## ****AngularJS Interview Questions****

### ****1. Explain data binding in AngularJS.****

According to AngularJS.org, “Data-binding in Angular apps is the automatic synchronization of data between the model and view components. The way that Angular implements 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. When the model changes, the view reflects the change, and vice versa.”

There are two ways of data binding:

1. Data mining in classical template systems
2. Data binding in angular templates

### ****2. Name the key features of AngularJS?****

The key features of AngularJS are:

* Scope
* Controller
* Model
* View
* Services
* Data Binding
* Directives
* Filters
* Testable

### ****3. What are directives in AngularJS?****

A core feature of AngularJS, directives are attributes that allow you to invent new HTML syntax, specific to your application. They are essentially functions that execute when the Angular compiler finds them in the DOM.  Some of the most commonly used directives are ng-app,ng-controller and ng-repeat.

The different types of directives are:

* Element directives
* Attribute directives
* CSS class directives
* Comment directives

**4. What are Controllers in AngularJS?**

Controllers are Javascript functions which provide data and logic to HTML UI. As the name suggests, they control how data flows from the server to HTML UI.

### ****5. What is factory method in AngularJS?****

Factory method is used for creating a directive.  It is invoked when the compiler matches the directive for the first time. We can invoke the factory method using $injector.invoke.

Syntax: module.factory( 'factoryName', function );  
Result: When declaring factoryName as an injectable argument you will be provided with the value that is returned by invoking the function reference passed to module.factory.

### ****6. What is ng-app, ng-init and ng-model?****

* ng-app : Initializes application.
* ng-model : Binds HTML controls to application data.
* ng-Controller : Attaches a controller class to view.
* ng-repeat : Bind repeated data HTML elements. Its like a for loop.
* ng-if : Bind HTML elements with condition.
* ng-show : Used to show the HTML elements.
* ng-hide : Used to hide the HTML elements.
* ng-class : Used to assign CSS class.
* ng-src : Used to pass the URL image etc.

### ****7. Does Angular use the jQuery library?****

Yes, Angular can use jQuery if it’s present in the app when the application is being bootstrapped. If jQuery is not present in the script path, Angular falls back to its own implementation of the subset of jQuery that we call jQLite.

### ****8. What is $rootscope in AngularJS?****

Every application has a single root scope. All other scopes are descendant scopes of the root scope. Scopes provide separation between the model and the view, via a mechanism for watching the model for changes. They also provide event emission/broadcast and subscription facility.

**8. AngularJS Data Bindings (One-way, Two-way) with Examples**

The **data binding** is the data synchronization processes that work between the model and view components. In Angular, model treat as source of application and view is the projection of angular model.

In angularjs when model data got changed that time the view data will change automatically and vice versa.

We have two types of data bindings available in angularjs those are

                1. One-Way data binding

                2. Two-Way data binding

We will learn each binding in detail with examples in angularjs.

**One-Way data binding**

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

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

</div>

<script>

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

app.controller('myCtrl', function($scope) {

$scope.firstname = "John";

$scope.lastname = "Doe";

});

</script>

**Two-Way data binding**

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

Name: <input ng-model="firstname">

<h1>{{firstname}}</h1></div>

<script>

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

app.controller('myCtrl', function($scope) {

$scope.firstname = "John";

});

**9. How to initialize angular.**

**1. Automatic Initialization**

**2. Manual Initialization**

**Automatic Initialization**

<!doctype html>

<html ng-app="optionalModuleName">

<body>

I can add: {{ 1+2 }}.

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

</body>

</html>

As a best practice, consider adding an ng-strict-di directive on the same element as ng-app:

<!doctype html>

<html ng-app="optionalModuleName" ng-strict-di>

<body>

I can add: {{ 1+2 }}.

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

</body>

</html>

**Manual Initialization**

<!doctype html>

<html>

<body>

<div ng-controller="MyController">

Hello {{greetMe}}!

