1. Why were client-side frameworks like Angular introduced?

Ans : Back in the day, web developers used VanillaJS and jQuery to develop dynamic websites but, as the logic of one's website grew, the code became more and more tedious to maintain. For applications that use complex logic, developers had to put in extra effort to maintain separation of concerns for the app. Also, jQuery did not provide facilities for data handling across views.For tackling the above problems, client-side frameworks like Angular came into the picture, which made life easier for the developers by handling separation of concerns and dividing code into smaller bits of information (In the case of Angular, called Components).

Client-side frameworks allow one to develop advanced web applications like Single-Page-Application. Not that we cannot develop SPAs using VanillaJS, but by doing so, the development process becomes slower.

2. How does an Angular application work?

Ans : Every Angular app consists of a file named angular.json. This file will contain all the configurations of the app. While building the app, the builder looks at this file to find the entry point of the application. Following is an image of the angular.json file.

3. What are some of the advantages of Angular over other frameworks?

Ans : Features that are provided out of the box - Angular provides a number of built-in features like,routing, state management, rxjs library and http servicesstraight out of the box. This means that one does not need tolook for the above stated features separately. They are allprovided with angular.

Declarative UI - Angular uses HTML to render the UI of an application. HTML isa declarative language and is much easier to use than JavaScript.

Long-term Google support - Google announced Long-term support for Angular. This means that Google plans to stick with Angular and further scale up its ecosystem.

4. List out differences between AngularJS and Angular

Ans : Architecture

AngularJS uses MVC or Model-View-Controller architecture, where the Model contains the business logic, Controller processes information and View shows the information present in the Model.

Angular replaces controllers with Components. Components are nothing but directives with a predefined template.

Language

AngularJS uses JavaScript language, which is a dynamically typed language.

Angular uses TypeScript language, which is a statically typed language and is a superset of JavaScript. By using statically typed language, Angular provides better performance while developing larger applications.

Mobile Support

AngularJS does not provide mobile support.

Angular is supported by all popular mobile browsers.

Structure

While developing larger applications, the process of maintaining code becomes tedious in the case of AngularJS.

In the case of Angular, it is easier to maintain code for larger applications as it provides a better structure.

Expression Syntax

While developing an AngularJS application, a developer needs to remember the correct ng-directive for binding an event, or a property. Whereas in Angular, property binding is done using "[ ]" attribute and event binding is done using "( )" attribute.

6. Explain string interpolation and property binding in Angular.

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

Data-binding is a feature in angular, which provides a way to communicate between the component(Model) and its view(HTML template).

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

In Angular, data from the component can be inserted inside the HTML template. In one-way binding, any changes in the component will directly reflect inside the HTML template but, vice-versa is not possible. Whereas, it is possible in two-way binding.

String interpolation and property binding allow only one-way data binding.

String interpolation uses the double curly braces {{ }} to display data from the component. Angular automatically runs the expression written inside the curly braces, for example, {{ 2 + 2 }} will be evaluated by Angular and the output 4, will be displayed inside the HTML template. Using property binding, we can bind the DOM properties of an HTML element to a component's property. Property binding uses the square brackets [ ] syntax.

7. 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.

8. 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. Conversely, any changes in the model state are reflected in the UI state. This allows the framework to connect the DOM to the Model data via the controller. However, this approach affects performance since every change in the DOM has to be tracked.

9. 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.

10. 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

11. 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.

12. 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.

13. 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

14. 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

15. What are Components in Angular?

Components Heirarchy

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.

16. What are Pipes in Angular?

angular pipes

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.

17. What are Pipes in Angular?

angular pipes

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.

18. 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.

19. 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.

20. What are controllers?

AngularJS controllers control the data of AngularJS applications. They are regular JavaScript Objects. The ng-controller directive defines the application controller.

21. 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.

22. 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 }}

23. What are Template statements?

Template statements are properties or methods used in HTML for responding to user events. With these template statements, the application that you create or are working on, can have the capability to engage users through actions such as submitting forms and displaying dynamic content.

For example,

<button (click)="deleteHero()">Delete hero</button>

The template here is deleteHero. The method is called when the user clicks on the button.

24. 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.

25. What is Angular Framework?

Angular is a TypeScript-based open-source front-end platform that makes it easy to build applications with in web/mobile/desktop. The major features of this framework such as declarative templates, dependency injection, end to end tooling, and many more other features are used to ease the development.

26. What are the key components of Angular?

Angular has the below key components,

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.

Services: It is used to create components which can be shared across the entire application.

Metadata: This can be used to add more data to an Angular class.

27. What are components?

Components are the most basic UI building block of an Angular app which formed a tree of Angular components. These components are subset of directives. Unlike directives, components always have a template and only one component can be instantiated per an element in a template. Let's see a simple example of Angular component

28. What are the differences between Component and Directive?

In a short note, A component(@component) is a directive-with-a-template.

