Forms - Reactive Forms

1. In reactive forms, what is the purpose of the FormBuilder service?

a) To create instances of form controls

b) To define form layouts

c) To manage form validation

d) To handle form submissions

2. How is form control validation handled in reactive forms?

a) Using HTML5 validation attributes

b) Using the ngValidate directive

c) Using the Validators class in Angular

d) By implementing a custom validation function

3. What is the difference between reactive forms and template-driven forms in Angular?

a) Reactive forms use two-way data binding, while template-driven forms use one-way data binding

b) Reactive forms are synchronous, while template-driven forms are asynchronous

c) Reactive forms are driven by data streams, while template-driven forms are driven by HTML templates

d) There is no significant difference; they are just different approaches to achieve the same goal

4. How can you dynamically add form controls in reactive forms?

a) Using the [formGroup] directive

b) Using the addControl method of the FormControl class

c) Using the FormArray class

d) By directly manipulating the HTML template

5. What is the purpose of the patchValue method in reactive forms?

a) To set the value of a form control

b) To validate a form

c) To reset a form

d) To partially update the values of a form

6. How do you handle complex form structures with nested form groups in reactive forms?

a) Using the Validators class

b) Using the FormArray class

c) Using the FormControl class

d) Using the FormBuilder service

Template-Driven Forms

7. In template-driven forms, how are form controls created in the HTML template?

a) Using the FormControl directive

b) Using the formControl attribute

c) Using the ngModel directive

d) Using the ngControl directive

8. What is the purpose of the ngForm directive in template-driven forms?

a) To define a form layout

b) To create form controls

c) To manage form validation

d) To represent the form itself and give access to its controls

9. How is form validation handled in template-driven forms?

a) Using the Validators class

b) Using HTML5 validation attributes

c) By implementing a custom validation function in TypeScript

d) By using the ngValidate directive

10. How can you disable a submit button until a form is valid in template-driven forms?

a) Using the [disabled] attribute with a condition

b) Using the ngDisabled directive

c) By handling the ngSubmit event

d) By adding the disabled class to the button

Angular Pipes for Transforming Output

11. What is the purpose of the async pipe in Angular when working with HTTP requests?

a) To handle asynchronous operations in templates

b) To define an asynchronous function

c) To create a new async object

d) To handle errors in asynchronous code

12. How is a custom pipe created in Angular?

a) By using the ngPipe command in the Angular CLI

b) By creating a TypeScript class decorated with @Pipe

c) By using the pipe method in the @NgModule decorator

d) By implementing a function in the component and using it in the template

13. What is the purpose of the DatePipe in Angular?

a) To format dates

b) To filter data based on dates

c) To perform arithmetic operations on dates

d) To create date objects

14. How can you chain multiple pipes in Angular?

a) Using the | symbol

b) Using the pipe method

c) Using the => symbol

d) By concatenating pipe names

15. What is the primary advantage of using pipes in Angular?

a) To handle form submissions

b) To improve performance by reducing the amount of logic in templates

c) To create dynamic templates

d) To manage routing in the application

HTTP Requests in Angular

16. How is the HttpClient module used for making HTTP requests in Angular?

a) By importing it from the @angular/core module

b) By injecting it into the component constructor

c) By using the Http class

d) By adding it to the providers array in @NgModule

17. What is the purpose of the get method in the HttpClient module?

a) To send a GET request to the server

b) To retrieve data from the local storage

c) To define a new route

d) To create a new instance of the HttpClient

18. How can you handle errors in HTTP requests using the HttpClient module?

a) By using the catchError operator

b) By using the try...catch statement

c) By adding a try-catch block in the component

d) By handling the error event

19. What is the purpose of the subscribe method when making HTTP requests?

a) To define the HTTP method (GET, POST, etc.)

b) To handle asynchronous responses and errors

c) To send the HTTP request

d) To set headers for the HTTP request

20. How can you send data in the body of an HTTP POST request using the HttpClient module?

a) By using the params property

b) By using the body property

c) By adding data directly to the URL

d) By using the data property

Services and Dependency Injection

21. What is a service in Angular?

a) A class with a specific decorator

b) A function that returns an observable

c) A reusable component

d) A class with a @Service decorator

22. How is a service typically provided in Angular?

a) By adding it to the providers array in the @NgModule decorator

b) By creating a separate module for each service

c) By using the @Service decorator in the component

d) By directly instantiating it in the component

23. What is the purpose of dependency injection in Angular?

a) To manage the dependencies between components

b) To inject services into components, directives, and other services

c) To create instances of classes

d) To handle asynchronous operations

24. How can you create a singleton service in Angular?

a) By using the @Singleton decorator

b) By setting the providedIn property to 'root'

