Angular Interview Practice (Sample)

Study Guide

BY EXAMZIFY

Everything you need from our exam experts!

Sample study guide. Visit https://angularinterview.examzify.com

Notice: Examzify makes every reasonable effort to obtain from reliable sources accurate, complete, and timely information about this product.

Questions

1. **What does the @ViewChild decorator allow you to do?**
   - **A. Access parent components only**
   - **B. Access child components, directives, or DOM elements**
   - **C. Define component styles**
   - **D. Manage component lifecycle events**

2. **What is the ngOnInit lifecycle hook used for?**
   - **A. To set up the routing for the application**
   - **B. To perform initialization logic after component creation**
   - **C. To handle data binding updates**
   - **D. To destroy the component safely**

3. **What does Angular Universal allow developers to do?**
   - **A. Optimize client-side rendering**
   - **B. Implement server-side rendering for Angular applications**
   - **C. Integrate with external REST APIs**
   - **D. Create static HTML pages only**

4. **Which library does Angular heavily use for handling asynchronous operations?**
   - **A. Promise.js**
   - **B. RxJS**
   - **C. AsyncAwait**
   - **D. Observable.js**

5. **How can you run unit tests in an Angular project?**
   - **A. ng test**
   - **B. npm run test**
   - **C. ng unit**
   - **D. npm test run**

6. **What is the main purpose of the DOM?**
   - **A. To store data in a database**
   - **B. To provide a structured representation of a document**
   - **C. To improve web page loading speed**
   - **D. To handle server-side operations**

7. Which directive is used for dynamically applying styles based on a condition?

A. ngFor

B. ngIf

C. ngStyle

D. ngModel

8. What is the function of the environment.ts file in an Angular project?

A. To manage application routes

B. To define configuration settings for different environments

C. To connect to external APIs

D. To store user preferences

9. What is the main advantage of using Angular CLI?

A. It ensures all Angular applications are identical

B. It simplifies the task of managing application dependencies

C. It provides a structured approach to building, testing, and deploying Angular applications

D. It eliminates the need for a backend server

10. What is the main feature of Ahead-of-Time (AOT) compilation in Angular?

A. It reduces the size of the code at runtime

B. It compiles code during the build process into efficient JavaScript

C. It provides real-time debugging

D. It changes the way components are loaded

Answers

1. **B**
2. **B**
3. **B**
4. **B**
5. **A**
6. **B**
7. **C**
8. **B**
9. **C**
10. **B**

Explanations

1. What does the @ViewChild decorator allow you to do?

A. Access parent components only

B. Access child components, directives, or DOM elements

C. Define component styles

D. Manage component lifecycle events

The @ViewChild decorator in Angular is a powerful feature that allows developers to access child components, directives, or DOM elements from a parent component. When you use @ViewChild, you can reference a specific component or directive in the template and then interact with it programmatically in the parent component class. For example, if there's a child component that exposes certain methods or properties, using @ViewChild enables the parent component to call those methods or directly modify its properties. This is particularly useful for implementing features that require direct manipulation of the child component, like calling a method to refresh data or enabling/disabling controls based on certain conditions. In addition to accessing components, @ViewChild can also be used to get a reference to DOM elements, which allows you to perform tasks like manipulating styles or listening to events directly without needing to bind everything through Angular's templated approach. By leveraging @ViewChild correctly, you enhance the interactivity and responsiveness of your application while maintaining clean and organized code structures within your components.

2. What is the ngOnInit lifecycle hook used for?

A. To set up the routing for the application

B. To perform initialization logic after component creation

C. To handle data binding updates

D. To destroy the component safely

The ngOnInit lifecycle hook is specifically designed for performing initialization logic after a component has been created. This hook is invoked shortly after the component is instantiated and is a suitable place for any setup you need to do, such as fetching data from a service, initializing properties, or subscribing to observables. Using ngOnInit ensures that your component’s dependencies are fully set up and ready to use. This is particularly useful since constructors are often used for dependency injection and setting up initial values, but they should not contain complex logic. Therefore, ngOnInit is the appropriate place to implement such logic, making sure that the component's view and its data are properly initialized and ready before rendering to the user. Additionally, while the other options do represent important functionalities within Angular, they either relate to areas outside of direct component initialization (like setting up routing) or to maintaining the component's state or lifecycle in other contexts (like handling data binding updates or safely destroying the component).

