What is Angular? Why was it introduced?

Angular was introduced to create Single Page applications. This framework brings structure and consistency to web applications and provides excellent scalability and maintainability.

Angular is an open-source, JavaScript framework wholly written in TypeScript. It uses HTML's syntax to express your application's components clearly.

What is TypeScript?

TypeScript is a superset of JavaScript that offers excellent consistency. It is highly recommended, as it provides some syntactic sugar and makes the code base more comfortable to understand and maintain. Ultimately, TypeScript code compiles down to JavaScript that can run efficiently in any environment.

What is data binding? Which type of data binding does Angular deploy?

Data binding is a phenomenon that allows any internet user to manipulate Web page elements using a Web browser. It uses dynamic HTML and does not require complex scripting or programming. We use data binding in web pages that contain interactive components such as forms, calculators, tutorials, and games. Incremental display of a webpage makes data binding convenient when pages have an enormous amount of data.

Angular uses the two-way binding. Any changes made to the user interface are reflected in the corresponding model state.

What are Single Page Applications (SPA)?

Single-page applications are web applications that load once with new features just being mere additions to the user interface. It does not load new HTML pages to display the new page's content, instead generated dynamically. This is made possible through JavaScript's ability to manipulate the DOM elements on the existing page itself. A SPA approach is faster, thus providing a seamless user experience.

What are decorators in Angular?

Decorators are a design pattern or functions that define how Angular features work. They are used to make prior modifications to a class, service, or filter. Angular supports four types of decorators, they are:

Class Decorators

Property Decorators

Method Decorators

Parameter Decorators

Mention some advantages of Angular.

Some of the common advantages of Angular are -

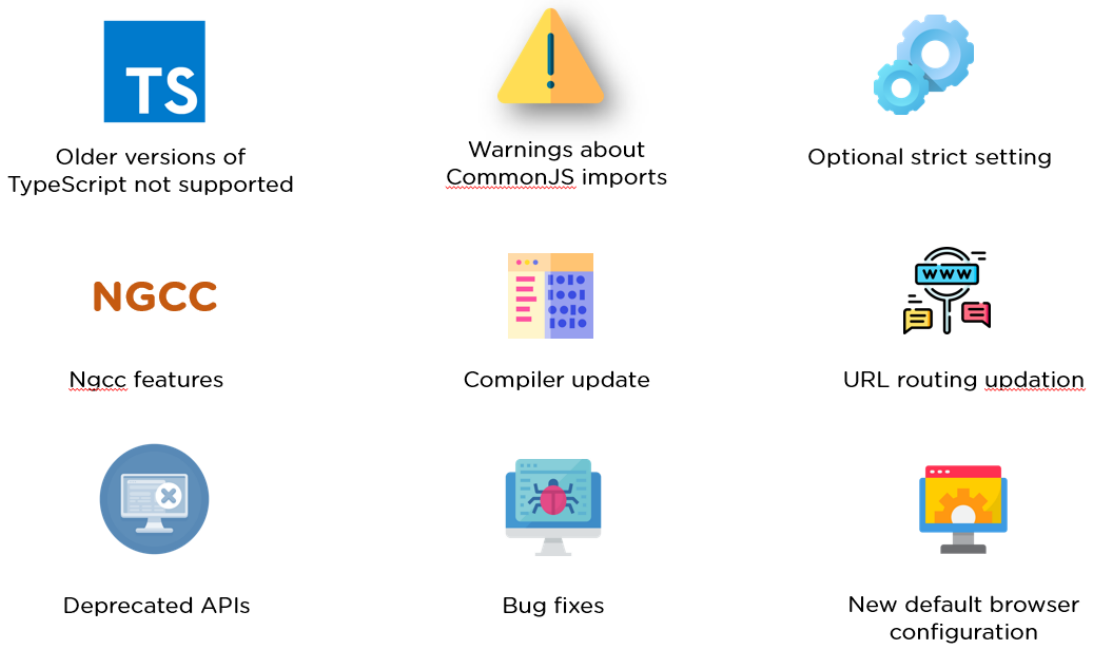
MVC architecture - Angular is a full-fledged MVC framework. It provides a firm opinion on how the application should be structured. It also offers bi-directional data flow and updates the real DOM.

