Angular interview

1. How does angular application works?

Ans: Every Angular app consists of a file named **angular.json**. This file consists of configuration of the app. Builder looks to this file and find the entry point of the the application. Default main.ts is defined.

In main.ts file create browser enrnment along with this called function bootstrap module which bootstraps the application. For ex:

Import {platformBrowserDynamic} from ‘@angular/platform-browse-dynamic,

platformBrowserDynamic.bootstrapModule(AppModule)

AppModule is declared inside the app.module.ts file That file contains all declaration.In this file App Component is bootstrapped, this file interacts the webpage and serves the data. Inside this file selector which is used to access the component , templateurl contains html and style contains css. After that angular calls the index file, the file call the roots component which is defined inside the index file.

1. Advantage of angular over other frameworks?

Ans: Number of built in features like: routing, state management, rxjs lib, httpservice

Typescript: Typescript is object oriented complied language.

Ease to find the errors using typescript

Javascript is not strongly typed check but typescript comes with an optional static type check.

Typescript supports object oriented concepts like classes interfaces etc.

Typescript compiler convert its code to javascript equivalent.

Angular Material.

Pojo model: plain old javascript object to make code handy and independent.This model enable us to to keep our code clean and less code requires also with this application loads quickly.

1. Difference between Angularjs and angular

Architecture: In Angularjs MVVM and in angular controllers replaces with components

Language: Javascript and in angular typescript.

Mobile Support

Structure: In angular js maintain code is difficult and in angular it is easy to maintain

Property binding: []

Event binding: ()

Q: Aot Compilation ?

Angular app constis of templates and components which browser cannot understand. All the application needs to be complied first before running inside the browser.

1. JIT
2. AOT

JIT: Develpment ->Production->App downloaded to browser->Angular parses and compiles the code into browser.

AOT: Development-> Angular parses and compiles the code into browser->Production->App download into browser.

In JIT compliation the application compiles inside the browser during runtime.

In Aot application complies during the build time.

With AOT application render fast and detect errors during build time.

Aot complier adds htmls and templates into js file before they run into the browsers..Due to this no extra html files to be read.

Q: Components, Modules and Services in Angular.

Ans: Building blocks which controls a part of the UI application.

A component is defined using the @component Decorator.Component having three parts:

Templates, Stylesheet, class which contains the business logic.

Modules: module in angular refers to place where you can group the components, directives, pipes and services which are related to application.

To define module we use ngmodule

Services: services are object which initialized only once during the application. The main objective of the service is to share the data functions with different component of an angular application. A service is defiend using @injectable decorator. Any function inside service can be used by any component just importing the service.

Q: Lifecycles of angular..

Ans: ngonchanges: This method is called before ngoninit and when one or more input properties of the component changes. This method receive simplechanges object which contain previous and current value of the property.

Ngoninit: This is called after ngonchanges hook. This event is called when component is initialized and sets the input properties of the component.

Ngdocheck:  It gets called after **ngOnChanges** and **ngOnInit** and is used to detect and act on changes that cannot be detected by Angular. We can implement our change detection algorithm in this hook.

**ngAfterViewInit:** It responds after a component's view, or a child component's view is initialized.

**ngOnDestroy( )** It gets called just before Angular destroys the component. This hook can be used to clean up the code and detach event handlers.

String Interpolation and property binding:

String interpolation and property binding are parts of **data-binding** in Angular.

String interpolation uses the double curly braces **{{ }}** to display data from the component.  Property binding uses the square brackets **[ ]** syntax.

Data-binding can be done in two ways, **one-way** binding and **two-way** binding.

Angular support both one way and two way data binding.

One way databinding is a simple one way communication where HTML template is changed when we make changes in TypeScript code.

We have seen that in one-way data binding any change in the template (view) were not be reflected in the component TypeScript code. To resolve this problem, Angular provides two-way data binding. The two-way binding has a feature to update data from component to view and vice-versa.

In two way data binding, property binding and event binding are combined together.

[(ngModel)]

Q: Difference between angular and javascript expression?

the biggest difference is that Angular expressions allow us to write JavaScript in HTML which is not the case when it comes to JavaScript expressions.

Angular have pipes while javascript does not have pipes

Q: Observable different from promises?

The Observable isn’t an Angular specific feature, but a new standard for managing async data that will be included in the ES7 release

Lazy. An observable is not called until we subscribe to the observable

Can be cancelled by using the unsubscribe() method

Observable provides operators like map, forEach, filter, reduce, retry, retryWhen etc.

There are also powerful operators like retry(),  reply(), retryWhen(), delay().

Emits multiple values over a period of time

The next difference is that Promises are always **asynchronous**. Even when the promise is immediately resolved. Whereas an Observable, can be both **synchronous** and **asynchronous**.

Promises:

Emits a single value

Not Lazy

Cannot be cancelled

Q: Directives in Angular ?

A directive is a class in Angular that is declared with a **@Directive** decorator.

Every directive has its own behaviour and can be imported into various components of an application.

**When to use a directive?**

Consider an application, where multiple components need to have similar functionalities. The norm thing to do is by adding this functionality individually to every component but, this task is tedious to perform. In such a situation, one can create a **directive** having the required functionality and then, import the directive to components which require this functionality.

**Types of directives:**

**Component directives**

A @Component decorator is actually a @Directive decorator extended with template-oriented features. Whenever Angular renders a directive, it changes the DOM according to the instructions given by the directive. Directive appears within an element tag similar to attributes.

The Angular Directive can be classified into two types: ***structural***and ***attribute***directives.

**Structural directives:**

Structural directives start with a \* sign. These directives are used to manipulate and change the structure of the DOM elements. For example, \*ngIf directive, \*ngSwitch directive, and \*ngFor directive.

**Attribute Directives:** Attribute directives are used to change the look and behavior of the DOM elements. For example: ngClass directive, and ngStyle directive etc.

**ngClass Directive:**

**ngStyle Directive**

**Share Data between components?**

**Parent-child: get using** @Input() data:string

In the child component, we are using @Input decorator to capture data coming from a parent component and using it inside the child component’s template.

