AngularJS

People matter, results count.

Table of Contents

- **Following topics will be covered**

AngularJs Introduction

- AngularJS (commonly referred to as "Angular.js" or "AngularJS 1.X") is a JavaScript-based open-source front-end web application framework mainly maintained by Google and by a community of individuals and corporations to address many of the challenges encountered in developing single-page applications.
- The JavaScript components complement Apache Cordova, the framework used for developing cross-platform mobile apps.
- It aims to simplify both the development and the testing of such applications by providing a framework for client-side model–view–controller (MVC) and model–view–viewmodel (MVVM) architectures, along with components commonly used in rich Internet applications.

Core Features

- Data-binding − It is the automatic synchronization of data between model and view components.
- Scope − These are objects that refer to the model. They act as a glue between controller and view.
- Controller − These are JavaScript functions that are bound to a particular scope.
- Services − AngularJS come with several built-in services for example $https: to make a XMLHttpRequests. These are singleton objects which are instantiated only once in app.
- Filters − These select a subset of items from an array and returns a new array.
- Directives − Directives are markers on DOM elements (such as elements, attributes, css, and more). These can be used to create custom HTML tags that serve as new, custom widgets. AngularJS has built-in directives (ngBind, ngModel...)

Core Features(Contd…)

- Templates – These are the rendered view with information from the controller and model. These can be a single file (like index.html) or multiple views in one page using "partials".
- Routing – It is concept of switching views.
- Model View Whatever – MVC is a design pattern for dividing an application into different parts (called Model, View and Controller), each with distinct responsibilities. AngularJS does not implement MVC in the traditional sense, but rather something closer to MVVM (Model-View-ViewModel). The Angular JS team refers it humorously as Model View Whatever.
- Deep Linking – Deep linking allows you to encode the state of application in the URL so that it can be bookmarked. The application can then be restored from the URL to the same state.
- Dependency Injection – AngularJS has a built-in dependency injection subsystem that helps the developer by making the application easier to develop, understand, and test.