Modules: Angular consists of different design patterns like components, directives, pipes, and services, which help in the smooth creation of applications.

Dependency injection: Components dependent on other components can be easily worked around using this feature.

Other generic advantages include clean and maintainable code, unit testing, reusable components, data binding, and excellent responsive experience

What are the new updates with Angular 10?


Older versions of TypeScript not supported
Warnings about CommonJS imports
Optional strict setting
NGCC
Ngcc features
Compiler update
URL routing updation
Deprecated APIs
Bug fixes
New default browser configuration

Older versions of TypeScript not supported - Previous versions of Angular supported typescript 3.6, 3.7, and even 3.8. But with Angular 10, TypeScript bumped to TypeScript 3.9.

Warnings about CommonJS imports - Logging of unknown property bindings or element names in templates is increased to the "error" level, which was previously a "warning" before.

Optional strict setting - Version 10 offers a stricter project setup when you create a new workspace with ng new command.

ng new --strict

NGCC Feature - Addition of NGCC features with a program based entry point finder.

Updated URL routing

Deprecated APIs - Angular 10 has several deprecated APIs.

Bug fixes - With this Angular 10 version, there have been a number of bug fixes, important ones being the compiler avoiding undefined expressions and the core avoiding a migration error when a nonexistent symbol is imported.

New Default Browser Configuration - Browser configuration for new projects has been upgraded to outdo older and less used browsers.

What are Templates in Angular?

Angular Templates are written with HTML that contains Angular-specific elements and attributes. In combination with the model and controller's information, these templates are further rendered to provide a dynamic view to the user

What are Annotations in Angular?

Annotations in Angular are used for creating an annotation array. They are the metadata set on the class that is used to reflect the Metadata library.

What are Directives in Angular?

Directives are attributes that allow the user to write new HTML syntax specific to their applications. They execute whenever the Angular compiler finds them in the DOM.

Angular supports three types of directives :

Component Directives, Structural Directives & Attribute Directives .

What is an AOT compilation? What are its advantages?

The Ahead-of-time (AOT) compiler converts the Angular HTML and TypeScript code into JavaScript code during the build phase, i.e., before the browser downloads and runs the code.

Some of its advantages are as follows.

Faster rendering

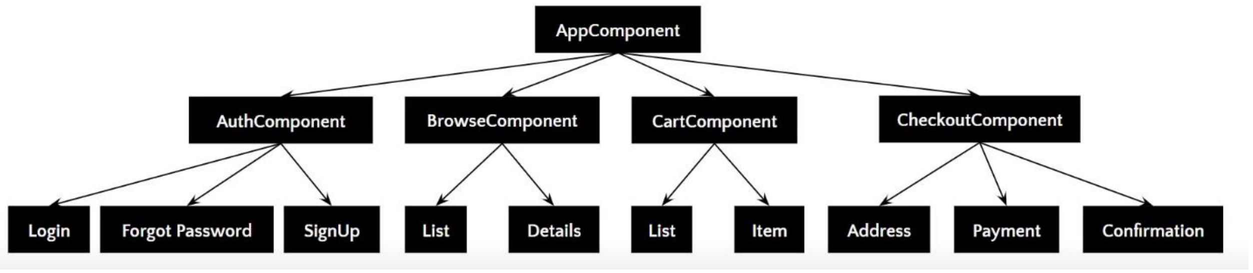
Fewer asynchronous requests

Smaller Angular framework download size

Quick detection of template errors

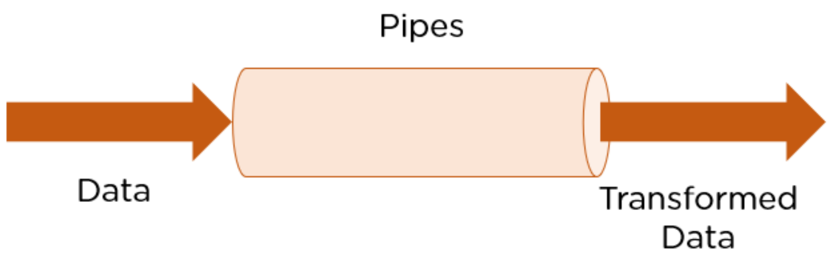
Better security

What are Components in Angular?