**Child to parent using @ViewChild decorator**

**@ViewChild** decorator is used to reference the child component as “child” property.

**Child to parent using @Output and EventEmitter**

**Receive data using**  <app-child (dataEvent)="receiveData($event)"></app-child>

Dependency Injection:

Dependency Injection (DI) is a core concept of Angular 2+ and allows a class receive dependencies from another class. Most of the time in Angular, dependency injection is done by injecting a service class into a component or module class.

@Injectable()

**View Child:**

ViewChild and ContentChild are two very important features of Angular. It is used to access Child Component in the Parent Component.

@ViewChild and @ViewChildren are the types of decorators used to access the child component class and its different properties into the parent component. It's similar to the inheritance.

View child also used to get the element properties like <div id=”a”>

View child is access only single property while viewchildren get multiple properties using querylist.

Change Detection: Change Detection means updating the DOM whenever data is changed. Angular provides two strategies for Change Detection.

In its default strategy, whenever any data is mutated or changed, Angular will run the change detector to update the DOM.

In the onPush strategy, Angular will only run the change detector  when a new reference is passed to @Input() data.

What is digest cycle?

Digest cycle is the process of monitoring watchlist to track the changes in the value of the watch variable. The digest cycle is implicitly triggered, but we can also trigger it manually using $apply() function.

Q: What is pipe?

Pipe (|) is used to transform input data into desired format. For example,

**What are the key components of Angular?**

**Component:** These are the basic building blocks of angular application to control HTML views.

**Modules:** An angular module is set of angular basic building blocks like component, directives, services etc. An application is divided into logical pieces and each piece of code is called as "module" which perform a single task.

**Templates:** This represent the views of an Angular application.

**Metadata:** Metadata is used to decorate the class so that it can configure the expected behavior of a class. Decorators are the core concept when developing with Angular (versions 2 and above). The user can use metadata to a class to tell Angular app that AppComponent is the component. Metadata can be attached to the TypeScript using the decorator.

### What are the differences between Component and Directive?

Basically there are three types of directives in Angular2 according to documentation.

* Component
* Structural directives
* Attribute directives

# Component

It is also a type of directive with template,styles and logic part which is most famous type of directive among all in Angular2. In this type of directive you can use other directives whether it is custom or builtin in the @Component annotation like following:

@View decorator or templateurl/template are mandatory

Directive doesn't use View

Directive is use to design re-usable components

### What is a template?

A template is a HTML view where you can display data by binding controls to properties of an Angular component. You can store your component's template in one of two places. You can define it inline using the template property, or you can define the template in a separate HTML file and link to it in the component metadata using the @Component decorator's templateUrl property.

### What is a module?

Modules are logical boundaries in your application and the application is divided into separate modules to separate the functionality of your application. Lets take an example of **app.module.ts** root module declared with **@NgModule** decorator as below,

import { NgModule } from '@angular/core';

import { BrowserModule } from '@angular/platform-browser';

import { AppComponent } from './app.component';

@NgModule ({

imports: [ BrowserModule ],

declarations: [ AppComponent ],

bootstrap: [ AppComponent ],

providers: []

})

export class AppModule { }

The NgModule decorator has five important(among all) options

1. The imports option is used to import other dependent modules. The BrowserModule is required by default for any web based angular application
2. The declarations option is used to define components in the respective module
3. The bootstrap option tells Angular which Component to bootstrap in the application
4. The providers option is used to configure set of injectable objects that are available in the injector of this module.
5. The entryComponents option is a set of components dynamically loaded into the view.

## **@Inject()**

is a manual mechanism for letting Angular know that a parameter must be injected.

## **@Injectable()**

lets Angular know that a class can be used with the dependency injector.

**Providers** tell the injector how to create the service. Without a provider, the injector would not know that it is responsible for injecting the service nor be able to create the service. Usually, providers are mentioned in the module or component metadata. For example, if a component want to call the service 'FileWriter', the component should mention in the metadata, that this service should be created and injected by the injector.

### What is the difference between constructor and ngOnInit?

**ngOnInit**

One of the Angular life cycle hook method

Called by Angular

Actual business logic performed here

Angular calls ngOnInit when it finishes creating a component DOM

**Constructor**

Typescript feature nothing to do with Angular

Called by Javascript Engine

Used for Injecting dependencies

Constructor is automaticlly called at the time of creating object of the class

### What is the purpose of async pipe?

The async pipe in angular will subscribe to an Observable or Promise and return the latest value it has emitted. Whenever a new value is emitted from an Observable or Promise, the async pipe marks the component to be checked for changes. When the component gets destroyed, the async pipe unsubscribes automatically to avoid potential memory leaks.

## Subject.next()

The subject next method is used to send messages to an observable which are then sent to all angular components that are subscribers (a.k.a. observers) of that observable.

### What happens if you use script tag inside template?

Angular recognizes the value as unsafe and automatically sanitizes it, which removes the **<script>** tag but keeps safe content such as the text content of the <script> tag. This way it eliminates the risk of script injection attacks. If you still use it then it will be ignored and a warning appears in the browser console.

### What are template statements?

A template statement responds to an event raised by a binding target such as an element, component, or directive. The template statements appear in quotes to the right of the = symbol like **(event)="statement"**.

Let's take an example of button click event's statement

<button (click)="editProfile()">Edit Profile</button>

In the above expression, editProfile is a template statement.

### What are pipes?

A pipe takes in data as input and transforms it to a desired output.

### What is a parameterized pipe?

In Angular, we can pass any number of parameters to the pipe using a colon (:) and when we do so, it is called Angular Parameterized Pipes.

