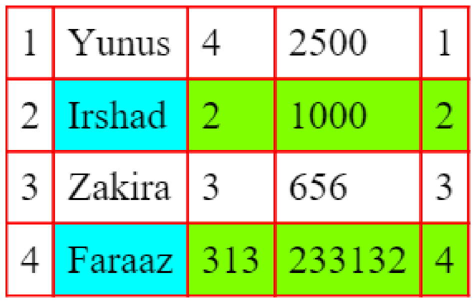
```
FileWriter file = new FileWriter("E:\\IdeaProjects\\Angular\\courses.json");
                                file.write(gson.toJson(courseList));
                                file.flush();

                                file.close();
```

- Angular program reads the JSON file.

```
var app = angular.module('myApp', []);
app.controller('myCtrl',function($scope, $http) {
  $http.get('courses.json').success(function (response) {
$scope.courses = response;
});
});
```



[{"id":1,"name":"Yunus","duration":"4","fee":2500.0},{"id":2,"name":"Irshad","duration":"2","fee":1000.0},{"id":3,"name":"Zakira","duration":"3","fee":656.0},
{"id":4,"name":"Faraaz","duration":"313","fee":233132.0}]

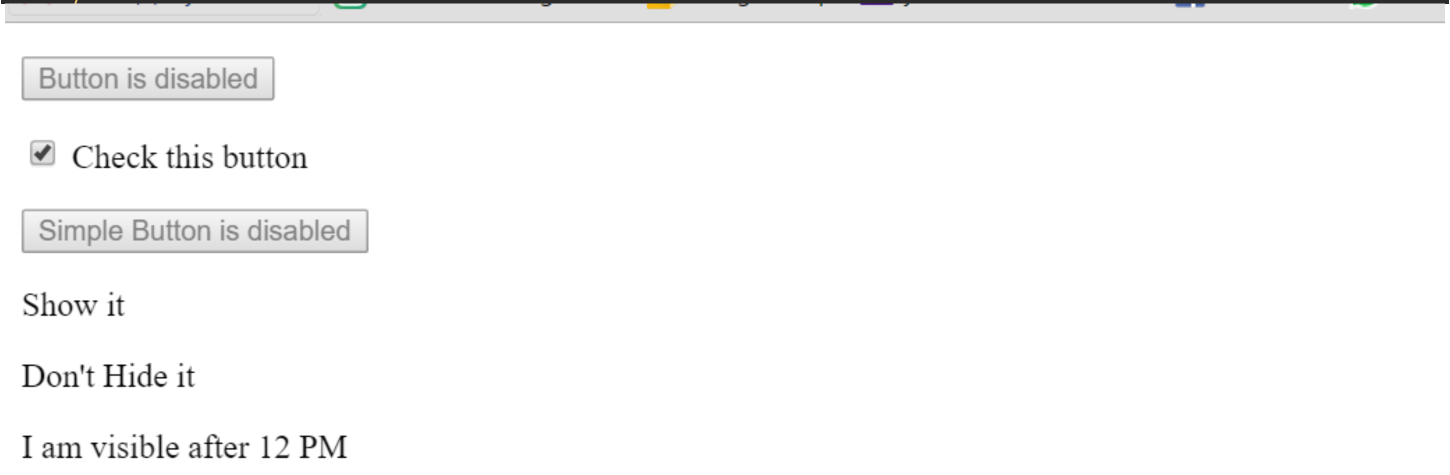


1	Yunus	4	2500	1
2	Irshad	2	1000	2
3	Zakira	3	656	3
4	Faraaz	313	233132	4

AngularJS HTML DOM

- The **ng-disabled** directive binds AngularJS application data to the disabled attribute of HTML elements.
- If the value of **mySwitch** evaluates to **true**, the button will be disabled:
- The **ng-show** directive shows or hides an HTML element.
- The ng-show directive shows (or hides) an HTML element based on the **value** of ng-show.
- The **ng-hide** directive hides or shows an HTML element.

```
<div ng-app="" ng-init="buttonName=true;hour=13">
   <p><button ng-disabled="buttonName"> Button is disabled</button></p>
   <p><input type="checkbox" ng-model="buttonName"> Check this button </p>
   <p><button disabled> Simple Button is disabled</button></p>
   <p ng-show="true"> Show it</p>
   <p ng-show="false"> Don't Show it</p>
   <p ng-hide="true"> Hide it</p>
   <p ng-hide="false"> Don't Hide it</p>
   <p ng-show="hour > 12"> I am visible after 12 PM</p>
</div>
```