3. What does Angular Universal allow developers to do?

A. Optimize client-side rendering

B. Implement server-side rendering for Angular applications

C. Integrate with external REST APIs

D. Create static HTML pages only

Angular Universal enables developers to implement server-side rendering (SSR) for Angular applications. This feature allows pages to be rendered on the server rather than in the user’s browser, ensuring that users receive fully rendered pages more quickly. It enhances the performance of applications, particularly for initial page loads, and improves search engine optimization (SEO) since search engine crawlers can easily index the pre-rendered content. With server-side rendering, the application becomes more accessible and user-friendly, as users can see rendered content faster, reducing load time and increasing perceived performance. This improvement is especially beneficial for applications that require dynamic content to be available immediately. The other options, while related to aspects of web development, do not accurately describe the core functionality of Angular Universal. For example, client-side rendering is not the focus of Angular Universal; rather, it seeks to optimize how applications are served on the server side. Integrating with external REST APIs pertains to how Angular interacts with backend services, which is not the primary purpose of Angular Universal. Lastly, creating static HTML pages does not capture the dynamic rendering capabilities that Angular Universal provides, as it primarily focuses on rendering Angular applications server-side rather than generating static content.

4. Which library does Angular heavily use for handling asynchronous operations?

A. Promise.js

B. RxJS

C. AsyncAwait

D. Observable.js

Angular heavily relies on RxJS for handling asynchronous operations due to its powerful capability to work with streams of data, allowing developers to easily manage events and handle them reactively. RxJS provides a wide array of operators and utilities that facilitate complex asynchronous programming models, including the handling of multiple values over time, which is particularly useful for scenarios like HTTP requests, user input events, and other asynchronous tasks. Observables in RxJS enable Angular applications to respond to changes in data or input in a very efficient way. They allow for composing asynchronous operations with ease, and their support for combining multiple streams of data is particularly advantageous in building reactive interfaces. This aligns seamlessly with Angular's architecture, which promotes the use of reactive programming paradigms to handle state changes and side effects effectively. In contrast, while Promise.js and Async/Await are also methods for handling asynchronous operations in JavaScript, they do not offer the same level of flexibility and composition capabilities that Observables provide. They are more suited for handling single asynchronous events rather than complex data flows, which is essential in many Angular applications. Observable.js, on the other hand, is not a recognized standard library used by Angular.

5. How can you run unit tests in an Angular project?

A. ng test

B. npm run test

C. ng unit

D. npm test run

The correct way to run unit tests in an Angular project is by using the Angular CLI command "ng test". This command leverages the Angular testing framework, typically Jasmine, to execute the tests defined in the project. When you run "ng test", it automatically compiles the application in test mode and launches Karma, which is a test runner that facilitates running the tests in the browser. This command also initiates a watch mode by default, meaning that it will continuously monitor your files for changes and rerun tests as you develop, providing instant feedback. The other options, while they refer to methods of executing tests, do not specifically utilize the Angular CLI or the framework's conventions. For example, "npm run test" may also run tests if defined appropriately in the package.json file but doesn't explicitly link to the Angular CLI's orchestration of unit testing. Other options either represent non-existing commands in Angular or imply a different context that does not directly correspond to Angular's standardized testing process. Thus, understanding and utilizing "ng test" is fundamental for efficiently running unit tests in Angular applications.

6. What is the main purpose of the DOM?

A. To store data in a database

B. To provide a structured representation of a document

C. To improve web page loading speed

D. To handle server-side operations

The main purpose of the DOM, or Document Object Model, is to provide a structured representation of a document, typically HTML or XML. The DOM acts as an interface that browsers use to understand the hierarchy and structure of the web page, allowing developers to manipulate the content, structure, and style of the document dynamically using languages like JavaScript. When a web page is loaded, the browser creates a DOM tree that represents the elements of the page, such as headings, paragraphs, links, and other HTML tags. This tree structure allows for easy access and manipulation of the elements, enabling developers to add, remove, or modify content and attributes on the fly. This dynamic interaction enhances user experiences significantly since changes can occur based on user actions without requiring a full page reload. Understanding the role of the DOM is critical for web development, as it forms the foundation for client-side scripting and interactive features on modern websites. The other options presented do not accurately define the primary purpose of the DOM. While storing data in a database, improving web page loading speed, and handling server-side operations are important aspects of web development, they are not functions of the DOM itself. Instead, they pertain to back-end processes, optimization techniques, or external data management that interact with