### What is a custom pipe?

Apart from built-inn pipes, you can write your own custom pipe with the below key characteristics,

1. A pipe is a class decorated with pipe metadata **@Pipe** decorator, which you import from the core Angular library For example,
2. @Pipe({name: 'customFileSizePipe'})
3. export class FileSizePipe implements PipeTransform {
4. transform(size: number, extension: string = 'MB'): string {
5. return (size / (1024 \* 1024)).toFixed(2) + extension;
6. }
7. }

### What is a bootstrapping module?

Every application has at least one Angular module, the root module that you bootstrap to launch the application is called as bootstrapping module. It is commonly known as AppModule

### What is the difference between pure and impure pipe?

A pure pipe is only called when Angular detects a change in the value or the parameters passed to a pipe. For example, any changes to a primitive input value (String, Number, Boolean, Symbol) or a changed object reference (Date, Array, Function, Object). An impure pipe is called for every change detection cycle no matter whether the value or parameters changes. i.e, An impure pipe is called often, as often as every keystroke or mouse-move.

pure: false

### What is HttpClient and its benefits?

Most of the Front-end applications communicate with backend services over HTTP protocol using either XMLHttpRequest interface or the fetch() API. Angular provides a simplified client HTTP API known as **HttpClient** which is based on top of XMLHttpRequest interface. This client is avaialble from @angular/common/http package. You can import in your root module as below,

import { HttpClientModule } from '@angular/common/http';

The major advantages of HttpClient can be listed as below,

1. Contains testability features
2. Provides typed request and response objects
3. Intercept request and response
4. Supports Observalbe APIs
5. Supports streamlined error handling

### How can you read full response?

The response body doesn't may not return full response data because sometimes servers also return special headers or status code which which are important for the application workflow. Inorder to get full response, you should use observe option from HttpClient,

getUserResponse(): Observable<HttpResponse<User>> {

return this.http.get<User>(

this.userUrl, { observe: 'response' });

}

Now HttpClient.get() method returns an Observable of typed HttpResponse rather than just the JSON data.

### What is RxJS?

### The full form of RxJS is Reactive Extension for Javascript. It is a javascript library that uses observables to work with reactive programming that deals with asynchronous data calls, callbacks and event-based programs.

### (map, filter, reduce, every, etc.) to allow handling asynchronous events as collections.

## **When to use RxJS?**

If your project consists of lots of async task handling than RxJS is a good choice. It is loaded by default with the Angular project.

### Reactive Programming Is Programming With Asynchronous Data Streams.

### What is subscribing?

An Observable instance begins publishing values only when someone subscribes to it. So you need to subscribe by calling the **subscribe()** method of the instance, passing an observer object to receive the notifications.

### What is an observer?

### Observers are simply a set of callbacks, one for each type of notification delivered by the Observable: next, error, and complete.

### What is multicasting?

### Multi-casting is the practice of broadcasting to a list of multiple subscribers in a single execution.

### What are the utility functions provided by RxJS?

* **map()** : Used to map values of different data types
* **filter()** : Used for filtering streams
* **concat()** : Used to concatenate multiple strings
* **merge()**: Used to recursively descend into object properties in the source copy, while forming a deep copy of the same.

## What is Angular Elements?

The Angular Elements package (@angular/elements) allows you to create native custom elements and author web components with Angular.

The @angular/elements package provides a createCustomElement() API that can be used to transform Angular Components to native Custom Elements.

### Angular Elements hosts an Angular component, providing a bridge between the data and logic defined in the component and standard DOM APIs, thus, providing a way to use Angular components in non-Angular environments.

### What are dynamic components?

### Dynamic components are the components in which components location in the application is not defined at build time.i.e, They are not used in any angular template. But the component is instantiated and placed in the application at runtime.

ViewContainerRef

componentFactoryResolver

If you want to insert new component or template, you need to tell Angular where to put this element.

And that’s what ViewContainerRef is:

### simple registry that maps Components to generated [ComponentFactory](https://angular.io/api/core/ComponentFactory) classes that can be used to create instances of components. Use to obtain the factory for a given component type, then use the factory's create() method to create a component of that type.

### What is Angular Router?

### Angular Router is a mechanism in which navigation happens from one view to the next as users perform application tasks. It borrows the concepts or model of browser's application navigation.

### What is router outlet?

The RouterOutlet is a directive from the router library and it acts as a placeholder that marks the spot in the template where the router should display the components for that outlet. Router outlet is used like a component,

### What are active router links?

RouterLinkActive is a directive that toggles css classes for active RouterLink bindings based on the current RouterState.

### What is router state?

RouterState is a tree of activated routes. Every node in this tree knows about the "consumed" URL segments, the extracted parameters, and the resolved data.

### What are router events?

During each navigation, the Router emits navigation events through the Router.events property allowing you to track the lifecycle of the route.

### What is activated route?

ActivatedRoute contains the information about a route associated with a component loaded in an outlet.

### What is the purpose of Wildcard route?

If the URL doesn't match any predefined routes then it causes the router to throw an error and crash the app. In this case, you can use wildcard route.

{ path: '\*\*', component: PageNotFoundComponent }

**Why do we need compilation process?**

The Angular components and templates cannot be understood by the browser directly. Due to that Angular applications require a compilation process before they can run in a browser. For example, In AOT compilation, both Angular HTML and TypeScript code converted into efficient JavaScript code during the build phase before browser runs it.

### What are the two phases of AOT?

The AOT compiler works in three phases,

1. **Code Analysis:** The compiler records a representation of the source
2. **Code generation:** It handles the interpretation as well as places restrictions on what it interprets.
3. **Validation:** In this phase, the Angular template compiler uses the TypeScript compiler to validate the binding expressions in templates.

### How do you specify angular template compiler options?

The angular template compiler options are specified as members of the **angularCompilerOptions** object in the tsconfig.json file. These options will be specified adjecent to typescript compiler options.

### How do you enable binding expression validation?

You can enable binding expression validation explicitly by adding the compiler option **fullTemplateTypeCheck** in the "angularCompilerOptions" of the project's tsconfig.js

$**any**() **type cast**

You can switch to the latest version **Angular** 9. Sometimes a binding expression will be reported as a **type** error and it is not possible or difficult to fully specify the **type**. To silence the error, you can use the $**any cast function** to **cast** the expression to the **any type**.

**Type narrowing** gives TypeScript more information about the **type** being **narrowed**, and allows the TypeScript compiler to be more helpful to you. When used responsibly, **narrowing types** can make your code easier to understand.

#### **@HostListener() Decorator and @HostBinding() Decorator**

#### @HostBinding and @HostListener are two decorators in Angular that can be really useful in custom directives.

**HostListener** lets you listen for events on the host element or component.

**@HostBinding** lets you set properties on the element or component that hosts the directive

### What is zone?

Angular's change detection is heavily dependent on Zones, **Zone.js** is the key to Angular's change detection.

Angular runs the code of your components within its zone where most async APIs (addEventListener, setTimeout, ...) are patched so the zone can notify Angular when such an async callback has happend. This is when Angular runs change detection.