c) By adding it to the providers array in a specific module

d) By using the @Injectable decorator with a specific option

25. What is the role of the Injector in Angular's dependency injection system?

a) To create instances of classes

b) To manage the dependencies between components

c) To inject services into components

d) To provide a centralized registry for all services

Advanced Topics - Forms and HTTP Requests

26. How can you implement file uploads in Angular forms?

a) By using the file input type and handling it in the component

b) By using the HttpInterceptor for intercepting file uploads

c) By using the FileReader class

d) By using the @angular/forms module

27. How can you handle authentication in HTTP requests in Angular?

a) By using the Authorization header

b) By sending authentication data as URL parameters

c) By using the HttpClient module's built-in authentication methods

d) By implementing a custom authentication service

28. What is CORS, and how does it relate to HTTP requests in Angular?

a) CORS is a type of HTTP request method

b) CORS stands for Centralized Origin Request System

c) CORS is a security feature implemented by browsers to restrict web pages from making requests to a different domain

d) CORS is a type of authentication token used in HTTP requests

29. How can you handle long-polling or server-sent events in Angular?

a) By using the EventSource class

b) By implementing a custom observable

c) By using the HttpClient module's sse property

d) By handling the poll event in the component

30. What is the purpose of the FormArray class in reactive forms?

a) To create an array of form controls

b) To manage form submissions

c) To define a new form layout

d) To handle asynchronous operations in forms

Error Handling and Debugging

31. How can you debug HTTP requests in Angular?

a) By using the browser's developer tools

b) By adding console.log statements in the component

c) By using the debugger statement in the code

d) By using the Angular CLI's built-in debugging tool

32. What is the purpose of the finalize operator in HTTP requests?

a) To handle errors in HTTP requests

b) To execute a piece of code after the HTTP request completes, regardless of success or failure

c) To log information about the HTTP request

d) To add a final step in the HTTP request pipeline

33. How can you handle global errors in Angular?

a) By using the ErrorHandler service

b) By using the @ErrorHandler decorator

c) By adding a global error handler function in the AppModule

d) By using the window.onerror event

34. How can you intercept and modify HTTP requests globally in Angular?

a) By using the HttpInterceptor interface

b) By using the @Intercept decorator

c) By adding a middleware function to the providers array in the @NgModule decorator

d) By modifying the Http class directly

35. What is the purpose of the retry operator in HTTP requests?

a) To retry the HTTP request a specific number of times in case of failure

b) To handle errors in HTTP requests

c) To log information about the HTTP request

d) To automatically refresh the page after a successful HTTP request

Observables and Reactive Programming

36. How can you handle multiple HTTP requests sequentially in Angular?

a) By using the mergeMap operator

b) By using the forkJoin operator

c) By using the switchMap operator

d) By using the concatMap operator

37. What is the purpose of the map operator in reactive programming?

a) To create a new observable

b) To transform the emitted values from the source observable

c) To merge multiple observables

d) To filter values in an observable

38. How can you debounce user input in Angular forms?

a) By using the debounce operator

b) By using the throttle operator

c) By using the delay operator

d) By adding a timeout in the component

39. What is the difference between forkJoin and combineLatest operators in reactive programming?

a) forkJoin emits the last value from each observable, while combineLatest emits every time any observable emits a value

b) forkJoin emits a single value with an array of the last values from each observable, while combineLatest emits each time any observable emits a value

c) forkJoin waits for all observables to complete before emitting a single value, while combineLatest emits values whenever any observable emits

d) There is no significant difference; they can be used interchangeably

40. How can you handle race conditions in Angular using observables?

a) By using the race operator

b) By using the switchMap operator

c) By using the retry operator

d) By using the debounce operator

Angular Testing - Forms and HTTP Requests

41. How can you test a component with a reactive form in Angular?

a) By using the TestBed and the By.directive selector

b) By using the TestBed and the By.css selector

c) By using the TestBed and the By.directive selector with a custom form control directive

d) By directly accessing the form controls in the component

42. What is the purpose of the TestBed in Angular testing?

a) To test HTTP requests

b) To configure and create an Angular testing module

c) To simulate user interactions

d) To define test cases

43. How can you mock HTTP requests in Angular testing?

a) By using the HttpClientTestingModule

b) By using the HttpTestingController

c) By using the TestBed with a custom HTTP service

d) By overriding the HttpClient class in the test

44. What is the purpose of the flush method in Angular HTTP testing?

a) To simulate a delay in the HTTP response

b) To complete the observable and emit a value immediately

c) To simulate a network error

d) To reset the HTTP testing controller

45. How can you test a form with asynchronous validation in Angular?

a) By using the async function in the test

b) By using the fakeAsync function in the test

c) By using the TestBed with a custom asynchronous validator

d) By directly calling the asynchronous validator function in the test

Advanced Angular Topics

46. What is the purpose of the ngContainer directive in Angular templates?

a) To create a container for form controls

b) To create a container for Angular components without adding an additional element to the DOM