Components are the basic building blocks of the user interface in an Angular application. Every component is associated with a template and is a subset of directives. An Angular application typically consists of a root component, which is the AppComponent, that then branches out into other components creating a hierarchy.

What are Pipes in Angular?



Pipes are simple functions designed to accept an input value, process, and return as an output, a transformed value in a more technical understanding. Angular supports several built-in pipes. However, you can also create custom pipes that cater to your needs.

Some key features include:

Pipes are defined using the pipe “|” symbol.

Pipes can be chained with other pipes.

Pipes can be provided with arguments by using the colon (:) sign

What is the PipeTransform interface?

As the name suggests, the interface receives an input value and transforms it into the desired format with a transform() method. It is typically used to implement custom pipes.

```
import { Pipe, PipeTransform } from '@angular/core';
 @Pipe({
  name: 'demopipe'
})
export class DemopipePipe implements PipeTransform {

  transform(value: unknown, ...args: unknown[]): unknown {

    return null;
  }
}
```

What are Pure Pipes?

These pipes are pipes that use pure functions. As a result of this, a pure pipe doesn't use any internal state, and the output remains the same as long as the parameters passed stay the same. Angular calls the pipe only when it detects a change in the parameters being passed. A single instance of the pure pipe is used throughout all components.

What are Impure Pipes?

For every change detection cycle in Angular, an impure pipe is called regardless of the change in the input fields. Multiple pipe instances are created for these pipes. Inputs passed to these pipes can be mutable.

```
@Pipe({
  name: 'demopipe',
  pure : true/false
})
export class DemopipePipe implements PipeTransform {
```

What is an ngModule?

NgModules are containers that reserve a block of code to an application domain or a workflow. @NgModule takes a metadata object that generally describes the way to compile the template of a component and to generate an injector at runtime. In addition, it identifies the module's components, directives, and pipes, making some of them public, through the export property so that external components can use them.

What are filters in Angular? Name a few of them.

Filters are used to format an expression and present it to the user. They can be used in view templates, controllers, or services. Some inbuilt filters are as follows:

date - Format a date to a specified format.

filter - Select a subset of items from an array.

Json - Format an object to a JSON string.

limitTo - Limits an array/string, into a specified number of elements/characters.

lowercase - Format a string to lowercase.

What is view encapsulation in Angular?

View encapsulation defines whether the template and styles defined within the component can affect the whole application or vice versa. Angular provides three encapsulation strategies:

Emulated - styles from the main HTML propagate to the component.

Native - styles from the main HTML do not propagate to the component.

None - styles from the component propagate back to the main HTML and therefore are visible to all components on the page.

Explain the lifecycle hooks in Angular

In Angular, every component has a lifecycle. Angular creates and renders these components and also destroys them before removing them from the DOM. This is achieved with the help of lifecycle hooks. Here's the list of them -

ngOnChanges() - Responds when Angular sets/resets data-bound input properties.

ngOnInit() - Initialize the directive/component after Angular first displays the data-bound properties and sets the directive/component's input properties/

ngDoCheck() - Detect and act upon changes that Angular can't or won't detect on its own.

ngAfterContentInit() - Responds after Angular projects external content into the component's view.

ngAfterContentChecked() - Respond after Angular checks the content projected into the component.

ngAfterViewInit() - Respond after Angular initializes the component's views and child views.

ngAfterViewChecked() - Respond after Angular checks the component's views and child views.

ngOnDestroy() - Cleanup just before Angular destroys the directive/component

What is String Interpolation in Angular?