This would require knowledge about the inner workings of the AssetService you're using.

If you initialized AssetService outside Angular or AssetService by other means executes code outside Angulars zone, then Angular doesn't get notified about happened async callbacks and doesn't run change detection.

With zone.run(...) you explicitely make code execute inside Angulars zone and change detection is run afterwards.

**What is the purpose of common module?**

The commonly-needed services, pipes, and directives provided by @angular/common module. Apart from these HttpClientModule is available under @angular/common/http.

**What is angular animation?**

Angular's animation system is built on CSS functionality in order to animate any property that the browser considers animatable. These properties includes positions, sizes, transforms, colors, borders etc. The Angular modules for animations are **@angular/animations** and **@angular/platform-browser** and these dependencies are automatically added to your project when you create a project using Angular CLI.

### How to inject the dynamic script in angular?

Using DomSanitizer we can inject the dynamic Html,Style,Script,Url.

**What is a service worker and its role in Angular?**

A service worker is a script that runs in the web browser and manages caching for an application. Starting from 5.0.0 version, Angular ships with a service worker implementation. Angular service worker is designed to optimize the end user experience of using an application over a slow or unreliable network connection, while also minimizing the risks of serving outdated content.

### What are the design goals of service workers?

Below are the list of design goals of Angular's service workers,

1. It caches an application just like installing a native application
2. A running application continues to run with the same version of all files without any incompatible files
3. When you refresh the application, it loads the latest fully cached version
4. When changes are published then it immediately updates in the background
5. Service workers saves the bandwidth by downloading the resources only when they changed.

### What is Angular Ivy?

Angular Ivy is a new rendering engine for Angular.

--enable-ivy

Generates Smaller Bundle

Loads faster in slower network

Improved template type checking

### What is Angular Language Service?

The Angular Language Service is a way to get completions, errors, hints, and navigation inside your Angular templates whether they are external in an HTML file or embedded in annotations/decorators in a string. It has the ability to autodetect that you are opening an Angular file, reads your tsconfig.json file, finds all the templates you have in your application, and then provides all the language services.

[**⬆ Back to Top**](https://github.com/sudheerj/angular-interview-questions#table-of-contents)

### How do you install angular language service in the project?

You can install Angular Language Service in your project with the following npm command,

npm install --save-dev @angular/language-service

After that add the following to the "compilerOptions" section of your project's tsconfig.json

"plugins": [

{"name": "@angular/language-service"}

]

**Note:** The completion and diagnostic services works for .ts files only. You need to use custom plugins for supporting HTML files.

### Explain the features provided by Angular Language Service?

**Autocompletion:** Autocompletion can speed up your development time by providing you with contextual possibilities and hints as you type with in an interpolation and elements.

**Error checking:** It can also warn you of mistakes in your code.

**Navigation:** Navigation allows you to hover a component, directive, module and then click and press F12 to go directly to its definition.

What are web workers?

[Web workers](https://developer.mozilla.org/en-US/docs/Web/API/Web_Workers_API) allow you to run CPU-intensive computations in a background thread, freeing the main thread to update the user interface.

If you are building an application where lot of calculation on ui or heavy data table using web workers can help increase your application's performance.

### What are the limitations with web workers?

You need to remember two important things when using Web Workers in Angular projects,

1. Some environments or platforms(like @angular/platform-server) used in Server-side Rendering, don't support Web Workers. In this case you need to provide a fallback mechanism to perform the computations to work in this environments.
2. Running Angular in web worker using @angular/platform-webworker is not yet supported in Angular CLI.

### What are the case types in Angular?

**camelCase :** \*ngFor

**dash-case \*ng-if**

**UPPER\_UNDERSCORE\_CASE =** NUMBER\_OF\_USERS

Titlecase for capatalize

### What are the class decorators in Angular?

A class decorator is a decorator that appears immediately before a class definition, which declares the class to be of the given type, and provides metadata suitable to the type

The following list of decorators comes under class decorators,

1. @Component()
2. @Directive()
3. @Pipe()
4. @Injectable()
5. @NgModule()

### What are class field decorators?

The class field decorators are the statements declared immediately before a field in a class definition that defines the type of that field. Some of the examples are: @input and @output,

@Input() myProperty;

@Output() myEvent = new EventEmitter();

### What is declarable in Angular?

Declarable is a class type that you can add to the declarations list of an NgModule. The class types such as components, directives, and pipes comes can be declared in the module. The structure of declarations would be,

declarations: [

YourComponent,

YourPipe,

YourDirective

],

### What is Angular DSL?

A domain-specific language (DSL) is a computer language specialized to a particular application domain. Angular has its own Domain Specific Language (DSL) which allows us to write Angular specific html-like syntax on top of normal html. It has its own compiler that compiles this syntax to html that the browser can understand. This DSL is defined in NgModules such as animations, forms, and routing and navigation.

Basically you will see 3 main syntax in Angular DSL.

1. (): Used for Output and DOM events.
2. []: Used for Input and specific DOM element attributes.
3. \*: Structural directives(\*ngFor or \*ngIf) will affect/change the DOM structure.

### what is an rxjs subject in Angular

An RxJS Subject is a special type of Observable that allows values to be multicasted to many Observers. While plain Observables are unicast (each subscribed Observer owns an independent execution of the Observable), Subjects are multicast.

A Subject is like an Observable, but can multicast to many Observers. Subjects are like EventEmitters: they maintain a registry of many listeners.

**What is Bazel tool?**

Bazel is a powerful build tool developed and massively used by Google and it can keep track of the dependencies between different packages and build targets. In Angular8, you can build your CLI application with Bazel. **Note:** The Angular framework itself is built with Bazel.

### What is platform in Angular?

A platform is the context in which an Angular application runs. The most common platform for Angular applications is a web browser, but it can also be an operating system for a mobile device, or a web server. The runtime-platform is provided by the @angular/platform-\* packages and these packages allow applications that make use of @angular/core and @angular/common to execute in different environments. i.e, Angular can be used as platform-independent framework in different environments, For example,

1. While running in the browser, it uses platform-browser package.
2. When SSR(server-side rendering ) is used, it uses platform-server package for providing web server implementation.