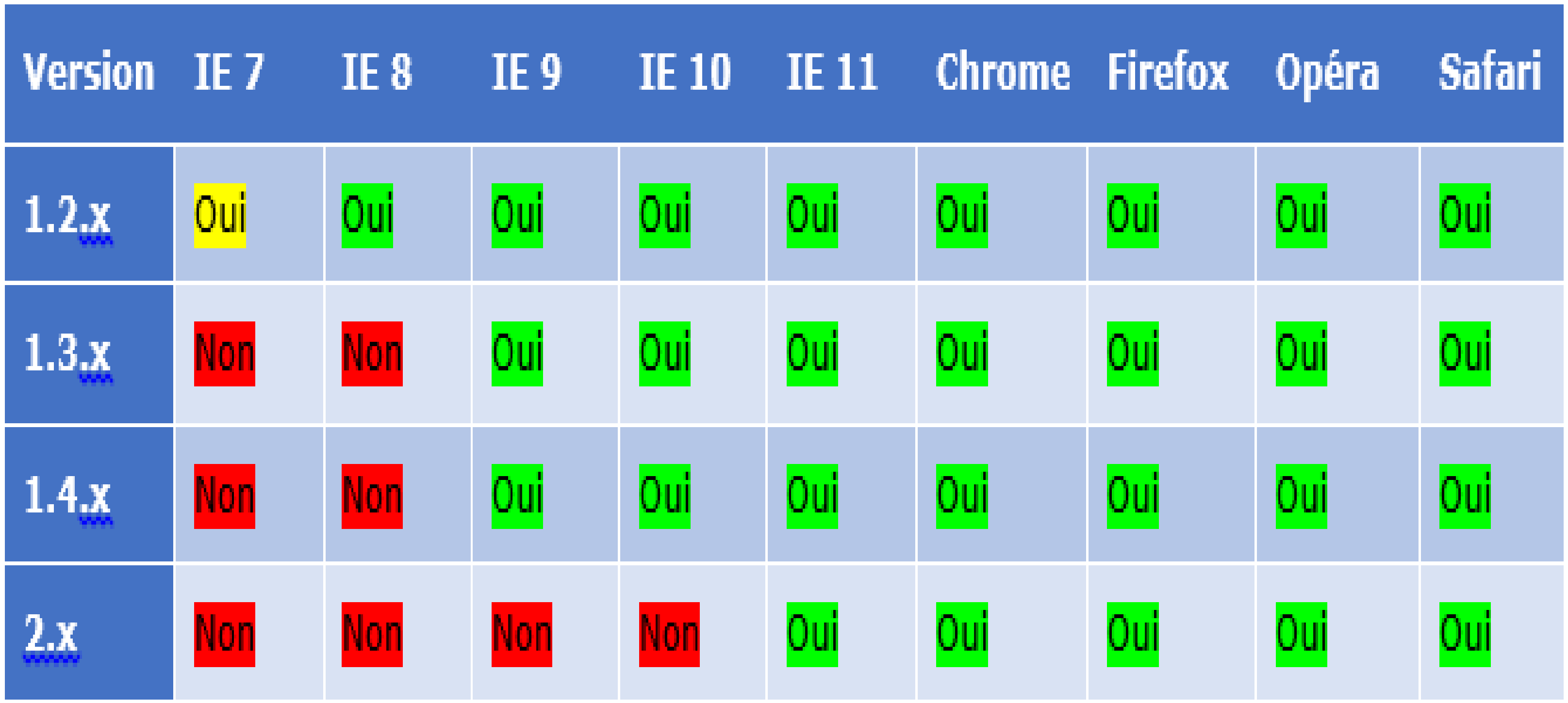
Angular Advantages

- AngularJS provides capability to create Single Page Application in a very clean and maintainable way.
- AngularJS provides data binding capability to HTML thus giving user a rich and responsive experience
- AngularJS code is unit testable.
- AngularJS uses dependency injection and make use of separation of concerns.
- AngularJS provides reusable components.
- With AngularJS, developer write less code and get more functionality.
- In AngularJS, views are pure html pages, and controllers written in JavaScript do the business processing.
- On top of everything, AngularJS applications can run on all major browsers and smart phones including Android and iOS based phones/tablets.

Angular disadvantages

- Not Secure – Being JavaScript only framework, application written in AngularJS are not safe. Server side authentication and authorization is must to keep an application secure.
- Not degradable – If your application user disables JavaScript then user will just see the basic page and nothing more.