String Interpolation is a one-way data-binding technique that outputs the data from TypeScript code to HTML view. It is denoted using double curly braces. This template expression helps display the data from the component to the view.

{{ data }}

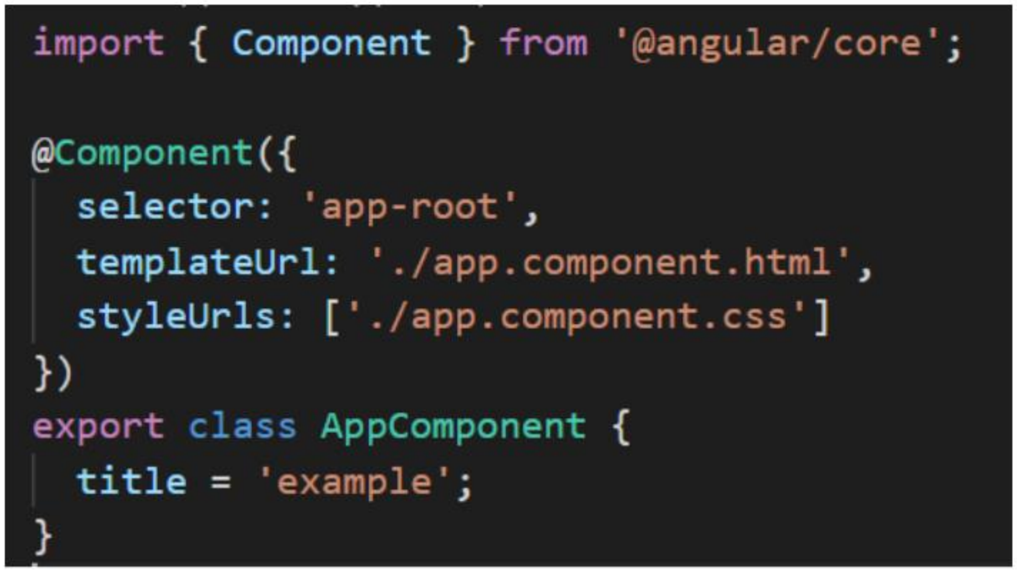
What is the difference between AOT and JIT?

Ahead of Time (AOT) compilation converts your code during the build time before the browser downloads and runs that code. This ensures faster rendering to the browser. To specify AOT compilation, include the --aot option with the ng build or ng serve command.

The Just-in-Time (JIT) compilation process is a way of compiling computer code to machine code during execution or run time. It is also known as dynamic compilation. JIT compilation is the default when you run the ng build or ng serve CLI commands.

Explain the @Component Decorator.

TypeScript class is one that is used to create components. This genre of class is then decorated with the "@Component" decorator. The decorato's purpose is to accept a metadata object that provides relevant information about the component.

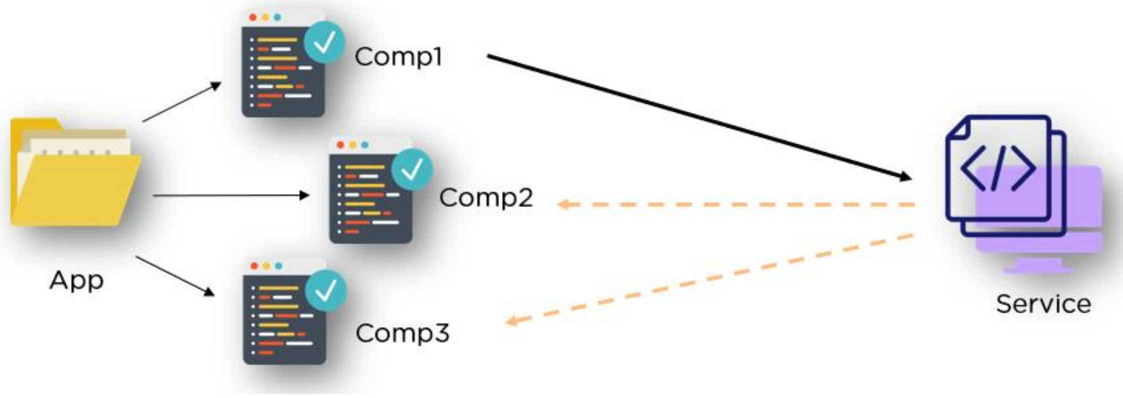


```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'example';
}
```

What are Services in Angular?