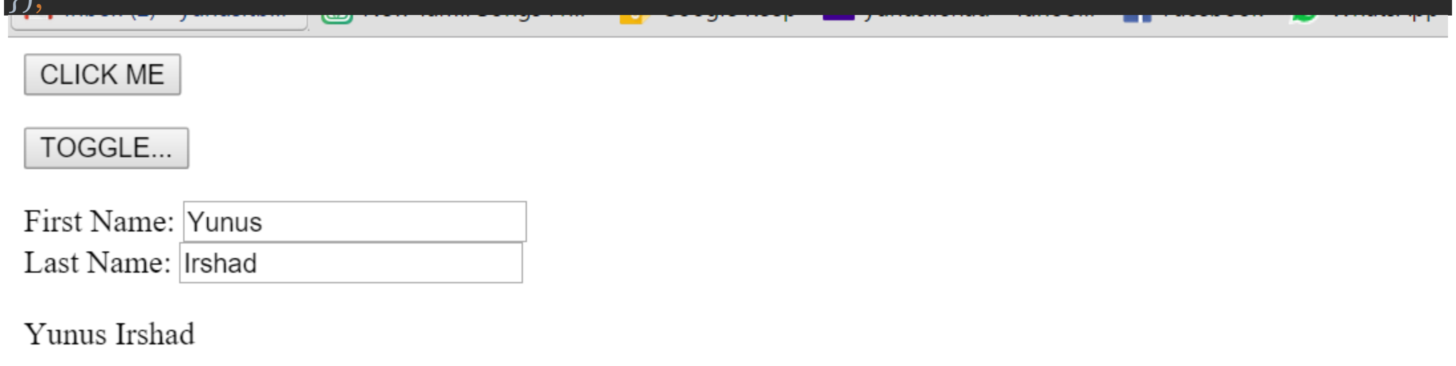
```
Show it

Don't Hide it

I am visible after 12 PM
```

AngularJS Events

- The **ng-click** directive defines an AngularJS click event.
- The **ng-show or ng-hide** directive can also be used to set the **visibility** of a part of an application.

```
• <div ng-app="myApp" ng-controller="myCtrl">
    <button ng-click="count = count+1">CLICK ME</button>
     <p>{{count}}</p>

     <button ng-click="toggle()">TOGGLE...</button>
     <p ng-show="myVar">First Name: <input type="text" ng-model="Fname"><br>
     Last Name: <input type="text" ng-model="Lname"></p>
     <p>{{Fname +  " "+Lname}}</p>
  </div>
```

```
var app = angular.module('myApp', []);
app.controller('myCtrl',function($scope) {
      /*$scope.count = 0;*/
      $scope.Fname="Yunus";
      $scope.Lname="Irshad";
      $scope.myVar=true;
   $scope.toggle = function()
   {
      $scope.myVar=!$scope.myVar;
   };
});
```

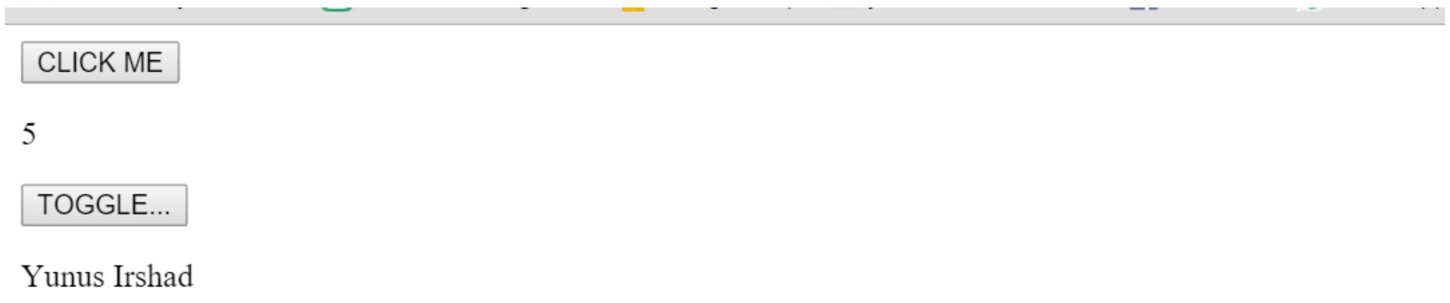

CLICK ME

TOGGLE...

