Angular JS

- **Purpose of Angular JS:** It is used to create dynamic web based applications
- **Goal :** Enhancing the functionality of user interface and handling interactions with the user.
- Single Page Applications (SPA) : It loads only a *single web document, and then updates the body content of that single document via JavaScript APIs such as Fetch when different content is to be shown*.
- It is a primary *front end java script framework.*
- It calls actual HTML *elements to create dynamic web pages*.
- **Difference between Angular JS and javascript**

S.No.	Javascript	Angular JS
1.	It focuses to create interactive web pages.	It focuses to create Single Page Applications (SPA)
2.	developers write code in plain text	It calls actual HTML elements to create dynamic web pages.
3.	It follows own syntax	It works with HTML's build-in elements.
4.	It is open source and object oriented programming language	It is framework based on the MVC model.
5.	It is developed by Netscpae	It is developed by Google
6.	It is faster than angular JS	It is slower

- **ng-app directive:** It defines the *root element of the Angular JS application*
  - It is automatically initialized the application when a web page is loaded.
  - It is also used to *load various angular JS modules* in angular JS application.
  -
- **Ng-init:**

  It is used to initialize the values of the application data.

Model View Controller (MVC) Architecture

- **Goal:** It is leading frameworks is used to *build JS heavy single page based web applications.* (create dynamic web pages)
- It is a software design pattern for *developing web applications*.
- It isolates the application logic from the user interface.
- **Three parts of MVC Architecture:**
  1. **Model** (HTML)
     - It is the *lowest level of the pattern*
     - It is responsible for **maintaining data** (managing application data)
     - It responds to the *requests from view and to the instructions from controller to update itself*.
     - The DOM is *connected to scope variables, and bindings are used to access the variable properties*.
     - ***Example***: ng-model directives bind the input fields to the controller properties.
  2. **View** (Scope)
     - It is responsible for *displaying all or portion of the data* of the user
     - It is used to present the data in required format
     - It doesn’t use standard HTML for view of data
     - Data-bound HTML describes the data.
     - ***HTML tags that use data-binding can better render dynamic data***.
  3. **Controller** (Javascript)
     - It is a code *controls the interaction between the model and view*.
     - It receives all the *requests for the application and then works with the model to prepare the needed data for view*.
     - It receives the input, validate and it performs the business logic.
     - **Syntax for defining controller object:**

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

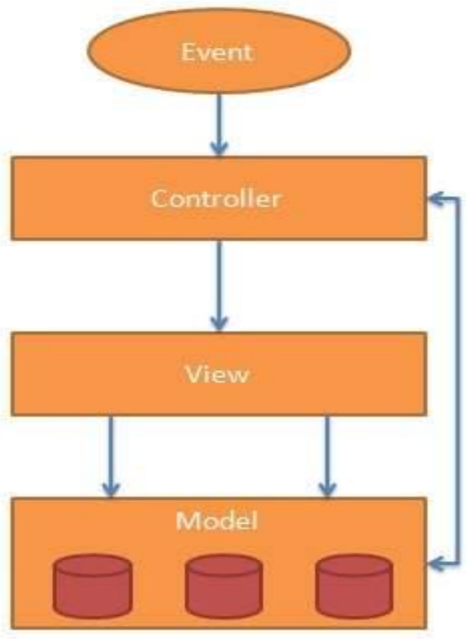
       Definition of Controller:

       ```
       <script>
       ```

```
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope) {
  $scope.firstName = "John";
  $scope.lastName = "Doe";
});
</script>
```

- The $scope is the application object (the owner of application variables and functions).

❖ **<u>Diagrammatic Representation of MVC Architecture</u>**



❖ **<u>Empty dependency array:</u>** it is used to define dependency modules of angular JS libraries.

❖ **<u>rootscope:</u>**

- All applications have a **<u>$rootScope</u>** which is the scope created on the HTML element that contains the ng-app directive.
- The *<u>rootScope is available in the entire application</u>*.

❖ **<u>$scope Object:</u>**

- It is an *<u>application object</u>* (it act as a owner of application variables and functions).
- AngularJS will invoke the controller with a $scope object.
- It is **<u>a child object</u>** that is used to *<u>bind the HTML (view) & Javascript (Controller) in a webpage</u>*.
- It is created with **the ng-app directive by using ng-controller directive**.