</div>

<script src="http://code.angularjs.org/snapshot/angular.js"></script>

<script>

angular.module('myApp', [])

.controller('MyController', ['$scope', function ($scope) {

$scope.greetMe = 'World';

}]);

angular.element(function() {

angular.bootstrap(document, ['myApp']);

});

</script>

</body>

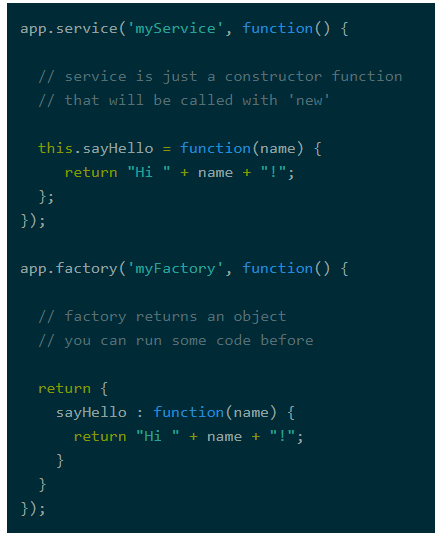
</html>

**10. Difference between service and factory**

**[Which one to use?](https://blog.thoughtram.io/angular/2015/07/07/service-vs-factory-once-and-for-all.html" \l "which-one-to-use)**

Asking that question on the internet takes us to a couple of articles and StackOverflow answers. The first is [this](https://stackoverflow.com/questions/13762228/confused-about-service-vs-factory) answer. It says:

“Basically the difference between the service and factory is as follows:”



1. **Difference between ng-show and ng-if?**

**ng-show**

ng-show (and its sibling ng-hide) toggle the appearance of the element by adding the CSS display: none and addclass ng-hide style.

**ng-if**

ng-if, on the other hand, actually ****removes**** the element from the DOM when the condition is false and only adds the element back once the condition turns true.

1. **Difference between custom filter and custom directive.**

**Custom Filter**

<!DOCTYPE html>

<html>

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.4/angular.min.js"></script>

<body>

<ul ng-app="myApp" ng-controller="namesCtrl">

<li ng-repeat="x in names">

{{x | myFormat}}

</li>

</ul>

<script>

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

app.filter('myFormat', function() {

return function(x) {

var i, c, txt = "";

for (i = 0; i < x.length; i++) {

c = x[i];

c = c.toUpperCase();

txt += c;

}

return txt;

};

});

app.controller('namesCtrl', function($scope) {

$scope.names = [

'Jani',

'Carl',

'Margareth',

'Hege',

'Joe',

'Gustav',

'Birgit',

'Mary',

'Kai'

];

});

</script>

</body>

</html>

# **$routeProvider vs $stateProvider in AngularJS**

**$routeProvider**

<body ng-app="myApp">

<a href="#!red">Red</a>

<div ng-view></div>

<script>

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

app.config(function($routeProvider) {

$routeProvider

.when("/red", {

templateUrl : "red.htm"

})

});

**$stateProvider**

$stateProvider

.state("contact", {

url: "/contact/",

templateUrl: '/app/Aisel/Contact/views/contact.html',

controller: 'ContactCtrl'

});

# **What is a digest cycle in angularJS?**

In each digest cycle Angular compares the old and the new version of the scope model values. The digest cycle is triggered automatically. We can also use $apply() if we want to trigger the digest cycle manually.

****15. What is the difference between one-way binding and two-way binding?****– One way binding implies that the scope variable in the html will be set to the first value its model is bound to (i.e. assigned to)  
– Two way binding implies that the scope variable will change it’s value everytime its model is assigned to a different value.

## **16.** $watch()

The $scope.watch() function creates a watch of some variable. When you register a watch you pass two functions as parameters to the $watch() function:

* A value function
* A listener function

Here is an example:

$scope.$watch(function() {},

function() {}

);

**17. **Define scope in AngularJS.****

Scope is a special JavaScript object that plays the role of joining the controller (JavaScript) with the views (HTML). The controller sets properties on the scope and the view binds to them.