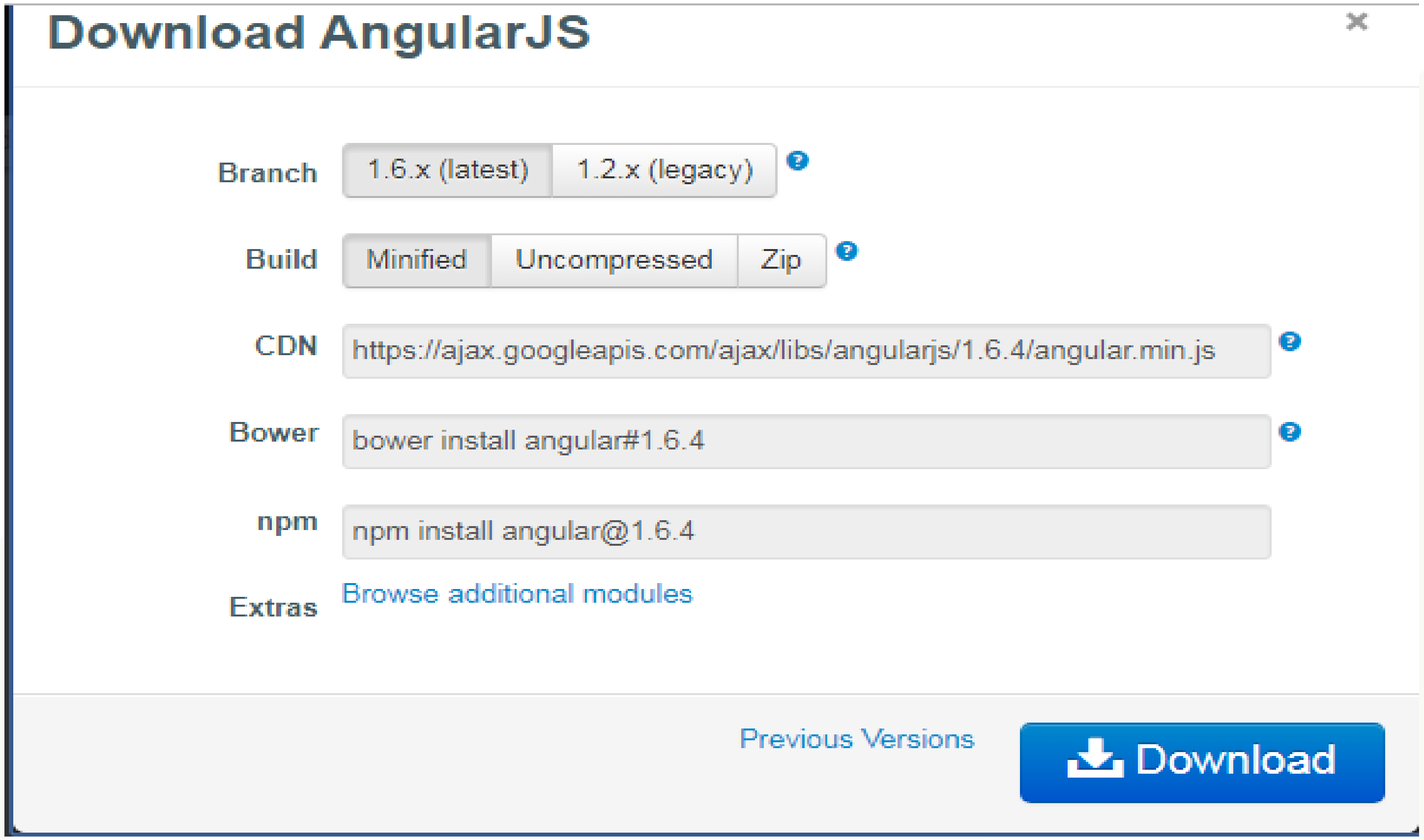
AngularJs Support



Version	IE 7	IE 8	IE 9	IE 10	IE 11	Chrome	Firefox	Opéra	Safari
1.2.x	Oui	Oui	Oui	Oui	Oui	Oui	Oui	Oui	Oui
1.3.x	Non	Non	Oui	Oui	Oui	Oui	Oui	Oui	Oui
1.4.x	Non	Non	Oui	Oui	Oui	Oui	Oui	Oui	Oui
2.x	Non	Non	Non	Non	Oui	Oui	Oui	Oui	Oui

Angular Installation Steps

- Open the https://www.angularjs.org/ link in browser.
- Click “Download AngularJs” button. After clicking copy the Url to Script tag of your code snippet.



Download AngularJS

Branch	1.6.x (latest) / 1.2.x (legacy)
Build	Minified / Uncompressed / Zip
CDN	https://ajax.googleapis.com/ajax/libs/angularjs/1.6.4/angular.min.js
Bower	bower install angular#1.6.4
npm	npm install angular@1.6.4
Extras	Browse additional modules