- When ***adding properties to the $scope object in the controller***, the view (HTML) gets access to these properties.
- **Example**:

```
<script>
       var app = angular.module('myApp', []);
       app.controller('personCtrl', function($scope) {
         $scope.firstName = "John";
         $scope.lastName = "Doe";
         $scope.fullName = function() {
           return $scope.firstName + " " + $scope.lastName;
         };
       });
</script>
```

❖ **run()**

- It is used for *tasks that need to be executed once when the application starts up*, while controller() is used for defining the behavior of specific parts of the UI.

❖ **Sample Angular JS code:**

```
<html>
<head>
  <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.5/angular.min.js">
</script>
</head>
<body ng-app="myapp">
    <div ng-controller="HelloController" >
            <h2>Hello {{a}} !</h2>
    </div>
 <script>
    angular.module("myapp",[]).controller("HelloController", function($scope) {
$scope.a = "CSE Department";  } );
</script>
</body>
</html>
```

❖ **Code to display the textbox content:**

```
<html>
```

```
<head>
   <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.5/angular.min.js"></script>
</head>
<body ng-app="myapp" ng-controller="bb">
<input type="text"  ng-model="name"></input>
   <p>{{name}}</p>
   <script>
      angular.module("myapp", []).controller("bb", function($scope) {
         $scope.name = "";
      });
   </script>
</body>
</html>
```

- **$interval:**
  - $interval service is a function that allows to *execute a function repeatedly at a specified interval*.

- **Program for change the color of the text of text every 1seconds:**

```
<html>
<head>
   <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.5/angular.min.js"></script
>
  <style>
    .col1
    {
            color:red;
    }
    .col2
    {
            color:blue;
    }
```

```
</style>
</head>
<body ng-app="myapp" ng-controller="bb">
<input type="text"  ng-model="name"></input>
   <p  ng-class="t">{{name}}</p>

   <script>
      angular.module("myapp", []).controller("bb", function($scope, $interval) {
        $scope.name="";
           var c=["col1", "col2"];
           var ci=0;
           $scope.t=c[ci];
           $interval(function() {
                   ci = (ci+1) % c.length;
                   $scope.t=c[ci];
           }, 1000);
      });
   </script>
</body>
</html>
```

❖ **<u>Program for addition of 2 Numbers</u>**

```
<html>
<head>
   <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.5/angular.min.js"></script
>
</head>
<body ng-app="myapp"  ng-controller="gg">
    <form>
           <input ng-model="a"   ></input>
           <input ng-model="b"  ></input>
```

```
                <input ng-model="c" ></input>
                <input type="submit" ng-click="clic()"></input>
    </form>
<script>
    angular.module("myapp",[]).controller("gg",function($scope)
    {
                $scope.a =0;
                $scope.b=0;
                $scope.c=0;
                $scope.clic=function()
                {
                        $scope.c=parseInt($scope.a) + parseInt($scope.b);
                };

    });
</script>
</body>
</html>
```

❖ **Addition of two numbers** (immediate change)

```
<html>
<head>
   <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.5/angular.min.js"></script
>
</head>
<body ng-app="myapp"  ng-controller="gg">
    <form>
                <input ng-model="a"   ng-change="cal()"></input>
                <input ng-model="b"  ng-change="cal()"></input>
                <input ng-model="c" ></input>
    </form>
    <script>
        angular.module("myapp", []).controller("gg", function($scope) {
            $scope.a = 0;
            $scope.b = 0;
            $scope.c = 0;
            $scope.cal = function() {
                $scope.c = parseInt($scope.a || 0) + parseInt($scope.b || 0)

            };
        });
    </script></body>
</html>
```

❖ **Program to change the font-style and font color:**

```
<html>
<head>
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.5/angular.min.js"></script
>
   <style>
    .col1
    {
            color:red;
    }
    .col2
    {
            color:blue;
    }
</style>
</head>
<body ng-app="myapp" ng-controller="bb">
<input type="text"  ng-model="name"></input>
```

```
    <p  ng-class="t"  ng-style="{'font-size':f+'px'}">{{name}}</p>

    <script>
      angular.module("myapp", []).controller("bb", function($scope, $interval) {
        $scope.name="";
            var c=["col1", "col2"];
            var ci=0;
            $scope.t=c[ci];
            $scope.f=12;
            $interval(function() {
                    ci = (ci+1) % c.length;
                    $scope.t=c[ci];
                    $scope.f = $scope.f+2;
            }, 1000);

      });
    </script>
  </body>
  </html>
```