Angular Services perform tasks that are used by multiple components. These tasks could be data and image fetching, network connections, and database management among others. They perform all the operational tasks for the components and avoid rewriting of code. A service can be written once and injected into all the components that use that service.



What are Promises and Observables in Angular?

While both the concepts deal with Asynchronous events in Angular, Promises handle one such event at a time while observables handle a sequence of events over some time.

Promises - They emit a single value at a time. They execute immediately after creation and are not cancellable. They are Push errors to the child promises.

Observables - They are only executed when subscribed to them using the subscribe() method. They emit multiple values over a period of time. They help perform operations like forEach, filter, and retry, among others. They deliver errors to the subscribers. When the unsubscribe() method is called, the listener stops receiving further values.

What is ngOnInit? How is it defined?

ngOnInit is a lifecycle hook and a callback method that is run by Angular to indicate that a component has been created. It takes no parameters and returns a void type.

```
export class MyComponent implements OnInit {

 constructor() { }

 ngOnInit(): void {

   //....

 }

}
```

How to use ngFor in a tag?

The ngFor directive is used to build lists and tables in the HTML templates. In simple terms, this directive is used to iterate over an array or an object and create a template for each element.

```
<ul>

    <li *ngFor = "let items in itemlist"> {{ item }} </li>

 </ul>
```

“Let item” creates a local variable that will be available in the template

“Of items” indicates that we are iterating over the items iterable.

The * before ngFor creates a parent template.

What are Template and Reactive forms?

Template-driven approach

In this method, the conventional form tag is used to create forms. Angular automatically interprets and creates a form object representation for the tag.

Controls can be added to the form using the NGModel tag. Multiple controls can be grouped using the NGControlGroup module.

A form value can be generated using the “form.value” object. Form data is exported as JSON values when the submit method is called.

Basic HTML validations can be used to validate the form fields. In the case of custom validations, directives can be used.

Arguably, this method is the simplest way to create an Angular App.

Reactive Form Approach

This approach is the programming paradigm oriented around data flows and propagation of change.

With Reactive forms, the component directly manages the data flows between the form controls and the data models.

Reactive forms are code-driven, unlike the template-driven approach.

Reactive forms break from the traditional declarative approach.

Reactive forms eliminate the anti-pattern of updating the data model via two-way data binding.

Typically, Reactive form control creation is synchronous and can be unit tested with synchronous programming techniques.

What is Bootstrap? How is it embedded into Angular?

Bootstrap is a powerful toolkit. It is a collection of HTML, CSS, and JavaScript tools for creating and building responsive web pages and web applications.

There are two ways to embed the bootstrap library into your application.

Angular Bootstrap via CDN - Bootstrap CDN is a public Content Delivery Network. It enables you to load the CSS and JavaScript files remotely from its servers.

Angular Bootstrap via NPM - Another way to add Bootstrap to your Angular project is to install it into your project folder by using NPM (Node Package Manager).

```
npm install bootstrap

npm install jquery
```

What is Eager and Lazy loading?

Eager loading is the default module-loading strategy. Feature modules under Eager loading are loaded before the application starts. This is typically used for small size applications.

Lazy loading dynamically loads the feature modules when there's a demand. This makes the application faster. It is used for bigger applications where all the modules are not required at the start of the application.

What type of DOM does Angular implement?

DOM (Document Object Model) treats an XML or HTML document as a tree structure in which each node is an object representing a part of the document.

Angular uses the regular DOM. This updates the entire tree structure of HTML tags until it reaches the data to be updated. However, to ensure that the speed and performance are not affected, Angular implements Change Detection.