### How do you select an element with in a component template?

You can use @ViewChild directive to access elements in the view directly. Let's take input element with a reference,

<input #uname>

and define view child directive and access it in ngAfterViewInit lifecycle hook

@ViewChild('uname') input;

ngAfterViewInit() {

console.log(this.input.nativeElement.value);

}

### How do you detect route change in Angular?

In Angular7, you can subscribe to router to detect the changes. The subscription for router events would be as below,

this.router.events.subscribe((event: Event) => {})

### How do you pass headers for HTTP client?

You can directly pass object map for http client or create HttpHeaders class to supply the headers.

let headers = new HttpHeaders().set('header1', headerValue1); // create header object

headers = headers.append('header2', headerValue2); // add a new header, creating a new object

headers = headers.append('header3', headerValue3); // add another header

### What is the purpose of differential loading in CLI?

From Angular8 release onwards, the applications are built using differential loading strategy from CLI to build two separate bundles as part of your deployed application.

1. The first build contains ES2015 syntax which takes the advantage of built-in support in modern browsers, ships less polyfills, and results in a smaller bundle size.
2. The second build contains old ES5 syntax to support older browsers with all necessary polyfills. But this results in a larger bundle size.

### What are workspace APIs?

Angular 8.0 release introduces Workspace APIs to make it easier for developers to read and modify the angular.json file instead of manually modifying it. Currently, the only supported storage3 format is the JSON-based format used by the Angular CLI. You can enable or add optimization option for build target as below,

### How do you upgrade angular version?

ng update @angular/cli @angular/core

### What is NgUpgrade?

# Upgrade from AngularJS to Angular with ngUpgrade

### How to use polyfills in Angular application?

**Polyfills** in angular are few lines of code which make your application compatible for different browsers. The code we write is mostly in ES6(New Features: Overview and Comparison) and is not compatible with IE or firefox and needs some environment setups before being able to be viewed or used in these browsers.

### What are the ways to trigger change detection in Angular?

1. **NgZone.run(callback):** It evaluate the callback function inside the Angular zone.
2. **ChangeDetectorRef.detectChanges():** It detects only the components and it's children.

### What are the differences of various versions of Angular?

There are different versions of Angular framework. Let's see the features of all the various versions,

1. **Angular 1:**
   * Angular 1 (AngularJS) is the first angular framework released in the year 2010.
   * AngularJS is not built for mobile devices.
   * It is based on controllers with MVC architecture.
2. **Angular 2:**
   * Angular 2 was released in the year 2016. Angular 2 is a complete rewrite of Angular1 version.
   * The performance issues that Angular 1 version had has been addressed in Angular 2 version.
   * Angular 2 is built from scratch for mobile devices unlike Angular 1 version.
   * Angular 2 is components based.
3. **Angular 3:**
   * The following are the different package versions in Angular 2:
     + @angular/core v2.3.0
     + @angular/compiler v2.3.0
     + @angular/http v2.3.0
     + @angular/router v3.3.0
   * The router package is already versioned 3 so to avoid confusion switched to Angular 4 version and skipped 3 version.
4. **Angular 4:**
   * The *compiler generated code file size in AOT mode is very much reduced.*
   * With Angular 4 the production bundles size is reduced by hundreds of KB’s.
   * Animation features are removed from angular/core and formed as a separate package.
   * Supports Typescript 2.1 and 2.2.
   * Angular Universal
   * New HttpClient
5. **Angular 5:**
   * Angular 5 makes angular faster. It improved the loading time and execution time.
   * Shipped with new build optimizer.
   * Supports Typescript 2.5.
   * Service Worker
6. **Angular 6:**
   * It is released in May 2018.
   * Includes Angular Command Line Interface (CLI), Component Development KIT (CDK), Angular Material Package, Angular Elements.
   * Service Worker bug fixes.
   * i18n
   * Experimental mode for Ivy.
   * RxJS 6.0
   * Tree Shaking
7. **Angular 7:**
   * It is released in October 2018.
   * TypeScript 3.1
   * RxJS 6.3
   * New Angular CLI
   * CLI Prompts capability provide an ability to ask questions to the user before they run. It is like interactive dialog between the user and the CLI
   * With the improved CLI Prompts capability, it helps developers to make the decision. New ng commands ask users for routing and CSS styles types(SCSS) and ng add @angular/material asks for themes and gestures or animations.
8. **Angular 8:**
   * It is released in May 2019.
   * TypeScript 3.4
9. **Angular 9:**
   * It is released in February 2020.
   * TypeScript 3.7
   * Ivy enabled by default
10. **Angular 10:**
    * It is released in June 2020.
    * TypeScript 3.9
    * TSlib 2.0

### What are the security principles in angular?

Below are the list of security principles in angular,

1. You should avoid direct use of the DOM APIs.
2. You should enable Content Security Policy (CSP) and configure your web server to return appropriate CSP HTTP headers.
3. You should Use the offline template compiler.
4. You should Use Server Side XSS protection.
5. You should Use DOM Sanitizer.
6. You should Preventing CSRF or XSRF attacks.

### What is schematic?

It's a scaffolding library that defines how to generate or transform a programming project by creating, modifying, refactoring, or moving files and code. It defines rules that operate on a virtual file system called a tree.

### How do you prevent automatic sanitization?

Sometimes the applications genuinely need to include executable code such as displaying <iframe> from an URL. In this case, you need to prevent automatic sanitization in Angular by saying that you inspected a value, checked how it was generated, and made sure it will always be secure. Basically it involves 2 steps,

1. Inject DomSanitizer: You can inject DomSanitizer in component as parameter in constructor
2. Mark the trusted value by calling some of the below methods
   1. bypassSecurityTrustHtml
   2. bypassSecurityTrustScript
   3. bypassSecurityTrustStyle
   4. bypassSecurityTrustUrl
   5. bypassSecurityTrustResourceUrl

### What is DOM sanitizer?

DomSanitizer is used to help preventing Cross Site Scripting Security bugs (XSS) by sanitizing values to be safe to use in the different DOM contexts.

### What are Http Interceptors?

Http Interceptors are part of @angular/common/http, which inspect and transform HTTP requests from your application to the server and vice-versa on HTTP responses. These interceptors can perform a variety of implicit tasks, from authentication to logging.