Directives / ng attributes

- **Purpose:** It defines a *specified behaviour to that specific DOM element or transform the DOM element and its children*. (extend HTML with new attributes)
- **It is a central part of Angular JS**
- It is a marker on a DOM element (attribute, element name, comment or CSS class)
- It enables to create *reusable, modular, and expressive components* in any application, *enhancing code maintainability and developer productivity*.
- **Different types of Angular Directives:**

Directive	Purpose

ng-app	It defines the root element of an application.
ng-bind	It binds the content of an html element to application data.
ng-bind-template	It specifies that the text content should be replaced with a template.
ng-blur	It specifies a behaviour on blur events.
ng-change	It specifies an expression to evaluate when content is being changed by the user.
ng-checked	It specifies if an element is checked or not.
ng-class	It specifies css classes on html elements.
ng-click	It specifies an expression to evaluate when an element is being clicked.
ng-cloak	It prevents flickering when your application is being loaded. (<p ng-cloak>) → avoiding flickering effect
ng-controller	It defines the controller object for an application.
ng-copy	It specifies a behaviour on copy events. (example <p ng-copy="count = count + 1"> ffff </p>)
ng-csp	It changes the content security policy.
ng-cut	It specifies behaviour on cut events.
ng-dblclick	It specifies a behaviour on double-click events.
ng-focus	It specifies behaviour on focus events.
ng-hide	It hides or shows html elements.
ng-href	It specifies a URL for the <a> element.
ng-if	It removes the html element if a condition is false.
ng-include	It includes html in an application.
ng-init	It defines initial values for an application.
ng-keydown	It specifies a behavior on keydown events.
ng-keypress	It specifies a behavior on keypress events.
ng-keyup	It specifies a behavior on keyup events.
ng-list	It converts text into a list (array).
ng-open	It specifies the open attribute of an element.
ng-options	It specifies <options> in a <select> list.
ng-paste	It specifies a behavior on paste events.

ng-readonly	It specifies the readonly attribute of an element.
ng-required	It specifies the required attribute of an element.
ng-selected	It specifies the selected attribute of an element.
ng-show	It shows or hides html elements.
ng-src	It specifies the src attribute for the <img> element.
ng-style	It specifies the style attribute for an element.
ng-submit	It specifies expressions to run on onsubmit events.
ng-switch	It specifies a condition that will be used to show/hide child elements.
ng-value	It specifies the value of an input element.
ng-disabled	It specifies if an element is disabled or not.
ng-form	It specifies an html form to inherit controls from.
ng-model	It binds the value of html controls to application data.
ng-mousedown	It specifies a behavior on mousedown events.
ng-mouseenter	It specifies a behavior on mouseenter events.
ng-mouseleave	It specifies a behavior on mouseleave events.
ng-mousemove	It specifies a behavior on mousemove events.
ng-mouseover	It specifies a behavior on mouseover events.
ng-mouseup	It specifies a behavior on mouseup events.
ng-repeat	It defines a template for each data in a collection.

❖ **Example:** **ng-change** (ng-model must be necessary)

```
<html>
<head>
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.5/angular.min.js"></script
>
</head>
<body ng-app="demo" ng-controller="test">
```

```
                <form>
                        <input  ng-model="name" ng-change="hello()"></input>
                </form>
                <p>{{val}}</p>
<script>
        angular.module("demo",[]).controller("test",function($scope){
                $scope.val=0;
                $scope.hello=function()
                {
                        if($scope.name.length==0)
                                $scope.val=0;
                        else
                                $scope.val++;
                };
        });
</script>
</body>
</html>
```

❖ **Example 2:** ng-click

```
<html>
<head>


    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.5/angular.min.js"></script>
</head>
<body ng-app="demo" ng-controller="test" ng-click="ss()">
                <form>
                        <input  ng-model="name" ></input>
                </form>
                <p ng-style="{'color':f}">Welcome {{name}}</p>
<script>
```

```
        angular.module("demo",[]).controller("test",function($scope){
                $scope.name=0;
                $scope.c=0;
                $scope.f="red";
                $scope.ss=function()
                {
                        if ($scope.f=="red")
                                $scope.f="green";
                        else
                                $scope.f="red";
                };
        });
    </script>
    </body>
    </html>
```