With this, you have reached the end of the article. We highly recommend brushing up on the core concepts for an interview. It’s always an added advantage to write the code in places necessary.

Why were client-side frameworks like Angular introduced?

Client-side frameworks like Angular were introduced to provide a more responsive user experience. By using a framework, developers can create web applications that are more interactive and therefore provide a better user experience.

Frameworks like Angular also make it easier for developers to create single-page applications (SPAs). SPAs are web applications that only need to load a single HTML page. This makes them much faster and more responsive than traditional web applications.

How does an Angular application work?

An Angular application is a Single Page Application, or SPA. This means that the entire application lives within a single page, and all of the resources (HTML, CSS, JavaScript, etc.) are loaded when the page is first loaded. Angular uses the Model-View-Controller, or MVC, architecture pattern to manage its data and views. The Model is the data that the application uses, the View is what the user sees, and the Controller is responsible for managing communication between the Model and the View.

When a user interacts with an Angular application, the Angular framework will automatically update the View to reflect any changes in the data. This means that Angular applications are very responsive and fast, as the user does not need to wait for the page to reload in order to see updated data.

Angular applications are also very scalable, as they can be divided into small modules that can be loaded independently of each other. This means that an Angular application can be easily extended with new functionality without having to rewrite the entire application.

Overall, Angular applications are very fast, responsive, and scalable. They are easy to develop and extend, and provide a great user experience.

The following is is an example of coding from an angular.json file:

```
 "build": {

      "builder": "@angular-devkit/build-angular:browser",

      "options": {

        "outputPath": "dist/angular-starter",

        "index": "src/index.html",
        "main": "src/main.ts",

        "polyfills": "src/polyfills.ts",

        "tsConfig": "tsconfig.app.json",

        "aot": false,

        "assets": [

          "src/favicon.ico",

          "src/assets"

        ],

        "styles": [

          "./node_modules/@angular/material/prebuilt-themes/deeppurple-amber.css",

          "src/style.css"

        ]

      }

    }
```

How are Angular expressions different from JavaScript expressions?

One major difference between Angular expressions and JavaScript expressions is that Angular expressions are compiled while JavaScript expressions are not. This means that Angular expressions are more efficient since they're already pre-processed. Additionally, Angular expressions can access scope properties while JavaScript expressions cannot. Finally, Angular expressions support some additional features such as filters and directives which aren't available in JavaScript expressions.

Explain - Angular by default, uses client-side rendering for its applications.

This means that the Angular application is rendered on the client-side — in the user's web browser. Client-side rendering has a number of advantages, including improved performance and better security. However, there are some drawbacks to using client-side rendering, as well. One of the biggest drawbacks is that it can make your application more difficult to debug.

Client-side rendering is typically used for applications that are not heavily data-driven. If your application relies on a lot of data from a server, client-side rendering can be very slow. Additionally, if you're using client-side rendering, it's important to be careful about how you load and cache your data. If you're not careful, you can easily end up with an application that is very slow and difficult to use. When rendered on the server-side, this is called Angular Universal

How do you share data between components in Angular?

using: service, route, input, output, viewChild, viewChildren, contentChild, contentChildren,subject

Explain the concept of dependency injection.

Dependency injection is a technique used to create loosely coupled code. It allows pieces of code to be reused without the need for hard-coded dependencies. This makes code more maintainable and easier to test.

The first step is to create a dependency injection container. This is a simple object that will hold all of the dependencies that your code needs. Next, you need to register all of the dependencies that your code will need with the container. The registration process will vary depending on the library you are using, but it is usually just a matter of providing the dependency's name and constructor function.

Explain MVVM architecture.

MVVM architecture is an architectural pattern used mainly in software engineering. It stands for Model-View-ViewModel. MVVM is a variation of the traditional MVC (Model-View-Controller) software design pattern. The main difference between the two is that MVVM separates the user interface logic from the business logic, while MVC separates the data access logic from the business logic. This separation of concerns makes it easier to develop, test, and maintain software applications.

