1. What are the differences between AngularJS (angular 1.x) and Angular (Angular 2.x and beyond)?
2. Angular 2 is not an upgrade of Angular 1 but it is completely rewritten.
3. Angular 2 uses TypeScript which is a superset of JavaScript (It doesn’t mean only TypeScript but dart also).
4. Angular 1.x was not built with mobile support in mind, where Angular 2 is mobile-oriented.
5. Angular 1's core concept was $scope but you will not find $scope in Angular 2.0 and above.
6. Angular 1.x Controllers are gone in Angular v2. We can say that Controllers are replaced with “Components” in Angular 2.
7. In Angular 2, Structural directives syntax is changed. ng-repeat is replaced with \*ngFor.
8. In Angular 2, local variables are defined using hash(#) prefix.
9. Two-way data binding: ng-model has been replaced with [(ngModel)]
10. Angular 2 uses Hierarchical Dependency Injection system which is the major performance booster of it.
11. Angular 2 implements unidirectional-tree based change detection which, again, increases the performance.
12. If you compare the file size, Angular 2 is 20 kb less than Angular 1 which helps in decreasing the load time for apps.
13. Angular 2 provides more choice for languages. You can use any of the languages from ES5, ES6, TypeScript or Dart to write Angular 2 code while Angular 1.x had ES5, ES6, and Dart only. Addition of TypeScript is a great step as TypeScript is an awesome way to write JavaScript.
14. To filter output in our templates in Angular 1.x, we used the pipe character (|) and one or more filters. Now, in Angular 2, they are called pipes. The syntax remains same.
15. Angular 2 uses camelCase syntax for built-in directives. For example, ng-class is now ngClass and ng-model is now ngModel.
16. One of the biggest advantages of Angular is Dependency Injection. In Angular 2, DI is there but now there is a different way to inject dependencies. As everything is a class in Angular, so DI is achieved via a constructor.

In Angular 1.x, we can define a service via 5 different ways.

Factory

Service

Provider

Constant

Values

2)What is a component? Why would you use it?

**Component** is a special kind of directive that uses a simpler configuration which is suitable for a **component**-based **application** structure. This makes it easier **to** write an app in a way that's similar **to using** Web **Components** or **using** the new **Angular's** style of **application** architecture.

3)What is the minimum definition of a component?

The absolute minimal configuration for a @Component in angular is a template. Both template properties are set to optional because we have to define either template or templateUrl

4)What is a module, and what does it contain?

**Module** in Angular refers to a place where you can group the components, directives, pipes, and services, which are related to the application.

In case you are developing a website, the header, footer, left, center and the right section become part of a module.

To define module, we can use the **Ng Module**. When you create a new project using the Angular –cli command, the ng module is created in the app module.

5) What is a service, and when will you use it?

**Angular services are** singleton objects which get instantiated only once during the lifetime of an application. ... The main objective of a **service** is **to** organize and share business logic, models, or data and functions with different components of an **Angular** application

6)What is a promise? Explain it laymen's terms?

A **Promise** is an object representing the eventual completion or failure of an asynchronous operation. Essentially, a **promise** is a returned object you attach callbacks to, instead of passing callbacks into a function

7) What are the lifecycle hooks for components and directives?

**Directive** and **component** instances have a **lifecycle** as Angular creates, updates, and destroys them. Developers can tap into key moments in that **lifecycle** by implementing one or more of the **lifecycle hook** interfaces in the Angular core library. No **directive** or **component** will implement all of the **lifecycle hooks**.

1. What are pipes? Give me an example.

In Angular 2, Pipes are mainly used to change the data display format.

By using the Pipe operator (|), we can apply the Pipe's features to any of the property in our Angular project.

In addition to that, we can also chain pipe and pass parameters to the Pipe.

Angular 2 provides many built-in Pipes which include uppercase, lowercase, decimal, date, percent, currency etc.

They are all available for use in any template in our Angular 2 project.

Example:

<!DOCTYPE html>

<html>

<head>

<title></title>

<meta charset="utf-8" />

</head>

<body>

<table>

<thead>

<tr>

<th>Code</th>

<th>Name</th>

<th>Gender</th>

<th>Annual Salary</th>

<th>Date of Birth</th>

</tr>

</thead>

<tbody>

<tr \*ngFor='let employee of employees'>

<td>{{employee.code}}</td>

<td>{{employee.name|uppercase}}</td>

<td>{{employee.gender}}</td>

<td>{{employee.annualSalary}}</td>

<td>{{employee.dateOfBirth}}</td>

</tr>

<tr \*ngIf="!employees || employees.length==0">

<td colspan="5">

No employees to display

</td>

</tr>

</tbody>

</table>

</body>

</html>

9)What are the differences between reactive forms and template driven forms?

Template-driven forms use the FormsModule , whilereactive forms use the ReactiveFormsModule .Template-driven forms are asynchronous, whilereactive forms are synchronous.

10)What is a dumb, or presentation, component? What are the benefits of using dumb components?

Dumb Components. Dumb components are also called 'presentational' componentsbecause their only responsibility is to present something to the DOM. Once that is done, the component is done with it. No keeping tabs on it, no checking in once in a while to see how things are going.

11)When does a lazy loaded module is loaded?

Lazy Loading generally, is a concept where we delay loading of an object until it is needed. In Angular, all the JavaScript components declared in the declarations array app.module.ts arebundled and loaded in one fell swoop when a user visits our site.

12)Why angular uses url segment?

A UrlSegment is a part of a URL between the two slashes. It contains a path and the matrix parameters associated with the segment.

Example:

class [UrlSegment](https://angular.io/api/router/UrlSegment)

{

[constructor(path: string, parameters: { [name: string]: string; })](https://angular.io/api/router/UrlSegment#constructor())

[path: string](https://angular.io/api/router/UrlSegment#path)

[parameters: {...}](https://angular.io/api/router/UrlSegment#parameters)

[parameterMap](https://angular.io/api/router/UrlSegment#parameterMap)

[toString(): string](https://angular.io/api/router/UrlSegment#toString)

}