**19199.** What is Routing in AngularJS?

**Answer:** AngularJS Routing helps you to divide your app into multiple views and bind different views to Controllers. The magic of Routing is taken care by an AngularJS service $routeProvider. $routeProvider service provides method when() and otherwise() to define the routes for your app. Routing has dependency on ngRoute module.

**19197.** What methods $resource service object support?

**Answer:** The $resource service object supports the following methods:

1. get()

2. query()

3. save()

4. remove()

5. delete()

**19196.** How to enable caching in $http service?

**Answer:** You can enable caching in $http service by setting configuration property cache to true. When cache is enabled, $http service stores the response from the server in local cache. In this way, next time the response will be served from the cache without sending request to server.

1. $http.get("http://server/myserviceapi",{ cache:true }).sucess(function(){
3. //TO DO:
5. });

**19194.** What methods $http service support?

**Answer:** The $http service supports the following methods:

1. $http.get()

2. $http.head()

3. $http.post()

4. $http.put()

5. $http.delete()

6. $http.jsonp()

7. $http.patch()

**19193.** What is the difference between $http and $resource?

**Answer:** $http service is a core Angular service which allows you to make AJAX requests by using GET, HEAD, POST, PUT, DELETE, JSONP and PATCH methods. It is very much like the $.ajax() method in jQuery. It can be used with RESTful and Non-RESTful server-side data sources.

$http is good for quick retrieval of server-side data that doesn’t really need any specific structure or complex behaviors.

$resource warps $http and allows you to interact with RESTful server-side data sources. It requires the ngResource module to be installed which exist in angular-resource.js

$http is good for retrieval of RESTful server-side data sources that might need any specific structure or complex behaviors.

**19191.** What is the difference between Factory, Service and Provider?

**Answer:** Factory - A factory is a simple function which allows you to add some logic before creating the object. It returns the created object.

When to use: It is just a collection of functions like a class. Hence, it can be instantiated in different controllers when you are using it with constructor function.

Service - A service is a constructor function which creates the object using new keyword. You can add properties and functions to a service object by using this keyword. Unlike factory, it doesn’t return anything.

When to use: It is a singleton object. Use it when you need to share a single object across the application.

For example, authenticated user details.

Provider - A provider is used to create a configurable service object. It returns value by using $get() function.

When to use: When you need to provide module-wise configuration for your service object before making it available.

**19189.** What are different ways to create service in AngularJS?

**Answer:** There are five ways to create a service as given below:

1. Service

2. Factory

3. Provider

4. Value

5. Constant

**19188.** What is Service in AngularJS?

**Answer:** A service is a reusable singleton object which is used to organize and share code across your app. A service can be injected into controllers, filters, directives.

AngularJS offers several built-in services (like $http, $provide, $resource, $window, $parse) which always start with $ sign.

**19186.** How to apply validation in AngularJS?

- *Book id: 4, Type:SQ, editor id: 3, Topic:(23)AngularJS Fundamentals,* *added on: 11-10-2015 05:54 pm, page no: 0    Powered by: Intellect Software Ltd.*+

**Answer:** AngularJS provides you built-in validation directives to validate form client side. This makes your life pretty easy to handle client-side form validations without adding a lot of extra effort. AngularJS form validations are based on the HTML5 form validators.

AngularJS directives for form validation

Here is a list of AngularJS directive which can be applied on an input field to validate its value.

1. <input type="text"
2. ng-model="{ string }"
3. [name="{ string }"]
4. [ng-required="{ boolean }"]
5. [ng-minlength="{ number }"]
6. [ng-maxlength="{ number }"]
7. [ng-pattern="{ string }"]
8. [ng-change="{ string }"]>
9. </input>

**19184.** What is View in AngularJS?

**Answer:** The view is responsible for presenting your models data to end user. Typically it is the HTML markup which exists after AngularJS has parsed and compiled the HTML to include rendered markup and bindings.

**19182.** What is Controller in AngularJS?

**Answer:** The controller defines the actual behavior of your app. It contains business logic for the view and connects the model to view with the help of $scope. A controller is associated with a HTML element with the ng-controller directive.

