**Q1. What is AngularJS?**

Ans. 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

**Q3. Why this project is called "AngularJS"?**

Ans. Html has angle brackets i.e.<,> and ng sound like Angular. That’s why it is called AngularJS.

**Q4. What are the advantages of AngularJS?**

Ans. 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.

**Q6. What IDEs you can use for AngularJS development?**

Ans. AngularJS development can be done with the help of following IDEs:

1. Visual Studio 2012, 2013, 2015 or higher

2. Eclipse

3. WebStorm

4. Sublime Text

5. TextMate

**Q14. What is the size of angular.js file?**

Ans. The size of the compressed and minified file is < 36KB.

**Q15. What are AngularJS features?**

Ans. 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

**Q16. How AngularJS handle the security?**

Ans. 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

**Q17. What are Modules in AngularJS?**

Ans. 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**

<script type="text/javascript">

// defining module

**angular.module('myApp', []);**

//OR defining module which has dependency on other modules

**angular.module('myApp', ['dependentModule1', 'dependentModule2']);**

</script>

**Using an AngularJS module into your app**

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

<html **ng-app="myApp">**

<head>

...

</head>

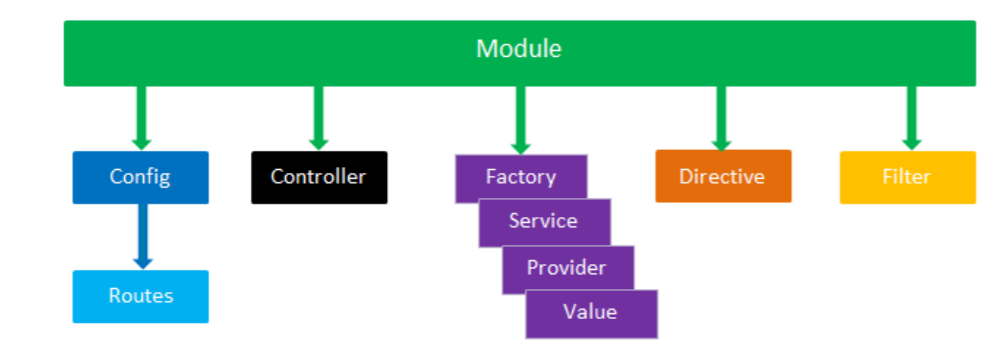
<body>

...

</body>

**Q18. What components can be defined within AngularJS modules?**

Ans. 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

**Q19. What is core module in AngularJS?**

Ans. ng is the **core module** in angular.

This module is **loaded by default when an angular app is started**. T

his module **provides the essential components** for your angular app like **directives, services/factories, filters, global APIs and testing components.**

**Q20. How angular modules load the dependencies?**

Ans. 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**.

**Q21. What is difference between config() and run() method in AngularJS?**

**Ans. 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**.

angular.module('myModule', []).

config(function (injectables) { // provider-injector

// This is an example of config block.

// You can have as many of these as you want.

// You can only inject Providers (not instances)

// into config blocks.

}).

run(function (injectables) { // instance-injector

// This is an example of a run block.

// You can have as many of these as you want.

// You can only inject instances (not Providers)

// into run blocks

});

**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**

**Q24. What is Angular Prefixes $ and $$?**

Ans. 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.

**Q25. What are Filters in AngularJS?**

Ans. 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**

<script type="text/javascript">

**{ { 14 | currency } } //returns $14.00**

</script>

**Q26. What are Expressions in AngularJS?**

Ans. 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 }}

**Q27. How AngularJS expressions are different from the JavaScript expressions?**

Ans. 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**

**Q28. What are Directives in AngularJS?**

Ans. 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.

**Q29. What is the role of ng-app, ng-init and ng-model directives?**

Ans. 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.**

**Q30. How to create custom directives in AngularJS?**

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

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

**//creating custom directive syntax**

**app.directive("myDir", function () {**

**return {**

**restrict: "E", //define directive type like E = element, A = attribute, C =**

**class, M = comment**

**scope: { //create a new child scope or an isolate scope**

**title: '@' //@ reads the attribute value,**

**//= provides two-way binding,**

**//& works with functions**

**},**

**template: "<div>{{ myName }}</div>",// define HTML markup**

**templateUrl: 'mytemplate.html', //path to the template, used by the directive**

**replace: true | false, // replace original markup with template yes/no**

**transclude: true | false, // copy original HTML content yes/no**

**controller: function (scope) { //define controller, associated with the directive**

**template**

**//TODO:**

**},**

**link: function (scope, element, attrs, controller) {//define function, used for**

**DOM manipulation**

**//TODO:**

**}**

**}**

**});**