❖ **Example 3:** ng-repeat and ng-bind, ng-blur, ng-copy, ng-hide, ng-if

```
<html>
<head>
            <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script
>
</head>
<body  ng-app="test"  ng-controller="ff">
            <form>
                    <input ng-model="a"    ng-blur="dd()"></input>
                    <input type="checkbox"   ng-model="c" > Click me </input>
            </form>
            <p  ng-style="{'color': f}"   ng-bind="a">  </p>
            <div ng-if="c">
            <ol>
            <li ng-repeat = "x in b">     {{x}} </li>
```

```
                </ol>
                </div>
                <p ng-hide="gh">  hhhii  </p>
                <script>
                        angular.module("test",[]).controller("ff", function($scope,
$interval, $location,$window)
                        {
                                $scope.a = "";
                                $scope.b = [1,2,3,4,5,6,7,8];
                                $scope.gh=true;
                                $scope.dd=function()
                                {
                                        $window.alert("focus gone");
                                        $scope.gh=!$scope.gh;
                                };

                        });
                </script>
</body>
</html>
```

❖ **Example 4:**

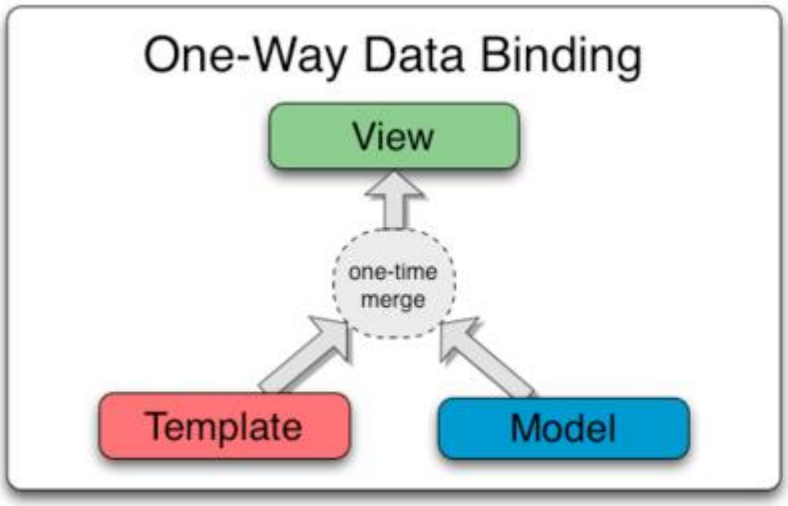
```
 <html>
 <head>

      <script
 src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.5/angular.min.js"></script
 >
 </head>
 <body ng-app="demo" ng-controller="test" ng-click="ss()">
                <form>
                        <input  ng-model="name" ></input>
                        <input type="checkbox" ng-model="v">Clcik here </input>
                </form>
                <p ng-style="{'color':f}"   ng-if="v" ng-change="clickme()">
                        Welcome {{name}}</p>
 <script>
     angular.module("demo",[]).controller("test",function($scope){
                $scope.name=0;
                $scope.c=0;
                $scope.f="red";
                $scope.v = "false";
                $scope.ss=function()
                {
                        if ($scope.f=="red")
                            $scope.f="green";
                        else
                            $scope.f="red";
                };
                $scope.clickme = function()
                {
                        $scope.v = !$scope.v;
                }
     });
 </script>
 </body>
 </html>
```

Expression and Data binding

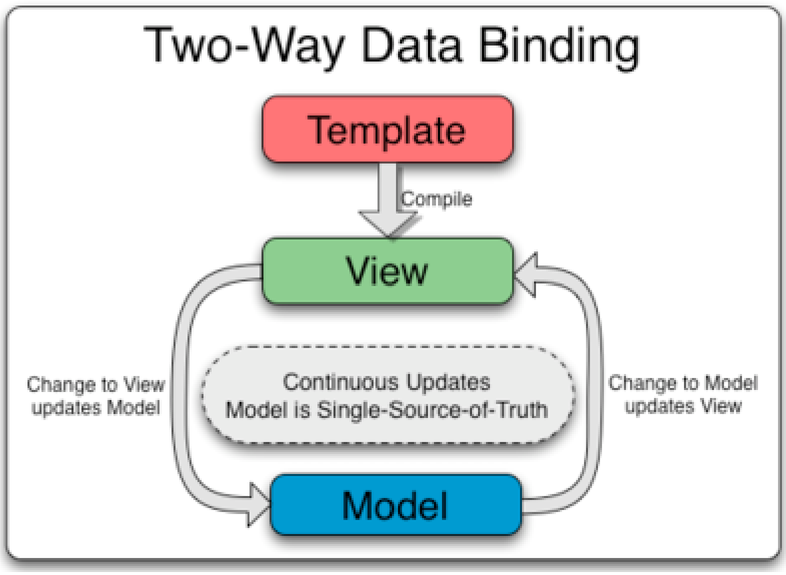
- Expression is used to *bind the data with the html element*.
- Expression must be *written inside of the HTML element*. (in java script expression must be defined inside of script tag).