**19181.** What is ViewModel in AngularJS?

**Answer:** A viewmodel is an object that provides specific data and methods to maintain specific views. Basically, it is a $scope object which lives within your AngularJS app's controller. A viewmodel is associated with a HTML element with the ng-model and ng-bind directives.

**19180.** What is Model in AngularJS?

**Answer:** Models are plain old JavaScript objects that represent data used by your app. Models are also used to represent your app's current state.

**19179.** Does AngularJS support MVC?

**Answer:** AngularJS is a MVC framework. It does not implement MVC in the traditional way, but rather something closer to MVVM Model-View-ViewModel).

**19178.** How AngularJS handle data binding?

**Answer:** AngularJS handle data-binding mechanism with the help of three powerful functions: $watch(), $digest() and $apply(). Most of the time AngularJS will call the $scope.$watch() and $scope.$digest() functions for you, but in some cases you may have to call these functions yourself to update new values.

**19177.** Explain Two-way and One-way data binding in AngularJS?

**Answer:** Two-way data binding - It is used to synchronize the data between model and view. It means, any change in model will update the view and vice versa. ng-model directive is used for two-way data binding.

One-way data binding - This binding is introduced in Angular 1.3. An expression that starts with double colon (::), is considered a one-time expression i.e. one-way binding.

**19176.** What is data binding in AngularJS?

**Answer:** AngularJS data-binding is the most useful feature which saves you from writing boilerplate code (i.e. the sections of code which is included in many places with little or no alteration). Now, developers are not responsible for manually manipulating the DOM elements and attributes to reflect model changes. AngularJS provides two-way data-binding to handle the synchronization of data between model and view.

**19163.** What are ng-repeat special variables?

**Answer:** The ng-repeat directive has a set of special variables which you are useful while iterating the collection. These variables are as follows:

- $index

- $first

- $middle

- $last

1. <html>
2. <head>
3. <script src="lib/angular.js"></script>
4. <script>
5. var app = angular.module("app", []);
6. app.controller("ctrl", function ($scope) {
7. $scope.friends = [
8. { name: 'shailendra', gender: 'boy' },
9. { name: 'kiran', gender: 'girl' },
10. { name: 'deepak', gender: 'boy' },
11. { name: 'pooja', gender: 'girl' }
12. ];
13. });
14. </script>
16. </head>
17. <body ng-app="app">
18. <div ng-controller="ctrl">
19. ul class="example-animate-container">
20. <li ng-repeat="friend in friends">

[{{$index + 1}}] {{friend.name}} is a {{friend.gender}}.

1. <span ng-if="$first">
2. (first element found)
3. </span>
4. <span ng-if="$middle">
5. (middle element found)
6. </span>
7. <span ng-if="$last">
8. (last element found)
9. </span>

1. </li>
2. </ul>
3. </div>
4. </body>
5. </html>

The $index contains the index of the element being iterated. The $first, $middle and $last returns a boolean value depending on whether the current item is the first, middle or last element in the collection being iterated.

**19160.** Explain directives ng-if, ng-switch and ng-repeat?

**Answer:** ng-if – This directive can add / remove HTML elements from the DOM based on an expression. If the expression is true, it add HTML elements to DOM, otherwise HTML elements are removed from the DOM.

1. <div ng-controller="MyCtrl">
2. <div ng-if="data.isVisible">ng-if Visible</div>
3. </div>
5. <script>
6. var app = angular.module("app", []);
7. app.controller("MyCtrl", function ($scope) {
8. $scope.data = {};
9. $scope.data.isVisible = true;
10. });
11. </script>

ng-switch – This directive can add / remove HTML elements from the DOM conditionally based on scope expression.

1. <div ng-controller="MyCtrl">
2. <div ng-switch on="data.case">
3. <div ng-switch-when="1">Shown when case is 1</div>
4. <div ng-switch-when="2">Shown when case is 2</div>
5. <div ng-switch-default>Shown when case is anything else than 1 and 2</div>
6. </div>
7. </div>

10. <script>
11. var app = angular.module("app", []);
12. app.controller("MyCtrl", function ($scope) {
13. $scope.data = {};
14. $scope.data.case = true;
15. });
16. </script>

ng-repeat - This directive is used to iterate over a collection of items and generate HTML from it.