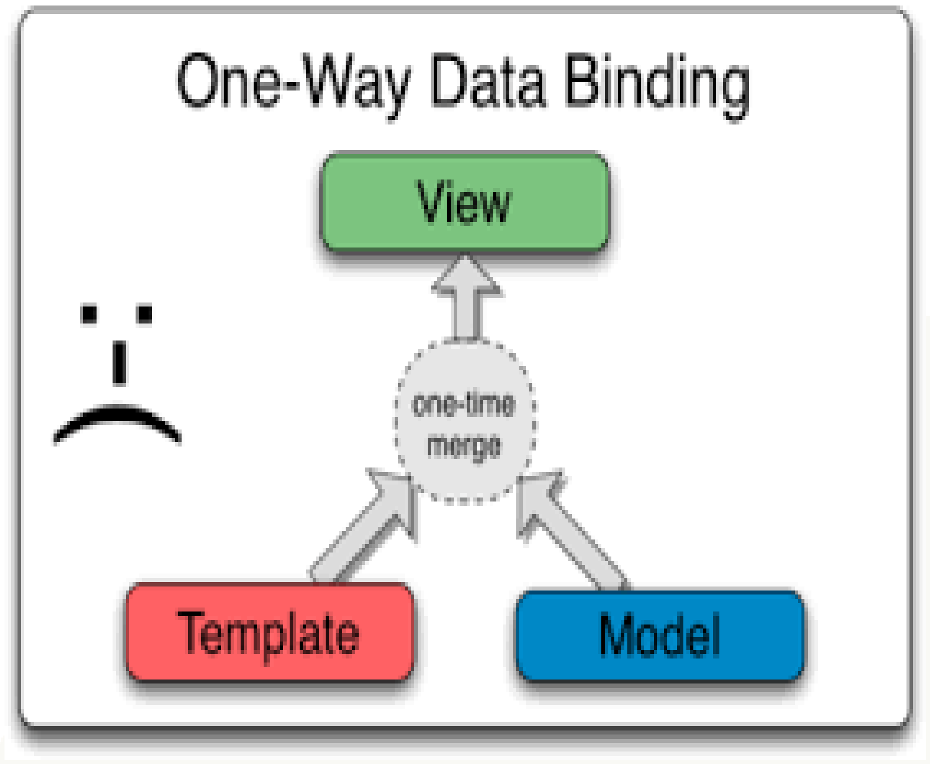
Angular IDE and Tools

- Web Storm
- Eclipse
- Sublime Text
- Aptana Studio
- Notepad++
- Plunker

Testing & Debugging Tools

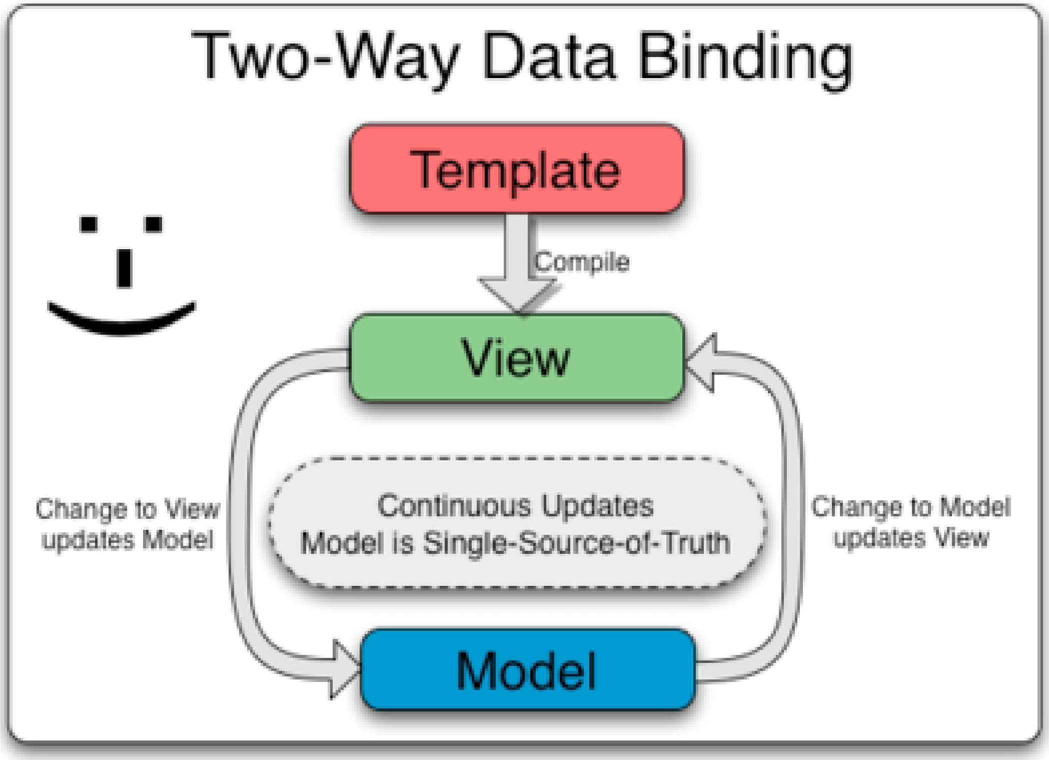
- Karma
- Jasmine
- MochaJS
- Protractor
- Angular Batarang
- Ns Inspector

Data Binding



- The one-way data binding is an approach where a value is taken from the data model and inserted into an HTML element.
- There is no way to update model from view. It is used in classical template systems.
- These systems bind data in only one direction.