- ❖ Data binding: It provides a way to *synchronize the data between the model and view components automatically.*
- ❖ **Principle used**: Single- Source of Truth (aggregating the data from many systems to a single location)
- ❖ Data binding is *act as a bridge between the view and business logic of the application*.
- ❖ **Types of data binding**:

1. One way data binding
   - The *value is taken from the data model and inserted into an HTML element.* There is no way to update model from view



2. **Two way data binding**
   - It provides the automatic synchronization of data between the model and view components.


Two-Way Data Binding
Template
Compile
View
Change to View updates Model
Continuous Updates Model is Single-Source-of-Truth
Change to Model updates View
Model

❖ **<u>Syntax for define expression:</u>** {{ expression }} or ng-bind = “expression”

ng-model = “name”

❖ **<u>Example:</u>**

```
<html>
<head>
   <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.5/angular.min.js"></script
>
</head>
<body ng-app="myapp"  ng-controller="gg">
    <form>
              <input ng-model="a"   ng-change="cal()"></input>
              <input ng-model="b"  ng-change="cal()"></input>
              <input ng-model="c" ></input>
    </form>
   <script>
      angular.module("myapp", []).controller("gg", function($scope) {
         $scope.a = 0;
         $scope.b = 0;
         $scope.c = 0;
         $scope.cal = function() {
            $scope.c = parseInt($scope.a || 0) + parseInt($scope.b || 0)

          };
        });
      </script></body>
   </html>
```

Forms in Angular JS

- **Goal**: It *provides data-binding and validation of input controls*
- **ng-model directive**: Input controls provides data binding using ng-model
- Angular js **provides client side validation**.
- **Input controls used in Angular JS:**
  1. Input elements
  2. Select elements
  3. Button Elements
  4. Textarea Elements
- **Different events supported in Angular JS:**

1. ng-click	5. ng-mouseenter	9. ng-keydown
2. ng-dbl-click	6. ng-mouseleave	10. ng-keyup
3. ng-mousedown	7. ng-mousemove	11. ng-keypress
4. ng-mouseup	8. ng-mouseover	12. ng-change

- **Example program for checkbox:**

```
<html>
<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></sc
ript>
<body>
<div ng-app="">
  <form>
    Check to show this:
    <input type="checkbox" ng-model="myVar">
  </form>
  <h1 ng-show="myVar">Checked</h1>
</div>
<p>The ng-show attribute is set to true when the checkbox is checked.</p>
</body>
</html>
```

- **Radio Buttons:**

```
<html>
<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script
>
<body ng-app="">
<form>
  Pick a topic:
  <input type="radio" ng-model="myVar" value="dogs">Dogs
  <input type="radio" ng-model="myVar" value="tuts">Tutorials
  <input type="radio" ng-model="myVar" value="cars">Cars
</form>
<div ng-switch="myVar">
  <div ng-switch-when="dogs">
    <h1>Dogs</h1>
    <p>Welcome to a world of dogs.</p>
  </div>
  <div ng-switch-when="tuts">
    <h1>Tutorials</h1>
    <p>Learn from examples.</p>
  </div>
  <div ng-switch-when="cars">
    <h1>Cars</h1>
    <p>Read about cars.</p>
  </div>
</div>
```

```
</body>
</html>
```

- **<u>Select:</u>**

```
<html>
<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script
>
<body ng-app="">
<form>
  Select a topic:
  <select ng-model="myVar">
    <option value="">
    <option value="dogs">Dogs
    <option value="tuts">Tutorials
    <option value="cars">Cars
  </select>
</form>
<div ng-switch="myVar">
  <div ng-switch-when="dogs">
     <h1>Dogs</h1>
     <p>Welcome to a world of dogs.</p>
  </div>
  <div ng-switch-when="tuts">
     <h1>Tutorials</h1>
     <p>Learn from examples.</p>
  </div>
  <div ng-switch-when="cars">
     <h1>Cars</h1>
     <p>Read about cars.</p>
  </div>
</div>
```

```
</body>
</html>
```

- **Validation:**

Case1:

```
<form name="myForm">
<input type="email" name="myInput" ng-model="myInput">
</form>
<p>The input's valid state is:  {{myForm.myInput.$valid}}   </p>
```