1. <div ng-controller="MyCtrl">
2. <ul>
3. <li ng-repeat="name in names"> {{ name }} </li>
4. </ul>
5. </div>

8. <script>
9. var app = angular.module("app", []);
10. app.controller("MyCtrl", function ($scope) {
11. $scope.names = ['Shailendra', 'Deepak', 'Kamal'];
12. });
13. </script>

**19157.** What directives are used to show and hide HTML elements in AngularJS?

**Answer:** ng-show and ng-hide directives are used to show and hide HTML elements in the AngularJS based on an expression. When the expression is true for ng-show or ng-hide, HTML element(s) are shown or hidden from the page. When the expression is false for ng-show or ng-hide, HTML element(s) are hidden or shown on the page.

1. <div ng-controller="MyCtrl">
2. <div ng-show="data.isShow">ng-show Visible</div>
3. <div ng-hide="data.isHide">ng-hide Invisible</div>
4. </div>
5. <script>
6. var app = angular.module("app", []);
7. app.controller("MyCtrl", function ($scope) {
8. $scope.data = {};
9. $scope.data.isShow = true;
10. $scope.data.isHide = true;
11. });
12. </script>

**19155.** What is $scope and $rootScope?

**Answer:** $scope - A $scope is a JavaScript object which is used for communication between controller and view. Basically, $scope binds a view (DOM element) to the model and functions defined in a controller.

$rootScope - The $rootScope is the top-most scope. An app can have only one $rootScope which will be shared among all the components of an app. Hence it acts like a global variable. All other $scopes are children of the $rootScope.

**19154.** What is scope in AngularJS?

- *Book id: 4, Type:SQ, editor id: 3, Topic:(26)Controllers,* *added on: 11-10-2015 05:09 pm, page no: 0    Powered by: Intellect Software Ltd.*+

**Answer:** Scope is a JavaScript object that refers to the application model. It acts as a context for evaluating angular expressions. Basically, it acts as glue between controller and view.

Controller<------> $Scope <------> View

Scopes are hierarchical in nature and follow the DOM structure of your AngularJS app. AngularJS has two scope objects: $rootScope and $scope.

**19149.** What is auto bootstrap process in AngularJS?

OR

How AngularJS is initialized automatically?

**Answer:** Angular initializes automatically upon DOMContentLoaded event or when the angular.js script is downloaded to the browser and the document.readyState is set to complete. At this point AngularJS looks for the ng-app directive which is the root of angular app compilation and tells about AngularJS part within DOM. When the ng-app directive is found then Angular will:

1. Load the module associated with the directive.

2. Create the application injector.

3. Compile the DOM starting from the ng-app root element.

This process is called auto-bootstrapping.

Example

1. <html>
2. <body ng-app="myApp">
3. <div ng-controller="Ctrl"> Hello {{msg}}! </div>
5. <script src="lib/angular.js"></script>
6. <script>
7. var app = angular.module('myApp', []);
8. app.controller('Ctrl', function ($scope) { $scope.msg = 'World'; });
9. </script>
10. /body>
11. </html>

**19146.** Can you define multiple restrict options on a directive?

**Answer:** You can also specify multiple restrict options to support more than one methods of directive invocation as an element or an attribute. Make sure all are specified in the restrict keyword as:

restrict: 'EA'

**19145.** What is restrict option in directive?

**Answer:** The restrict option in angular directive, is used to specify how a directive will be invoked in your angular app i.e. as an attribute, class, element or comment.