Data Binding(Contd…)


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

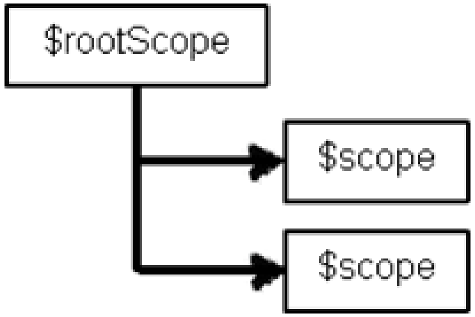
- Data-binding in Angular apps is the automatic synchronization of data between the model and view components.
- Data binding lets you treat the model as the single-source-of-truth in your application.
- The view is a projection of the model at all times.
- If the model is changed, the view reflects the change and vice versa.

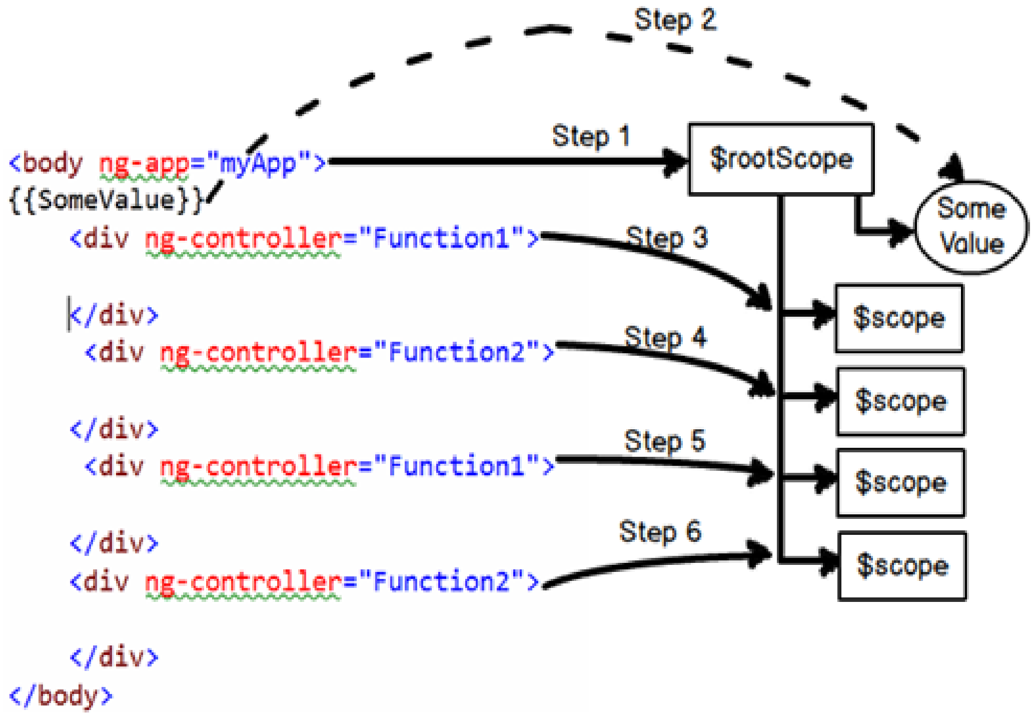
$scope and $rootscope

- $rootScope" is a parent object of all "$scope" angular objects created in a web page.
- The $scope in angularjs is an application object. It holds all the variables and functions of controller and allow us to access those variables and function in application view.
- 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 use the one in the current scope.

- Note:$scope is created with ng-controller while $rootscope is created with ng-app.