What are RxJs in Angular?

RxJs is a library that provides reactive programming support for Angular applications. It allows you to work with asynchronous data streams and handle events over time. RxJs is based on Observables, which are data streams that can be subscribed to and processed using operators. It provides a powerful and flexible way to handle complex asynchronous operations in Angular.

What exactly is a parameterized pipe?

A parameterized pipe in Angular is a pipe that accepts one or more arguments, also known as parameters. Pipes transform data in Angular templates, and parameterized pipes allow you to customize the transformation based on specific requirements. By passing parameters to a pipe, you can modify its behavior and apply different transformations to the data.

Explain the concept of dependency injection.

Dependency injection is a technique used to create loosely coupled code. It allows pieces of code to be reused without the need for hard-coded dependencies. This makes code more maintainable and easier to test. Dependency injection is often used in frameworks like AngularJS, ReactJS, and VueJS. It is also possible to use dependency injection in vanilla JavaScript. To use dependency injection in JavaScript, you need a dependency injection library. There are many different libraries available, but they all work in basically the same way.

The first step is to create a dependency injection container. This is a simple object that will hold all of the dependencies that your code needs. Next, you need to register all of the dependencies that your code will need with the container. The registration process will vary depending on the library you are using, but it is usually just a matter of providing the dependency's name and constructor function.

Once all of the dependencies have been registered, you can then inject them into your code. The injection process will again vary depending on the library you are using, but it is usually just a matter of providing the dependency's name. The library will then take care of instantiating the dependency and passing it to your code.

Explain MVVM architecture.

MVVM architecture is an architectural pattern used mainly in software engineering. It stands for Model-View-ViewModel. MVVM is a variation of the traditional MVC (Model-View-Controller) software design pattern. The main difference between the two is that MVVM separates the user interface logic from the business logic, while MVC separates the data access logic from the business logic. This separation of concerns makes it easier to develop, test, and maintain software applications.

The Model layer in MVVM architecture is responsible for storing and managing data. It can be a database, a web service, or a local data source. The View layer is responsible for displaying data to the user. It can be a graphical user interface (GUI), a command-line interface (CLI), or a web page. The ViewModel layer is responsible for handling user input and updating the View layer accordingly. It contains the business logic of the application.

MVVM architecture is often used in conjunction with other software design patterns, such as Model-View-Presenter (MVP) and Model-View-Controller (MVC). These patterns can be used together to create a complete software application.

MVVM architecture is a popular choice for modern software applications. It allows developers to create applications that are more responsive and easier to maintain. Additionally, MVVM architecture can be used to create applications that can be easily ported to different platforms.

What are RxJs in Angular?

RxJs is a library that provides reactive programming support for Angular applications. It allows you to work with asynchronous data streams and handle events over time. RxJs is based on Observables, which are data streams that can be subscribed to and processed using operators. It provides a powerful and flexible way to handle complex asynchronous operations in Angular.

What exactly is a parameterized pipe?

A parameterized pipe in Angular is a pipe that accepts one or more arguments, also known as parameters. Pipes transform data in Angular templates, and parameterized pipes allow you to customize the transformation based on specific requirements. By passing parameters to a pipe, you can modify its behavior and apply different transformations to the data.

What are class decorators?

Class decorators in Angular are a type of decorator that can be applied to a class declaration. They are used to modify the behavior of the class or add additional functionality. Class decorators are defined using the @ symbol followed by the decorator name and are placed immediately before the class declaration. They can be used for various purposes, such as adding metadata, applying mixins, or extending the functionality of a class.

What are Method decorators?

Method decorators in Angular are decorators that can be applied to methods within a class. They are used to modify the behavior of the method or add additional functionality. Method decorators are defined using the @ symbol followed by the decorator name and are placed immediately before the method declaration. They can be used for tasks like logging, caching, or modifying the method's behavior based on specific conditions.

What are property decorators?