There are four valid options for restrict:

'A' (Attribute)- <span my-directive></span>

'C' (Class)- <span class="my-directive:expression;"></span>

'E' (Element)- <my-directive></my-directive>

'M' (Comment)- <!-- directive: my-directive expression -->

**19142.** What are different ways to invoke a directive?

**Answer:** As an attribute

<span my-directive></span>

As a class

<span class="my-directive: expression;"></span>

As an element

<my-directive></my-directive>

As a comment

<!-- directive: my-directive expression -->

**19140.** How to create custom directives in AngularJS?

**Answer:** You can create your own custom directive by using following syntax:

1. var app = angular.module('app', []);
2. //creating custom directive syntax app.directive("myDir", function () { return { restrict: "E",
3. //define directive type like E = element, A = attribute, C = class, M = comment scope: { //create a new child scope or an isolate scope
4. title: '@' //@ reads the attribute value,//= provides two-way binding, //& works with functions
5. },template: "<div>{{ myName }}</div>",// define HTML markup
6. templateUrl: 'mytemplate.html', //path to the template, used by the directive replace: true | false, // replace original markup with template yes/no
7. transclude: true | false, // copy original HTML content yes/no
8. controller: function (scope) { //define controller, associated with the directive template //TODO: },
9. link: function (scope, element, attrs, controller) {//define function, used for DOM manipulation //TODO:
10. }
11. }
12. });

**19131.** What is the role of ng-app, ng-init and ng-model directives?

**Answer:** The main role of these directives is explained as:

- ng-app - Initialize the angular app.

- ng-init - Initialize the angular app data.

- ng-model - Bind the html elements like input, select, text area to angular app model.

**19130.** What are Directives in AngularJS?

**Answer:** AngularJS directives are a combination of AngularJS template markups (HTML attributes or elements, or CSS classes) and supporting JavaScript code. The JavaScript directive code defines the template data and behaviors of the HTML elements.

AngularJS directives are used to extend the HTML vocabulary i.e. they decorate html elements with new behaviors and help to manipulate html elements attributes in interesting way.

There are some built-in directives provided by AngularJS like as ng-app, ng-controller, ng-repeat, ng-model etc.

**19128.** How AngularJS expressions are different from the JavaScript expressions?

**Answer:** AngularJS expressions are much like JavaScript expressions but they are different from JavaScript expressions in the following ways:

1. Angular expressions can be added inside the HTML templates.

2. Angular expressions doesn't support control flow statements (conditionals, loops, or exceptions).

3. Angular expressions support filters to format data before displaying it.

**19126.** What are Expressions in AngularJS?

**Answer:** AngularJS expressions are much like JavaScript expressions, placed inside HTML templates by using double braces such as: {{expression}}. AngularJS evaluates expressions and then dynamically adds the result to a web page. Like JavaScript expressions, they can contain literals, operators, and variables.

There are some valid AngularJS expressions:

- {{ 1 + 2 }}

- {{ x + y }}

- {{ x == y }}

- {{ x = 2 }}

- {{ user.Id }}

**19125.** What are Filters in AngularJS?

**Answer:** Filters are used to format data before displaying it to the user. They can be used in view templates, controllers, services and directives. There are some built-in filters provided by AngularJS like as Currency, Date, Number, OrderBy, Lowercase, Uppercase etc. You can also create your own filters.

Filter Syntax

{{ expression | filter}}

Filter Example

1. <script type="text/javascript">
2. { { 14 | currency } } //returns $14.00
3. </script>

**19123.** What is Angular Prefixes $ and $$?

**Answer:** To prevent accidental name collisions with your code, Angular prefixes names of public objects with $ and names of private objects with $$. So, do not use the $ or $$ prefix in your code.

**19122.** What is Global API?

**Answer:** Global API provides you global functions to perform common JavaScript tasks such as comparing objects, deep copying, iterating through objects, and converting JSON data etc. All global functions can be accessed by using the angular object. The list of global functions is given below:

angular.lowercase - Converts the specified string to lowercase.

angular.uppercase - Converts the specified string to uppercase.

angular.forEach - Invokes the iterator function once for each item in obj collection, which can be either an object or an array.

angular.isUndefined - Determines if a reference is undefined.

angular.isDefined - Determines if a reference is defined.

angular.isObject - Determines if a reference is an Object.

angular.isString - Determines if a reference is a String.

angular.isNumber - Determines if a reference is a Number.

angular.isDate - Determines if a value is a date.

