1. What are RxJs in Angular?

RxJs is a library for reactive programming using observables in Angular. It provides a set of powerful tools for handling asynchronous operations, managing data streams, and responding to user events.

1. Explain string interpolation and property binding in Angular.

String interpolation and property binding are two ways to display dynamic data in an Angular template. String interpolation is a one-way binding that allows for embedding expressions within double curly braces {{}}. Property binding, on the other hand, is a one-way or two-way binding that allows for binding a component property to an element property using square brackets [].

1. How are observables different from promises?

Observables and promises are both used for handling asynchronous operations in Angular. However, observables are a more powerful and flexible abstraction that can handle multiple values over time, while promises are only designed to handle a single value. Observables can also be cancelled, retried, and composed, while promises are simpler and more straightforward.

1. Explain the concept of Dependency Injection?

Dependency Injection is a design pattern in Angular that allows for decoupling components and services from their dependencies. It involves injecting dependencies into a component or service through their constructor, instead of creating them directly. This makes the code more modular, testable, and scalable, and allows for easy swapping of dependencies.

1. What are pipes in Angular explain with an example?

Pipes are a feature in Angular that allow for transforming data in a template. They are defined as a directive and can be used to format, filter, and sort data before displaying it to the user. An example of a pipe in Angular is the DatePipe, which is used to format a date object in a specific format, such as "yyyy-MM-dd".

1. What exactly is a parameterized pipe?

A parameterized pipe is a type of pipe in Angular that accepts one or more arguments. It allows the user to pass dynamic values into the pipe to modify the output. The arguments are passed into the pipe after the pipe name, separated by a colon.

1. What are class decorators?

Class decorators are a type of decorator in Angular that are used to modify the behavior of a class. They are functions that are applied to a class declaration, immediately following the class keyword. They can be used to add metadata, modify the class behavior, or extend the class functionality.

1. What are Method decorators?

Method decorators are a type of decorator in Angular that are used to modify the behavior of a method within a class. They are functions that are applied to a method declaration, immediately preceding the method keyword. They can be used to add metadata, modify the method behavior, or extend the method functionality.

1. What are property decorators?

Property decorators are a type of decorator in Angular that can be used to add metadata or behavior to class properties. They are defined using the @propertyDecoratorName syntax, and can be used to add features such as validation, caching, or custom getters and setters to class properties.

1. What is the Component Decorator in Angular?

The Component decorator in Angular is a special kind of decorator that is used to define and configure components in an Angular application. It is used to decorate the component class and provides metadata about the component, such as its selector, template or templateUrl, and its dependencies. The Component decorator is a key part of the Angular framework and is used extensively in Angular applications.

1. What are lifecycle hooks in Angular?

Explain a few lifecycle hooks. Lifecycle hooks in Angular are methods that are defined in components and are called at different stages of the component lifecycle. They provide an opportunity to perform specific actions at different points in time, such as initializing data, detecting changes, and cleaning up resources. Some common lifecycle hooks in Angular include ngOnInit, ngOnChanges, ngOnDestroy, and ngAfterViewInit.

1. What are router links?

Router links are a feature in Angular that allow for navigation between different components and routes in an application. They are defined as a directive in Angular and can be used in HTML templates to create links that navigate to different routes. Router links can be parameterized to pass data between components, and they can be used to navigate to child routes or to routes defined in a different module.

1. What exactly is the router state?

The router state in Angular refers to the current state of the application's routing system. It includes information about the current URL, the active route, and any parameters or data associated with the route. The router state can be accessed and manipulated using the Angular Router service, which provides a set of methods for navigating, retrieving information, and modifying the routing state.

1. What does Angular Material means?

Angular Material is a UI component library for Angular applications that provides a set of pre-built UI components, such as buttons, cards, forms, and data tables. It is designed to be easy to use and customizable, and it follows the Material Design guidelines from Google. Angular Material is widely used in Angular applications and provides a consistent and responsive user interface.

1. What is ngOnInit?

ngOnInit is a lifecycle hook in Angular that is called when a component is initialized and its data is bound. It is used to perform initialization tasks, such as retrieving data from a server, initializing variables, or subscribing to observables. ngOnInit is called once, after the component's constructor is called and before the first change detection cycle.

1. What is transpiling in Angular?

Transpiling in Angular is the process of converting TypeScript code to JavaScript code that can be executed by browsers or other JavaScript runtimes. The Angular CLI uses the TypeScript compiler to transpile Angular code, which allows for the use of modern TypeScript features such as classes, interfaces, and modules. The transpiled code is optimized for performance and compatibility with different browsers and platforms.