$scope and $rootscope(contd…)





```
<body ng-app="myApp">
{{SomeValue}}
    <div ng-controller="Function1">

    </div>
     <div ng-controller="Function2">

    </div>
     <div ng-controller="Function1">

    </div>
    <div ng-controller="Function2">

    </div>
</body>
```

Angular Modules

- In angularjs applications we don’t have that main method instead we have modules that specify how our application will be structured and bootstrapped.
- AngularJS supports modular approach.
- Modules are used to separate logics say services, controllers, application etc. and keep the code clean.
- We define modules in separate js files and name them as per the module.js file.

- Below is the Syntax for creating angular modules:

```
<script type="text/javascript">
 var app = angular.module('angularmoduleapp', []);
 </script>
 <div ng-app="angularmoduleapp">....</div>
```

Angular Services

- In angularjs service is the function which is used to handle the server communication over the browser with help of XMLHttpRequest object and $http.
- In angularjs we have 30 built in services like $http, $location, $timeout, $interval, etc… and these are used to share the data and its behaviours in the controller, directive, filters and other services over the apps.
- In angularjs we can create our own custom services also based on our requirement.
- In angularjs service will create singleton instance over the angular apps and call the services using the service name in the controller.
- Out of 30 built in services,$http,$route is commonly used services in the applications.
- User defined services are created in 2 ways.
  - Factory method
  - Service method

Angular Services

- Using factory 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**
- Using service method, we define a service and then assign method to it. We've also injected an already available service to it.

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

$httpService

- AngularJS $http service allows us to communicate asynchronously with server endpoints using XHR [browser's XMLHttpRequest Object] API.
- $http is designed for general purpose AJAX calls that can deal with both RESTful & Non-RESTful API on your server.
- The $http API is based on the deferred/promise APIs exposed by the $q service which is an implementation of Promise interface, based on Kris Kowal’s Q approach, which is a standardized way of dealing with asynchronous calls in JavaScript.
- The Promise API comes with following flow:
  - 1. Each asynchronous task will return a promise object.
  - 2. Each promise object will have a then function that can take two arguments, a success handler and an error handler.
  - 3. The success or the error handler in the then function will be called only once, after the asynchronous task finishes.
  - 4. The then function will also return a promise, to allow chaining multiple calls.
  - 5. Each handler (success or error) can return a value, which will be passed to the next function in the chain of promises.
  - 6. If a handler returns a promise (makes another asynchronous request), then the next handler (success or error) will be called only after that request is finished.

$httpService(contd…)

- Let’s take a simple example to understand $http.

```
// Simple GET request example :
$http.get('/someUrl').
  then(function(response) {
    // this callback will be called asynchronously when the response is available
  }, function(response) {
    // this callback will be called asynchronously if an error occurs or server returns response
with an error status.
  });
```

- In above example, then function takes two callbacks as parameter. First parameter gets called if the response is success. In case of failure, second parameter gets called. Both of callback parameters takes response as input.
- The response object has these properties:
- data – {string|Object} – The response body transformed with the transform functions.
  status – {number} – HTTP status code of the response.
  headers – {function([headerName])} – Header getter function.
  config – {Object} – The configuration object that was used to generate the request.
  statusText – {string} – HTTP status text of the response.

Angular Filters

- In Angular, filters are used to format data.

```
Syntax:   {{expression | filter}}
               or
{{ expression | filter1 | filter2 | ... }}
               or
{{ expression | filter:argument1:argument2:... }}
```

Note:In templates, filters are only executed when their inputs have changed. This is more performant than executing a filter on each $digest as is the case with expressions.

Angular Filters(Contd…)

Filter	Description
Currency	It formats a number to a currency format.
Date	It formats a date to a specified format.
Filter	It select a subset of items from an array.
Json	It formats an object to a Json string.
Limit	It is used to limit an array/string, into a specified number of elements/characters.
Lowercase	It formats a string to lower case.
Number	It formats a number to a string.
Orderby	It orders an array by an expression.
Uppercase	It formats a string to upper case.

Angular Directives

- AngularJS facilitates you to extend HTML with new attributes. These attributes are called directives.
- There is a set of built-in directive in AngularJS which offers functionality to your applications. You can also define your own directives.
- Directives are special attributes starting with ng- prefix. Following are the most common directives:
- **ng-app:** This directive starts an AngularJS Application.
- **ng-init:** This directive initializes application data.
- **ng-model:** This directive defines the model that is variable to be used in AngularJS.
- **ng-repeat:** This directive repeats html elements for each item in a collection.