angular.isArray - Determines if a reference is an Array.

angular.isFunction - Determines if a reference is a Function.

angular.isElement - Determines if a reference is a DOM element (or wrapped jQuery element).

angular.copy - Creates a deep copy of source, which should be an object or an array.

angular.equals - Determines if two objects or two values are equivalent. Supports value types, regular expressions, arrays and objects.

angular.bind - Returns a function which calls function fn bound to self

angular.toJson - Serializes input into a JSON-formatted string. Properties with leading $$ characters will be stripped since angular uses this notation internally.

angular.fromJson - Deserializes a JSON string.

angular.bootstrap - Use this function to manually start up angular application.

angular.reloadWithDebugInfo - Use this function to reload the current application with debug information turned on.

angular.injector - Creates an injector object that can be used for retrieving services as well as for dependency injection

angular.element - Wraps a raw DOM element or HTML string as a jQuery element.

angular.module - Used for creating, registering and retrieving Angular modules.

**19116.** When dependent modules of a module are loaded?

**Answer:** A module might have dependencies on other modules. The dependent modules are loaded by angular before the requiring module is loaded.

In other words the configuration blocks of the dependent modules execute before the configuration blocks of the requiring module. The same is true for the run blocks. Each module can only be loaded once, even if multiple other modules require it.

**19115.** What is difference between config() and run() method in AngularJS?

**Answer:** Configuration block – This block is executed during the provider registration and configuration phase. Only providers and constants can be injected into configuration blocks. This block is used to inject module wise configuration settings to prevent accidental instantiation of services before they have been fully configured. This block is created using config() method.

1. angular.module('myModule', []).config(function (injectables) {
2. // provider-injector
3. // This is an example of config block.
4. // You can have as many of these as you want.
5. // You can only inject Providers (not instances)
6. // into config blocks.
7. }).run(function (injectables) {
8. // instance-injector
9. // This is an example of a run block.
10. // You can have as many of these as you want.
11. // You can only inject instances (not Providers)
12. // into run blocks
13. });

Run block – This block is executed after the configuration block. It is used to inject instances and constants. This block is created using run() method. This method is like as main method in C or C++.

The run block is a great place to put event handlers that need to be executed at the root level for the application. For example, authentication handlers.

**19110.** How angular modules load the dependencies?

**Answer:** An angular module use configuration and run blocks to inject dependencies (like providers, services and constants) which get applied to the angular app during the bootstrap process.

**19109.** What is core module in AngularJS?

**Answer:** ng is the core module in angular. This module is loaded by default when an angular app is started. This module provides the essential components for your angular app like directives, services/factories, filters, global APIs and testing components.

**19108.** What components can be defined within AngularJS modules?

**Answer:** You can define following components with in your angular module:

1. Directive

2. Filter

3. Controller

4. Factory

5. Service

6. Provider

7. Value

8. Config settings and Routes

**19106.** What are Modules in AngularJS?

**Answer:** AngularJS modules are containers just like namespace in C#. They divide an angular app into small, reusable and functional components which can be integrated with other angular app. Each module is identified by a unique name and can be dependent on other modules. In AngularJS, every web page (view) can have a single module assigned to it via ng-app directive.

Creating an AngularJS module

1. <script type="text/javascript">
2. //defining module
3. angular.module('myApp', []);
4. //OR defining module which has dependency on other modules
5. angular.module('myApp', ['dependentModule1', 'dependentModule2']);
6. </script>

Using an AngularJS module into your app

You can bootstrap your app by using your AngularJS module as given below:

1. <html ng-app="myApp">
2. <head> ... </head>
3. <body> ... </body>
4. </html>

**19100.** How AngularJS handle the security?

**Answer:** AngularJS provide following built-in protection from basic security holes:

1. Prevent HTML injection attacks.

2. Prevent Cross-Site-Scripting (CSS) attacks.

3. Prevent XSRF protection for server side communication.

Also, AngularJS is designed to be compatible with other security measures like Content Security Policy (CSP), HTTPS (SSL/TLS) and server-side authentication and authorization that greatly reduce the possible attacks.