Property decorators in Angular are decorators that can be applied to class properties. They are used to modify the behavior of the property or add additional functionality. Property decorators are defined using the @ symbol followed by the decorator name and are placed immediately before the property declaration. They can be used for tasks like validation, memoization, or accessing metadata associated with the property.

What are router links?

Router links in Angular are used for navigation within an application. They are defined using the routerLink directive and provide a way to navigate to different routes or components. Router links can be used in HTML templates and are typically placed on anchor (<a>) tags or other clickable elements. By specifying the destination route or component, router links allow users to navigate between different parts of an Angular application.

What exactly is the router state?

The router state in Angular represents the current state of the Angular router. It contains information about the current route, including the URL, route parameters, query parameters, and other related data. The router state can be accessed and manipulated using the Angular Router service. It provides a way to programmatically navigate, retrieve information about the current route, and handle route changes in Angular applications.

What is transpiling in Angular?

Transpiling in Angular refers to the process of converting TypeScript code into JavaScript code that web browsers can execute. Angular applications are built using TypeScript, a superset of JavaScript that adds static typing and additional features to the language. Since browsers can only run JavaScript, the TypeScript code needs to be transpiled into JavaScript before it can be executed. This is typically done using the TypeScript compiler (tsc) or build tools like Angular CLI.

What are HTTP interceptors?

HTTP interceptors in Angular are a feature that allows you to intercept HTTP requests and responses globally. Interceptors provide a way to modify or handle HTTP requests and responses at a centralized location before they reach the server or client. This can be useful for logging requests, adding authentication headers, handling errors, or modifying request/response data. Interceptors can be registered in the Angular module and are executed in a specific order based on the request/response pipeline.

What is Change Detection, and how does the Change Detection Mechanism work?

Change Detection in Angular is a mechanism that determines when and how to update the user interface based on changes in the application's data model. Angular uses a tree of change detectors to track changes in component properties and update the DOM accordingly. When a change occurs, Angular performs a process called change detection, which involves checking each component's properties for changes and updating the DOM if necessary. The change detection mechanism is responsible for keeping the UI in sync with the application's data.

How Angular2+ is different than Angular JS

Angular and AngularJS are two entirely different and unrelated frontend frameworks, although both are built by Google and have the same prefix in their name. AngularJS was released in 2010 and it was eventually replaced by Angular. Angular 7 is the latest version of Angular.

Angular is a TypeScript based development platform whereas AngularJS is a JavaScript-based platform.

Angular 7 comes with powerful methods to easily build frontend applications. These features include support for two-way data binding, Angular Elements, CLI workspaces, Extended library support, Animation improvements, RxJS, etc.

What are the essential building blocks of Angular?

Angular consists of a few core components that help you build applications:

Components - Components are the building blocks of an Angular application. Each component consists of a Typescript class containing a decorator, an HTML template, and styles.

Templates - Templates are a form of HTML tags that are present in components. These templates are used to declare how a component must be rendered.

Modules - Modules are also considered as building blocks in Angular. However, modules have a specific set of capabilities or workflow.

Services - Services could be broadly defined as a class with a well-defined purpose.

Metadata - Metadata is used to add data to a class. This data helps Angular understand how a class needs to be processed.

What is authentication and authorization in Angular?

During login, the credentials are sent to an authentication API. This API is present on the server and validation is done there. After a JWT (JSON Web Token) is returned, this token has information about the user and is used to identify the user. This process is called Authentication.

After authentication, users are given various levels of permissions/ access. Some users may have access to all the pages and some might not. This process of restricting the content is called Authorization.

What are activated routes in Angular?

Activated routes provide access to information about the route associated with a component. These components are loaded in an outlet.

This method is used to traverse the RouterState tree and get information from the nodes.

Differentiate between declaration, provider and import in the NgModule?

Declarations - Declarations are utilized to create directives for the current modules available. The selectors of the various directives are only matched with their HTML if they are declared or imported.

Providers - Provides services and values known to the dependency injection. Since they are injected into other services or directives, providers must be added to the root scope.

Import - Import is used to make declarations of other modules available in the current module.