First Name: Yunus
Last Name: Irshad

Yunus Irshad

AFTER


CLICK ME

5

TOGGLE...

Yunus Irshad

AngularJS Modules

- An AngularJS module defines an application.
- The module is a container for the different parts of an application.
- The module is a container for the application controllers.
- Controllers always belong to a module.

```
<div ng-app="myApp" ng-controller="myCtrl">
   <p>{{Fname +  " "+Lname}}</p>
</div>
<script src="myApp.js"></script>
<script src="ExternalController.js"></script>
```

Modules and Controllers in Files

- It is common in AngularJS applications to put the module and the controllers in JavaScript files.
- In this example, "myApp.js" contains an application module definition, while "myCtrl.js" contains the controller:

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

```
app.controller('myCtrl',function($scope) {
      $scope.Fname="Yunus";
      $scope.Lname="Irshad";
});
```

Functions can Pollute the Global Namespace

- Global functions should be avoided in JavaScript. They can easily be overwritten or destroyed by other scripts.
- AngularJS modules reduces this problem, by keeping all functions local to the module.

AngularJS Forms

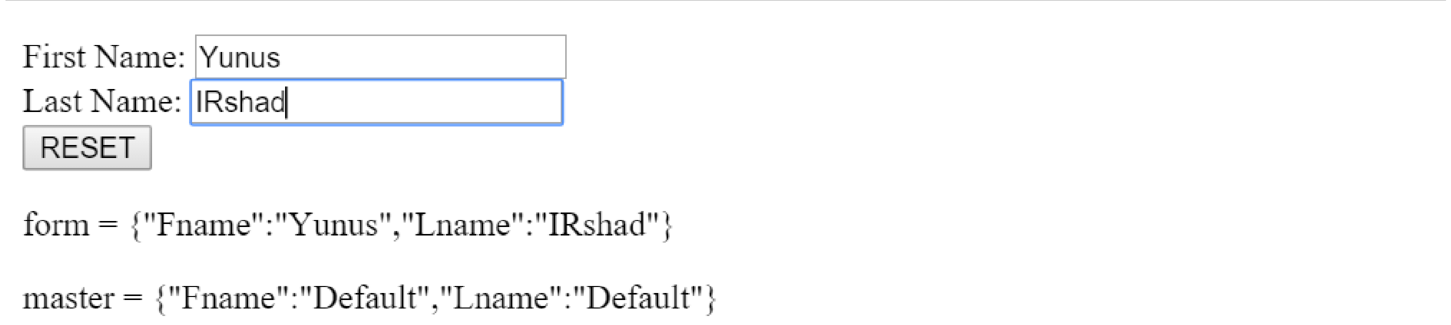
An AngularJS form is a collection of input controls.

HTML input elements are called HTML controls:

- input elements
- select elements
- button elements
- textarea elements

```
<div ng-app="myApp" ng-controller="myCtrl">
   <form novalidate>
      <p>First Name: <input type="text" ng-model="user.Fname"><br>
         Last Name: <input type="text" ng-model="user.Lname"><br>
      <button ng-click="reset()">RESET</button></p>
   </form>
```

```
    <p> form = {{user}}</p>
    <p> master = {{master}}</p>
  </div>
```



First Name: Yunus
Last Name: IRshad
RESET

form = {"Fname":"Yunus","Lname":"IRshad"}

master = {"Fname":"Default","Lname":"Default"}

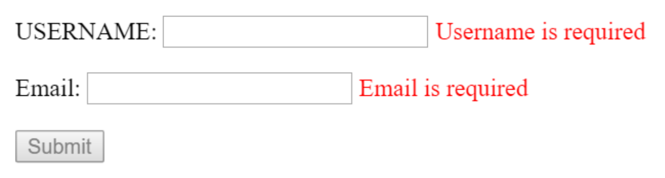
- The **novalidate** attribute is new in HTML5. It disables any default browser validation
- The **formCtrl** function sets initial values to the **master** object, and defines the **reset()** method.
- The **reset()** method sets the **user** object equal to the **master** object.