**19098.** What are AngularJS features?

**Answer:** The features of AngularJS are listed below:

1. Modules

2. Directives

3. Templates

4. Scope

5. Expressions

6. Data Binding

7. MVC (Model, View & Controller)

8. Validations

9. Filters

10. Services

11. Routing

12. Dependency Injection

13. Testing

**19097.** What is the size of angular.js file?

**Answer:** The size of the compressed and minified file is < 36KB.

**19096.** What browsers AngularJS support?

**Answer:** The latest version of AngularJS 1.3 support Safari, Chrome, Firefox, Opera 15+, IE9+ and mobile browsers (Android, Chrome Mobile, iOS Safari, Opera Mobile).

AngularJS 1.3 has dropped support for IE8 but AngularJS 1.2 will continue to support IE8.

**19095.** Is AngularJS a library, framework, plugin or a browser extension?

**Answer:** AngularJS is a first class JavaScript framework which allows you to build well structured, easily testable, and maintainable front-end applications. It is not a library since library provides you limited functionality or has dependencies to other libraries.

It is not a plugin or browser extension since it is based on JavaScript and compatible with both desktop and mobile browsers.

**19093.** How to use jQuery with AngularJS?

**Answer:** By default AngularJS use jQLite which is the subset of jQuery. If you want to use jQuery then simply load the jQuery library before loading the AngularJS. By doing so, Angular will skip jQLite and will started to use jQuery library.

**19092.** Does AngularJS has dependency on jQuery?

**Answer:** AngularJS has no dependency on jQuery library. But it can be used with jQuery library.

**19091.** How AngularJS is different from other JavaScript Framework?

**Answer:** Today, AngularJS is the most popular and dominant JavaScript framework for professional web development. It is well suited for small, large and any sized web app and web application.

AngularJS is different from other JavaScript framework in following ways:

1. AngularJS mark-up lives in the DOM.

2. AngularJS uses plain old JavaScript objects (POJO).

3. AngularJS is leverages with Dependency Injection.

**19090.** What are the advantages of AngularJS?

**Answer:** There are following advantages of AngularJS:

- Data Binding - AngularJS provides a powerful data binding mechanism to bind data to HTML elements by using scope.

- Customize & Extensible - AngularJS is customized and extensible as per you requirement. You can create your own custom components like directives, services etc.

-Code Reusability - AngularJS allows you to write code which can be reused. For example custom directive which you can reuse.

- Support – AngularJS is mature community to help you. It has widely support over the internet. Also, AngularJS is supported by Google which gives it an advantage.

- Compatibility - AngularJS is based on JavaScript which makes it easier to integrate with any other JavaScript library and runnable on browsers like IE, Opera, FF, Safari, Chrome etc.

- Testing - AngularJS is designed to be testable so that you can test your AngularJS app components as easy as possible. It has dependency injection at its core, which makes it easy to test.

**19089.** Why to use AngularJS?

**Answer:** There are following reasons to choose AngularJS as a web development framework:

1. It is based on MVC pattern which helps you to organize your web apps or web application properly.

2. It extends HTML by attaching directives to your HTML markup with new attributes or tags and expressions in order to define very powerful templates.

3. It also allows you to create your own directives, making reusable components that fill your needs and abstract your DOM manipulation logic.

4. It supports two-way data binding i.e. connects your HTML (views) to your JavaScript objects (models) seamlessly. In this way any change in model will update the view and vice versa without any DOM manipulation or event handling.

5. It encapsulates the behaviour of your application in controllers which are instantiated with the help of dependency injection.

6. It supports services that can be injected into your controllers to use some utility code to fulfil your need. For example, it provides $http service to communicate with REST service.

7. It supports dependency injection which helps you to test your angular app code very easily.

8. Also, AngularJS is mature community to help you. It has widely support over the internet.

**19088.** What is AngularJS?

**Answer:** AngularJS is an open-source JavaScript framework developed by Google. It helps you to create single-page applications or one-page web applications that only require HTML, CSS, and JavaScript on the client side. It is based on MV-\* pattern and allow you to build well structured, easily testable, and maintainable front-end applications.