c) To define a new layout for a form

d) To manage dependency injection in templates

47. How can you implement infinite scrolling in Angular?

a) By using the *ngIf directive

b) By using the *ngFor directive

c) By handling the scroll event and making additional

ANSWERS

Angular Basics

c) A web development framework

a) A superset of JavaScript

b) Client-side scripting

a) Two-way data binding

c) npm install -g @angular/cli

a) Module

c) A collection of components, directives, and services

c) To organize the application into logical units

c) ngIf

a) Binds a style property to an expression

TypeScript Basics

a) A superset of JavaScript

a) Declares a variable

c) To enforce a contract on a class

a) abstract class MyClass {}

b) To inherit from a class

Angular Components and Directives

d) Using property binding

c) To conditionally render content

c) By Angular itself

a) Executes after the component is created

b) To style an element based on an expression

b) ngFor

a) To add or remove CSS classes based on an expression

Angular Data Binding

b) Binding a variable's value to a template

a) [property]

b) To handle user input or interactions

b) (event)

a) One-way binding

c) To bind a variable to user input

d) [(data)]

a) It contains information about the event

Angular Architecture

a) To define the root component

c) By using decorators and TypeScript

c) To manage database connections

b) To define services and inject them into components

c) Using the RouterModule and @NgModule

Advanced Angular Concepts

b) Loading modules on-demand, improving application performance

a) Compiling TypeScript code before runtime

b) To encapsulate business logic and share data between components

c) Using services

a) To handle asynchronous operations in templates

TypeScript Advanced Concepts

a) A type representing an empty set of values

a) It makes a property or variable immutable

a) A type that can be used for any data type

b) A type representing unknown or uninitialized values

a) To create classes

Angular Testing

b) To configure and create an Angular testing module

b) Testing individual units or components in isolation

b) Automatically injecting dependencies using the TestBed

c) To write and run tests

b) Testing the entire application's workflow

Angular CLI

c) ng new project-name

b) To generate components, services, and more

d) ng generate component my-component

b) ng serve

b) To generate a production build

Angular Forms

c) Using ngModel directive and HTML5 validation attributes

b) To handle form submission events

d) Forms that are driven by data streams and synchronized with the component

b) As TypeScript classes

a) To build and configure form controls

Template-Driven Forms

b) Using the [expression | pipe] syntax

a) {{ text | uppercase }}

a) To format dates

a) {{ text | uppercase }}

a) Using the [expression] syntax

Angular Pipes for Transforming Output

a) To conditionally render content based on multiple conditions

a) To handle asynchronous operations in templates

b) By creating a TypeScript class decorated with @Pipe

a) To format dates

a) {{ text | uppercase }}

HTTP Requests in Angular

c) By using the HttpClientTestingModule

b) To inject it into the component constructor

a) To send a GET request to the server

a) To validate a form

b) To handle asynchronous responses and errors

Services and Dependency Injection

a) A class with a specific decorator

a) By adding it to the providers array in the @NgModule decorator

b) To inject services into components, directives, and other services

b) By setting the providedIn property to 'root'

d) To provide a centralized registry for all services

Advanced Topics - Forms and HTTP Requests

a) By using the ngPipe command in the Angular CLI

a) By using the catchError operator

b) By using the TestBed with a custom asynchronous validator

b) By using the HttpTestingController

b) By using the forkJoin operator

Error Handling and Debugging

a) By using the browser's developer tools

b) To execute a piece of code after the HTTP request completes, regardless of success or failure

c) By adding a global error handler function in the AppModule

a) By using the race operator

a) To handle errors in HTTP requests

Observables and Reactive Programming

b) By using the mergeMap operator

b) To transform the emitted values from the source observable

b) By using the throttle operator

b) forkJoin emits a single value with an array of the last values from each observable, while combineLatest emits each time any observable emits a value

a) By using the race operator

Angular Testing - Forms and HTTP Requests

b) By using the TestBed and the By.css selector

b) To configure and create an Angular testing module

a) By using the HttpClientTestingModule

b) To complete the observable and emit a value immediately

a) By using the async function in the test