### What is TestBed?

TestBed is an api for writing unit tests for Angular applications and it's libraries. Even though We still write our tests in Jasmine and run using Karma, this API provides an easier way to create components, handle injection, test asynchronous behaviour and interact with our application.

### What is router state?

The RouteState is an interface which represents the state of the router as a tree of activated routes.

### What is the difference between ngIf and hidden property?

The main difference is that \*ngIf will remove the element from the DOM, while [hidden] actually plays with the CSS style by setting display:none. Generally it is expensive to add and remove stuff from the DOM for frequent actions.

### What is the purpose of ngFor trackBy?

The main purpose of using \*ngFor with trackBy option is performance optimization. Normally if you use NgFor with large data sets, a small change to one item by removing or adding an item, can trigger a cascade of DOM manipulations. In this case, Angular sees only a fresh list of new object references and to replace the old DOM elements with all new DOM elements. You can help Angular to track which items added or removed by providing a trackBy function which takes the index and the current item as arguments and needs to return the unique identifier for this item.

### Is mandatory to pass static flag for ViewChild?

In Angular 8, the static flag is required for ViewChild. Whereas in Angular9, you no longer need to pass this property. Once you updated to Angular9 using ng update, the migration will remove { static: false } script everywhere.

### What is an entry component?

The entryComponent is the component which loads angular by force, that means these components are not referenced in the HTML template.

he entryComponents are only loaded dynamically and are never referenced in the component template. It refers to the array of components that are not found in HTML, instead are added by the ComponentFactoryResolver.

### What is a bootstrapped component?

A bootstrapped component is an entry component that Angular loads into the DOM during the bootstrap process or application launch time. Generally, this bootstrapped or root component is named as AppComponent in your root module using bootstrap property as below.

### How do you manually bootstrap an application?

You can use ngDoBootstrap hook for a manual bootstrapping of the application instead of using bootstrap array in @NgModule annotation. This hook is part of DoBootstap interface.

### What is Angular compiler?

The Angular compiler is used to convert the application code into JavaScript code. It reads the template markup, combines it with the corresponding component class code, and emits component factories which creates JavaScript representation of the component along with elements of @Component metadata.

[**⬆ Back to Top**](https://github.com/sudheerj/angular-interview-questions#table-of-contents)

### What is the role of ngModule metadata in compilation process?

The @NgModule metadata is used to tell the Angular compiler what components to be compiled for this module and how to link this module with other modules.

### How does angular finds components, directives and pipes?

The Angular compiler finds a component or directive in a template when it can match the selector of that component or directive in that template. Whereas it finds a pipe if the pipe's name appears within the pipe syntax of the template HTML.

**What are the differences between ngmodule and javascript module?**

Below are the main differences between Angular NgModule and javascript module,

| **NgModule** | **JavaScript module** |
| --- | --- |
| NgModule bounds declarable classes only | There is no restriction classes |
| List the module's classes in declarations array only | Can define all member classes in one giant file |
| It only export the declarable classes it owns or imports from other modules | It can export any classes |
| Extend the entire application with services by adding providers to provides array | Can't extend the application with services |

In this case, providedIn: 'root' specifies that the service should be provided in the root injector. A singleton is a class that allows only a single instance of itself to be created and gives access to that created instance.

### What is ngcc?

The ngcc(Angular Compatibility Compiler) is a tool which upgrades node\_module compiled with non-ivy ngc into ivy compliant format. The postinstall script from package.json will make sure your node\_modules will be compatible with the Ivy renderer.

### What classes should not be added to declarations?

The below class types shouldn't be added to declarations

1. A class which is already declared in any another module.
2. Directives imported from another module.
3. Module classes.
4. Service classes.
5. Non-Angular classes and objects, such as strings, numbers, functions, entity models, configurations, business logic, and helper classes.

### What is NgZone?

Angular provides a service called NgZone which creates a zone named angular to automatically trigger change detection when the following conditions are satisfied.

1. When a sync or async function is executed.
2. When there is no microTask scheduled.

### Is it mandatory to use injectable on every service class?

No. The @Injectable() decorator is not strictly required if the class has other Angular decorators on it or does not have any dependencies. But the important thing here is any class that is going to be injected with Angular is decorated. i.e, If we add the decorator, the metadata design:param types is added, and the dependency injection can do it's job. That is the exact reason to add the @Injectable() decorator on a service if this service has some dependencies itself.

### What is an optional dependency?

The optional dependency is a parameter decorator to be used on constructor parameters, which marks the parameter as being an optional dependency. Due to this, the DI framework provides null if the dependency is not found. For example, If you don't register a logger provider anywhere, the injector sets the value of logger(or logger service) to null in the below class.

import { Optional } from '@angular/core';

constructor(@Optional() private logger?: Logger) {

if (this.logger) {

this.logger.log('This is an optional dependency message');

} else {

console.log('The logger is not registered');

}

}

### What are the types of injector hierarchies?

There are two types of injector hierarchies in Angular

1. **ModuleInjector hierarchy:** It configure on a module level using an @NgModule() or @Injectable() annotation.
2. **ElementInjector hierarchy:** It created implicitly at each DOM element. Also it is empty by default unless you configure it in the providers property on @Directive() or @Component().

### What are reactive forms?

Reactive forms provide a model-driven approach to handling form inputs whose values change over time.

High-level **Differences between Template**-**driven** and **Reactive Forms**. ... **Template**-**driven forms** make use of the "FormsModule", while **reactive forms** are based on "ReactiveFormsModule". **Template**-**driven forms** are asynchronous in nature, whereas **Reactive forms** are mostly synchronous

### What are the different ways to group form controls?

**FormGroup**: It defines a form with a fixed set of controls those can be managed together in an one object. It has same properties and methods similar to a FormControl instance. This FormGroup can be nested to create complex forms as below.

**FormArray:** It defines a dynamic form in an array format, where you can add and remove controls at run time. This is useful for dynamic forms when you don’t know how many controls will be present within the group.