Angular Forms

- AngularJS facilitates you to create a form enriches with data binding and validation of input controls.
- Input controls are ways for a user to enter data. A form is a collection of controls for the purpose of grouping related controls together.

  Different Form states in AngularJS:

  - $invalid : AngularJS sets this state when any of the validations (required, ng-minlength, and others) mark any of the fields within the form as invalid.
  - $valid : The inverse of the previous state, which states that all the validations in the form are currently evaluating to correct.
  - $pristine : All forms in AngularJS start with this state. This allows you to figure out if a user has started typing in and modifying any of the form elements. Possible usage: disabling the reset button if a form is pristine.
  - $dirty : The inverse of $pristine, which states that the user made some changes (he can revert it, but the $dirty bit is set).
  - $error : This field on the form houses all the individual fields and the errors on each form element.

Angular Events

- AngularJS provides multiple events that can be associated with the HTML controls. These events are associated with the different HTML input elements.
- Following is a list of events supported in AngularJS:
  - ng-click
  - ng-dbl-click
  - ng-mousedown
  - ng-mouseup
  - ng-mouseenter
  - ng-mouseleave
  - ng-mousemove
  - ng-mouseover
  - ng-keydown
  - ng-keyup
  - ng-keypress
  - ng-change

Routing

- AngularJS Routes enable us to implement multiview SPAs [Single Page Applications].
- A multiview application would consist of multiple views[HTML Templates where each template is associated with a specific route] which would be loaded dynamically as a result of user action (clicking a link, typing a specific URL in browser e.g.).
- Using routing, AngularJS application can show different content based on which route is selected. Routes are basically bookmark-able URL's to specific content [specific view] of AngularJS application.

Hashbang URL's

- AngularJS Routing leverages hashbang URLs, means routes are added after # in URL. For example, normally URL looks like http://www.angularapp.com/first/page.
- In a Single-Page Application, however, it would usually look like http://www.angularapp.com/#/first/page.
- This is due to the fact that Browser treats URL with # differently than standard URL.

Routing(Contd…)

1.Including the AngularJS Route Module source code in Application’s HTML

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.4/angular-route.js"></script>

It can be included in head or body section.

2.Include the ngRoute module as a dependency of our main AngularJS app module

angular.module('routingDemoApp',['ngRoute'])

Next step is to include the ‘ngRoute’ module as a dependency to the main module. It is required before we can use anything from this routing module in our code.

3.Using ngView directive

<div ng-view></div>

Next step is to mark which section of the page AngularJS should change when the route changes. With the ngRoute module, this is done using the ng-view directive in the HTML. Please note that with ngRoute, there can be only One ng-view per application. If your application demands more, you should prefer ui-router over ngRoute.

Routing(Contd…)

4.Configure $routeProvider

- Define our routes in the config section using the $routeProvider service. Below is the simplest configuration to configure routes using @routeProvider.

```
module.config(['$routeProvider', function($routeProvider){
   $routeProvider
      .when('/',{template:'This is the default Route'})
      .when('/computers',{template:'This is the computers Route'})
      .when('/printers',{template:'This is the printers Route'})
      .otherwise({redirectTo:'/'});
}]);
```

Angular Controllers

- AngularJS controllers are used to control the flow of data of AngularJS application.
- A controller is defined using ng-controller directive.
- A controller is a JavaScript object containing attributes/properties and functions.
- Each controller accepts $scope as a parameter which refers to the application/module that controller is to control.
- Example:

```
<div ng-app="myApp" ng-controller="myCtrl">
First Name: <input type="text" ng-model="firstName"><br>
Last Name: <input type="text" ng-model="lastName"><br>
<br>Full Name: {{firstName + " " + lastName}}
</div><script>
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope) {
    $scope.firstName = "John";
    $scope.lastName = "Doe";
});</script>
```

References

- https://en.wikipedia.org/wiki/Comparison_of_JavaScript_frameworks
- https://docs.angularjs.org/guide/filter

Recap

JavaScript

$http

Route

ng-model

Two way binding

Thank You For Your Time

People matter, results count.