AngularJS Input Validation

- AngularJS forms and controls can provide validation services, and notify users of invalid input.
- Client-side validation cannot alone secure user input. Server side validation is also necessary.

```
  <form ng-app="myApp" ng-controller="myCtrl" name="myForm" novalidate>
    <p>USERNAME: <input type="text" name="Uname" ng-model="Uname" required>
    <span style="color:red" ng-show="myForm.Uname.$dirty && myForm.Uname.$invalid">
      <span ng-show="myForm.Uname.$error.required">Username is required</span>
    </span>
    </p>
    <p>Email: <input type="email" name="Email" ng-model="Email" required>
    <span style="color:red" ng-show="myForm.Email.$dirty && myForm.Email.$invalid">
      <span ng-show="myForm.Email.$error.required">Email is required</span>
      <span ng-show="myForm.Email.$error.email">Invalid Email address</span>
    </span>
    </p>
    <p>
      <input type="submit" ng-disabled="myForm.Uname.$dirty && myForm.Uname.$invalid ||
      myForm.Email.$dirty && myForm.email.$invalid">
    </p>
  </form>
```

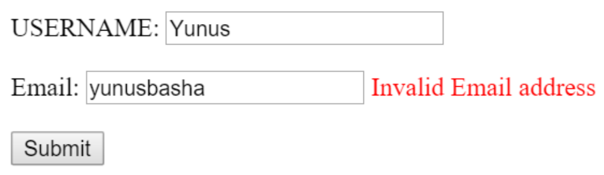
```
var app = angular.module('myApp', []);
app.controller('myCtrl',function($scope) {
   $scope.Uname = "Yunus";
   $scope.Email = "yunusitboss@gmail.com";
});
```



USERNAME: Username is required

Email: Email is required

Submit


USERNAME: Yunus

Email: yunusbasha Invalid Email address

- The model object has two properties: **user** and **email**.
- Because of **ng-show**, the spans with color:red are displayed only when user or email is **$dirty** and **$invalid**.

Property	**Description**
$dirty	The user has interacted with the field.
$valid	The field content is valid.
$invalid	The field content is invalid.
$pristine	User has not interacted with the field yet.

AngularJS API

- API stands for **A**pplication **P**rogramming **I**nterface

The AngularJS Global API is a set of global JavaScript functions for performing common tasks like:

- Comparing objects
- Iterating objects
- Converting data

The Global API functions are accessed using the angular object.

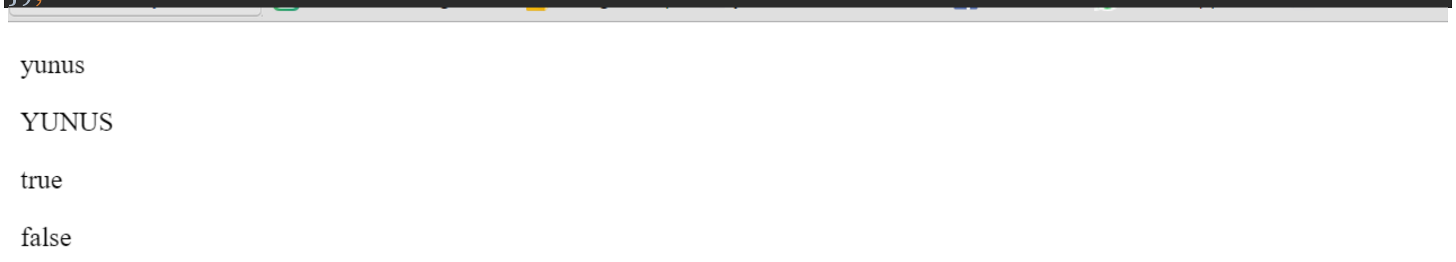
Below is a list of some common API functions:

API	**Description**

angular.lowercase()	Converts a string to lowercase
angular.uppercase()	Converts a string to uppercase
angular.isString()	Returns true if the reference is a string
angular.isNumber()	Returns true if the reference is a number

```
<div ng-app="myApp" ng-controller="myCtrl">
<p>{{lowercase}}</p>
<p>{{uppercase}}</p>
   <p>{{isString}}</p>
   <p>{{isNumber}}</p>
</div>
```

```
var app = angular.module('myApp', []);
app.controller('myCtrl',function($scope) {
   $scope.name1 = "Yunus";
  $scope.lowercase= angular.lowercase($scope.name1);
   $scope.uppercase= angular.uppercase($scope.name1);
   $scope.isString= angular.isString($scope.name1);         // true
   $scope.isNumber = angular.isNumber($scope.name1);
});
```



```
yunus

YUNUS

true

false
```