FormBuilder: The FormBuilder is the helper API to build forms in Angular.  It provides shortcuts to create the instance of the FormControl, [FormGroup](https://www.tektutorialshub.com/angular/formgroup-in-angular/) or FormArray. It reduces the code required to write the complex forms.

| **Form control state** | **If true** | **If false** |
| --- | --- | --- |
| Visited | ng-touched | ng-untouched |
| Value has changed | ng-dirty | ng-pristine |
| Value is valid | ng-valid | ng-invalid |

### What are the types of validator functions?

In reactive forms, the validators can be either synchronous or asynchronous functions,

1. **Sync validators:** These are the synchronous functions which take a control instance and immediately return either a set of validation errors or null. Also, these functions passed as second argument while instantiating the form control. The main use cases are simple checks like whether a field is empty, whether it exceeds a maximum length etc.
2. **Async validators:** These are the asynchronous functions which take a control instance and return a Promise or Observable that later emits a set of validation errors or null. Also, these functions passed as second argument while instantiating the form control. The main use cases are complex validations like hitting a server to check the availability of a username or email.

Q What is Routing Guard ?

Angular’s route guards are interfaces which can tell the router whether or not it should allow navigation to a requested route. They make this decision by looking for a true or false return value from a class which implements the given guard interface.

### What is a prototype chain

**Prototype chaining** is used to build new types of objects based on existing ones. It is similar to inheritance in a class based language. The prototype on object instance is available through **Object.getPrototypeOf(object)** or **proto** property whereas prototype on constructors function is available through object.prototype.

### What is the difference between Call, Apply and Bind

### **Call:** The call() method invokes a function with a given this value and arguments provided one by one

var employee1 = {firstName: 'John', lastName: 'Rodson'};

var employee2 = {firstName: 'Jimmy', lastName: 'Baily'};

function invite(greeting1, greeting2) {

console.log(greeting1 + ' ' + this.firstName + ' ' + this.lastName+ ', '+ greeting2);

}

invite.call(employee1, 'Hello', 'How are you?'); // Hello John Rodson, How are you?

invite.call(employee2, 'Hello', 'How are you?'); // Hello Jimmy Baily, How are you?

### **Apply:** Invokes the function and allows you to pass in arguments as an array

### What is the purpose of the array slice method

let arrayIntegers = [1, 2, 3, 4, 5];

let arrayIntegers1 = arrayIntegers.slice(0,2); // returns [1,2]

The map() method creates a new array with the results of calling a function for every array element.

The map() method calls the provided function once for each element in an array, in order.

### What is a first class function

### First-class functions means when functions in that language are treated like any other variable.

### What is a first order function

First-order function is a function that doesn’t accept another function as an argument and doesn’t return a function as its return value.

const firstOrder = () => console.log ('I am a first order function!');

### What is a higher order function

Higher-order function is a function that accepts another function as an argument or returns a function as a return value.

### What is a pure function

A **Pure function** is a function where the return value is only determined by its arguments without any side effects. i.e, If you call a function with the same arguments 'n' number of times and 'n' number of places in the application then it will always return the same value. Let's take an example to see the difference between pure and impure functions,

//Impure

let numberArray = [];

const impureAddNumber = number => numberArray.push (number);

//Pure

const pureAddNumber = number => argNumberArray =>

argNumberArray.concat ([number]);

//Display the results

console.log (impureAddNumber (6)); // returns 6

console.log (numberArray); // returns [6]

console.log (pureAddNumber (7) (numberArray)); // returns [6, 7]

console.log (numberArray); // returns [6]

### What is the purpose of the let keyword

The let statement declares a **block scope local variable**. Hence the variables defined with let keyword are limited in scope to the block, statement, or expression on which it is used.

### What is the Temporal Dead Zone

### In ECMAScript 6, accessing a let or const variable before its declaration (within its scope) causes a ReferenceError.

function somemethod() {

console.log(counter1); // undefined

console.log(counter2); // ReferenceError

var counter1 = 1;

let counter2 = 2;

}

### What are closures

### A closure is a function having access to the parent scope, even after the parent function has closed.

function makeFunc() {

var name = 'Mozilla';

function displayName() {

alert(name);

}

return displayName;

}

var myFunc = makeFunc();

myFunc();

### What is a service worker

A Service worker is basically a script (JavaScript file) that runs in the background, separate from a web page and provides features that don't need a web page or user interaction. Some of the major features of service workers are Rich offline experiences(offline first web application development), periodic background syncs, push notifications, intercept and handle network requests and programmatically managing a cache of responses.

### How do you manipulate DOM using a service worker

Service worker can't access the DOM directly. But it can communicate with the pages it controls by responding to messages sent via the postMessage interface, and those pages can manipulate the DOM.

### What is IndexedDB

IndexedDB is a low-level API for client-side storage of larger amounts of structured data, including files/blobs. This API uses indexes to enable high-performance searches of this data.

### What is a post message

Post message is a method that enables cross-origin communication between Window objects.(i.e, between a page and a pop-up that it spawned, or between a page and an iframe embedded within it). Generally, scripts on different pages are allowed to access each other if and only if the pages follow same-origin policy(i.e, pages share the same protocol, port number, and host).

### What are the three states of promise

Promises have three states:

1. **Pending:** This is an initial state of the Promise before an operation begins
2. **Fulfilled:** This state indicates that the specified operation was completed.
3. **Rejected:** This state indicates that the operation did not complete. In this case an error value will be thrown.

### What is the purpose of double exclamation

The double exclamation or negation(!!) ensures the resulting type is a boolean. If it was falsey (e.g. 0, null, undefined, etc.), it will be false, otherwise, true. For example, you can test IE version using this expression as below,

| **Null** | **Undefined** |
| --- | --- |
| It is an assignment value which indicates that variable points to no object. | It is not an assignment value where a variable has been declared but has not yet been assigned a value. |
| Type of null is object | Type of undefined is undefined |
| The null value is a primitive value that represents the null, empty, or non-existent reference. | The undefined value is a primitive value used when a variable has not been assigned a value. |
| Indicates the absence of a value for a variable | Indicates absence of variable itself |
| Converted to zero (0) while performing primitive operations | Converted to NaN while performing primitive operations |

### What is an event flow

Event flow is the order in which event is received on the web page. When you click an element that is nested in various other elements, before your click actually reaches its destination, or target element, it must trigger the click event for each of its parent elements first, starting at the top with the global window object. There are two ways of event flow

1. Top to Bottom(Event Capturing)
2. Bottom to Top (Event Bubbling)

### What is event bubbling

Event bubbling is a type of event propagation where the event first triggers on the innermost target element, and then successively triggers on the ancestors (parents) of the target element in the same nesting hierarchy till it reaches the outermost DOM element.