Some of the major differences are mentioned in a tabular form

Component Directive

To register a component we use @Component meta-data annotation To register directives we use @Directive meta-data annotation

Components are typically used to create UI widgets Directive is used to add behavior to an existing DOM element

Component is used to break up the application into smaller components Directive is use to design re-usable components

Only one component can be present per DOM element Many directives can be used per DOM element

@View decorator or templateurl/template are mandatory Directive doesn't use View

29. 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,

The NgModule decorator has five important(among all) options

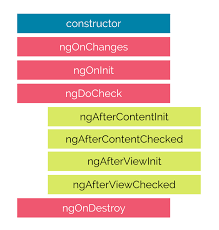
The imports option is used to import other dependent modules. The BrowserModule is required by default for any web based angular application

The declarations option is used to define components in the respective module

The bootstrap option tells Angular which Component to bootstrap in the application

The providers option is used to configure set of injectable objects that are available in the injector of this module.

The entryComponents option is a set of components dynamically loaded into the view.

30 What are lifecycle hooks available?  
Angular application goes through an entire set of processes or has a lifecycle right from its initiation to the end of the application. The representation of lifecycle in pictorial representation as follows,  
  
The description of each lifecycle method is as below,

* 1. ngOnChanges: When the value of a data bound property changes, then this method is called.
  2. ngOnInit: This is called whenever the initialization of the directive/component after Angular first displays the data-bound properties happens.
  3. ngDoCheck: This is for the detection and to act on changes that Angular can't or won't detect on its own.
  4. ngAfterContentInit: This is called in response after Angular projects external content into the component's view.
  5. ngAfterContentChecked: This is called in response after Angular checks the content projected into the component.
  6. ngAfterViewInit: This is called in response after Angular initializes the component's views and child views.
  7. ngAfterViewChecked: This is called in response after Angular checks the component's views and child views.
  8. ngOnDestroy: This is the cleanup phase just before Angular destroys the directive/component.

31. What is a data binding?

Data binding is a core concept in Angular and allows to define communication between a component and the DOM, making it very easy to define interactive applications without worrying about pushing and pulling data. There are four forms of data binding(divided as 3 categories) which differ in the way the data is flowing.

From the Component to the DOM:

Interpolation: {{ value }}: Adds the value of a property from the component

<li>Name: {{ user.name }}</li>

<li>Address: {{ user.address }}</li>

Property binding: [property]=”value”: The value is passed from the component to the specified property or simple HTML attribute

<input type="email" [value]="user.email">

From the DOM to the Component: Event binding: (event)=”function”: When a specific DOM event happens (eg.: click, change, keyup), call the specified method in the component

<button (click)="logout()"></button>

Two-way binding: Two-way data binding: [(ngModel)]=”value”: Two-way data binding allows to have the data flow both ways. For example, in the below code snippet, both the email DOM input and component email property are in sync

<input type="email" [(ngModel)]="user.email">

33. What is the difference between constructor and ngOnInit?

TypeScript classes has a default method called constructor which is normally used for the initialization purpose. Whereas ngOnInit method is specific to Angular, especially used to define Angular bindings. Even though constructor getting called first, it is preferred to move all of your Angular bindings to ngOnInit method. In order to use ngOnInit, you need to implement OnInit interface as below,

34. What is a service?

A service is used when a common functionality needs to be provided to various modules. Services allow for greater separation of concerns for your application and better modularity by allowing you to extract common functionality out of components.

Let's create a repoService which can be used across components,

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

import { Http } from '@angular/http';

@Injectable({ // The Injectable decorator is required for dependency injection to work

// providedIn option registers the service with a specific NgModule

providedIn: 'root', // This declares the service with the root app (AppModule)

})

export class RepoService{

constructor(private http: Http){

}

fetchAll(){

return this.http.get('https://api.github.com/repositories');

}

}

The above service uses Http service as a dependency.

35. What is interpolation?

Interpolation is a special syntax that Angular converts into property binding. It’s a convenient alternative to property binding. It is represented by double curly braces({{}}). The text between the braces is often the name of a component property. Angular replaces that name with the string value of the corresponding component property.

36.What are pipes?

A pipe takes in data as input and transforms it to a desired output. For example, let us take a pipe to transform a component's birthday property into a human-friendly date using date pipe.

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

@Component({

selector: 'app-birthday',

template: `<p>Birthday is {{ birthday | date }}</p>`

})

export class BirthdayComponent {

birthday = new Date(1987, 6, 18); // June 18, 1987

}

36.What are observables?

Observables are declarative which provide support for passing messages between publishers and subscribers in your application. They are mainly used for event handling, asynchronous programming, and handling multiple values. In this case, you define a function for publishing values, but it is not executed until a consumer subscribes to it. The subscribed consumer then receives notifications until the function completes, or until they unsubscribe.

37. 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,