7. Which directive is used for dynamically applying styles based on a condition?

A. ngFor

B. ngIf

C. ngStyle

D. ngModel

The choice of ngStyle is accurate because this directive is specifically designed to dynamically apply styles to HTML elements based on a condition or an expression evaluated in your Angular component. With ngStyle, you can bind an object where the keys are CSS property names and the values are the corresponding styles you wish to apply. This allows for easy manipulation of styles based on conditional logic, such as user interactions or component states. For instance, you might use ngStyle to change the background color of a button based on whether it is active or disabled. This dynamic approach gives developers the flexibility to create responsive and visually-rich interfaces. In contrast, ngFor is used for rendering lists by iterating through an array or collection, ngIf is intended for conditionally including or excluding elements from the DOM based on a boolean expression, and ngModel is primarily used for two-way data binding between form input elements and the component properties. Each of these directives serves different purposes in Angular, making ngStyle the right choice for conditional styling.

8. What is the function of the environment.ts file in an Angular project?

A. To manage application routes

B. To define configuration settings for different environments

C. To connect to external APIs

D. To store user preferences

The environment.ts file in an Angular project serves a crucial role in defining configuration settings for different environments, making option B the right choice. This file typically holds environment-specific variables such as API endpoints, feature toggles, and other settings that might differ between development, staging, and production environments.   For instance, you may have a different API URL in the development environment compared to the production environment. By managing these settings in the environment.ts file, you can easily switch between different configurations without the need to modify your application's core code. During the build process, the appropriate environment configuration can be selected (for example, using environment.prod.ts for production builds), thus ensuring that each deployment of the application is set up securely and efficiently according to the requirements of that specific context.   This organization promotes better maintainability and helps in avoiding hard-coded values throughout the application, enhancing its scalability and adaptability to different operational scenarios.

9. What is the main advantage of using Angular CLI?

- A. It ensures all Angular applications are identical
- B. It simplifies the task of managing application dependencies
- C. It provides a structured approach to building, testing, and deploying Angular applications
- D. It eliminates the need for a backend server

The main advantage of using Angular CLI is that it provides a structured approach to building, testing, and deploying Angular applications. The Angular CLI (Command Line Interface) is a powerful tool that streamlines the development process by offering predefined commands to generate components, services, and entire applications, which adheres to best practices. This structure facilitates consistency across projects, making it easier for developers to collaborate and maintain applications. It also includes features for scaffolding, which helps in automatically creating files and directories following Angular's conventions, thus reducing human error and saving time. Moreover, the CLI integrates tools for testing and deployment, allowing developers to run tests, build the application, and deploy it to production seamlessly. By utilizing Angular CLI, teams can focus more on writing code and implementing functionality rather than dealing with the intricacies of configuration and build processes, which is essential for efficient application development.

10. What is the main feature of Ahead-of-Time (AOT) compilation in Angular?

- A. It reduces the size of the code at runtime
- B. It compiles code during the build process into efficient JavaScript
- C. It provides real-time debugging
- D. It changes the way components are loaded

Ahead-of-Time (AOT) compilation is a powerful feature in Angular that compiles the application code during the build process rather than at runtime. This means that the Angular compiler converts the template and TypeScript code into efficient JavaScript before the application is served to the browser. By doing this compilation ahead of time, it helps in several ways. First, it results in faster rendering in the browser because the compiler's job is already done, reducing the amount of work the browser needs to do when the application is loaded. Second, early detection of template errors happens during the build time, which can save developers a significant amount of time during development. Finally, AOT compilation can optimize the application size since unnecessary code can be eliminated during the build process. This is why the key aspect of AOT compilation is its ability to efficiently compile code ahead of time, leading to improvements in performance and a more robust final build.