[**⬆ Back to Top**](https://github.com/sudheerj/javascript-interview-questions#table-of-contents)

### What is event capturing

Event capturing is a type of event propagation where the event is first captured by the outermost element, and then successively triggers on the descendants (children) of the target element in the same nesting hierarchy till it reaches the innermost DOM element.

### What is same-origin policy

### The same-origin policy is a policy that prevents JavaScript from making requests across domain boundaries. An origin is defined as a combination of URI scheme, hostname, and port number. JavaScript code and the Document Object Model (DOM), for example, a page cannot access the content of its iframe unless they are of the same origin.

### What is the purpose of void 0

### Void(0) is used to prevent the page from refreshing

### What is the use of preventDefault method

The preventDefault() method cancels the event if it is cancelable, meaning that the default action or behaviour that belongs to the event will not occur. For example, prevent form submission when clicking on submit button and prevent opening the page URL when clicking on hyperlink are some common use cases.

document.getElementById("link").addEventListener("click", function(event){

event.preventDefault();

});

### What is the use of stopPropagation method

The stopPropagation method is used to stop the event from bubbling up the event chain. For example, the below nested divs with stopPropagation method prevents default event propagation when clicking on nested div(Div1)

### What is an event delegation

Event delegation is a technique for listening to events where you delegate a parent element as the listener for all of the events that happen inside it.

For example, if you wanted to detect field changes in inside a specific form, you can use event delegation technique,

var form = document.querySelector('#registration-form');

// Listen for changes to fields inside the form

form.addEventListener('input', function (event) {

// Log the field that was changed

console.log(event.target);

}, false);

### What is ECMAScript

ECMAScript is the scripting language that forms the basis of JavaScript.

### How do you check whether a string contains a substring

var mainString = "hello", subString = "hell";

mainString.includes(subString)

var mainString = "hello", subString = "hell";

mainString.indexOf(subString) !== -1

**What are the various url properties of location object**

The below Location object properties can be used to access URL components of the page,

1. href - The entire URL
2. protocol - The protocol of the URL
3. host - The hostname and port of the URL
4. hostname - The hostname of the URL
5. port - The port number in the URL
6. pathname - The path name of the URL
7. search - The query portion of the URL
8. hash - The anchor portion of the URL

### How do you check if a string starts with another string

You can use ECMAScript 6's String.prototype.startsWith()

### What is tree shaking

Tree shaking is a form of dead code elimination. It means that unused modules will not be included in the bundle during the build process and for that it relies on the static structure of ES2015 module syntax,( i.e. import and export). Initially this has been popularized by the ES2015 module bundler rollup.

### What is a debugger statement

The debugger statement invokes any available debugging functionality, such as setting a breakpoint. If no debugging functionality is available, this statement has no effect. For example, in the below function a debugger statement has been inserted. So execution is paused at the debugger statement just like a breakpoint in the script source.

### How do you get the image width and height using JS

You can programmatically get the image and check the dimensions(width and height) using Javascript.

var img = new Image();

img.onload = function() {

console.log(this.width + 'x' + this.height);

}

img.src = 'http://www

**What are the ways to execute javascript after page load**

You can execute javascript after page load in many different ways,

1. **window.onload:**

window.onload = function ...

1. **document.onload:**

document.onload = function ...

1. **body onload:**

<body onload="script();">

### What is a freeze method

The **freeze()** method is used to freeze an object. Freezing an object does not allow adding new properties to an object,prevents from removing and prevents changing the enumerability, configurability, or writability of existing properties. i.e, It returns the passed object and does not create a frozen copy.

### How do you detect javascript disabled in the page

You can use the <noscript> tag to detect javascript disabled or not. The code block inside <noscript> gets executed when JavaScript is disabled, and is typically used to display alternative content when the page generated in JavaScript.

### What is a rest parameter

Rest parameter is an improved way to handle function parameters which allows us to represent an indefinite number of arguments as an array. The syntax would be as below,

function f(a, b, ...theArgs) {

// ...

}

For example, let's take a sum example to calculate on dynamic number of parameters,

function total(…args){

let sum = 0;

for(let i of args){

sum+=i;

}

return sum;

}

console.log(fun(1,2)); //3

console.log(fun(1,2,3)); //6

console.log(fun(1,2,3,4)); //13

console.log(fun(1,2,3,4,5)); //15

### What is a spread operator

Spread operator allows iterables( arrays / objects / strings ) to be expanded into single arguments/elements. Let's take an example to see this behavior,

function calculateSum(x, y, z) {

return x + y + z;

}

const numbers = [1, 2, 3];

console.log(calculateSum(...numbers)); // 6

### How do you copy properties from one object to other

You can use the Object.assign() method which is used to copy the values and properties from one or more source objects to a target object. It returns the target object which has properties and values copied from the target object. The syntax would be as below,

Object.assign(target, ...sources)

Let's take example with one source and one target object,

const target = { a: 1, b: 2 };

const source = { b: 3, c: 4 };

const returnedTarget = Object.assign(target, source);

console.log(target); // { a: 1, b: 3, c: 4 }

console.log(returnedTarget); // { a: 1, b: 3, c: 4 }

### What is the purpose of seal method

The **Object.seal()** method is used to seal an object, by preventing new properties from being added to it and marking all existing properties as non-configurable. But values of present properties can still be changed as long as they are writable. Let's see the below example to understand more about seal() method

### What is an anonymous function

An anonymous function is a function without a name! Anonymous functions are commonly assigned to a variable name or used as a callback function. The syntax would be as below,

function (optionalParameters) {

//do something

}

const myFunction = function(){ //Anonymous function assigned to a variable

//do something

};

[1, 2, 3].map(function(element){ //Anonymous function used as a callback function

//do something

});

View child static true and false meaning ?

12

Use { static: true } when you want to access the ViewChild in ngOnInit.

Use { static: false } will be accessible only in ngAfterViewInit. This is also what you want to do for when you have a structural directive (\*ngIf etc.) in your template.

In most cases { static: false } will work.

passing by value and reference

disadvantages of call back

event loop

design pattern in angular