- **Input Field States:** (property value – true / false)
  1. $untouched - The field has not been touched yet
  2. $touched - The field has been touched
  3. $pristine - The field has not been modified yet
  4. $dirty - The field has been modified
  5. $invalid - The field content is not valid
  6. $valid - The field content is valid
- **Form States:** (property value – true / false)
  1. $pristine - No fields have been modified yet
  2. $dirty - One or more have been modified
  3. $invalid - The form content is not valid
  4. $valid - The form content is valid
  5. $submitted - The form is submitted
- **Example for Input Validation:**

```
<html>
<head>
   <style>
          input.ng-valid
          {
                 background-color:pink;
          }
          input.ng-invalid
          {
```

```
                    background-color:red;
            }
    </style>
<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script
>
</head>
<body ng-app="demo"  ng-controller="Test">
<form  name="s">
    <input type="text" name="a" ng-model="a" required  pattern="[a-z]+">
</input>
    <p> {{a}}</p>
    <span ng-show="s.a.$touched && s.a.$invalid" ng-style="{color: 'red',
fontSize: '15px'}">Name field is required</span>
    <br>
    <input type="email" ng-model="b" required> </input>

</form>
</body>
</html>
```

Filters in Angular JS

- **Purpose**: It is sued to format the value of the expression for display to the user. (format data)
- **Transformations:**
    1. Formatting Numbers
    2. Dates
    3. Currencies
    4. Manipulating Arrays and Strings
- **Different formatting:**
    1. Currency   {{ a | currency}}
    2. Date   {{a | date:’MM/dd/yyyy’}}

3. String formation {{a | uppercase}}   / {{a | lowercase}}
4. Number of digits after decimal point {{a | number:2}}
5. Limit array size {{a | limitTo :2}}
6. Filter → Select a subset of items from an array

   Example 1:

   ```
   <ul>
     <li ng-repeat="x in names | filter : 'i'">
       {{ x }}
     </li>
   </ul>
   ```

   Example 2:

   ```
   <p><input type="text" ng-model="test"></p>

   <ul>
     <li ng-repeat="x in names | filter:test">
       {{ x }}
     </li>
   </ul>
   ```

7. orderBy → order an array by an expression

   ```
   <ul>
     <li ng-repeat="x in names | orderBy:'country'">
       {{ x.name + ', ' + x.country }}
     </li>
   </ul>
   <script>
   angular.module('myApp', []).controller('namesCtrl', function($scope) {
      $scope.names = [
         {name:'Jani',country:'Norway'},
         {name:'Carl',country:'Sweden'},
         {name:'Margareth',country:'England'},
         {name:'Hege',country:'Norway'},
   ```

```
                {name:'Joe',country:'Denmark'},
                {name:'Gustav',country:'Sweden'},
                {name:'Birgit',country:'Denmark'},
                ];
        });
```

8. Format an object to a JSON string → json

Services in Angular JS

- **Definition of Service**: It is a function or an object that *avails or limit to the application in AngularJS.*
- Services are javascript functions which are responsible to perform specific tasks. (call the controllers and filters based on the requirement)
- Services are injected into the Angular JS application using **dependency** injection mechanism.
- **Different Services:**
    1. $http services – request and response services
    2. $timeout services
    3. $interval service
    4. $controller
    5. $document
    6. $exceptionHandler
    7. $filter
    8. $xhrFactory
    9. $httpBackend
    10. $locale
    11. $location$parse
    12. $parse → it is used to convert any expression into appropriate functions.
    13. $window
    14. $templateRequest
    15. $animateCss
    16. $animate
- **s**

UI / UX framework

- **Purpose of UI / UX framework:** It is a *structured approach that designers* follow to *create consistent and user-friendly digital products, websites, or applications*.
  - User Interface Design (UI)
  - User Experience (UX)
- **5 Elements of UI / UX Framework:** (First 4 – UX and last – UI)
  1. Strategy (User needs, Business Goals)
  2. Scope (Functional and content requirements)
  3. Structure (Information Architecture and interactive design)
  4. Skeleton (Interface Design, Navigation Design, information Design)
  5. Surface (Visual Design) - UI Design
- **Lifecycle of UI/UX Design**
  1. Pre-design stage
  2. Design research
  3. Sketching
  4. Wire framing
  5. Visualization
  6. Slicing
- **Different Design Frameworks:**
  1. Material Design
  2. Bootstrap
  3. Ant Design
  4. Semantic UI
  5. Materialize CSS
- s