Angular 5

Introduction to Web Technology

Web development has come a long way in delivering functionalities online. There are tremendous changes happening both on the client-side technologies (like JS/Angular) and the server side technologies (like Java/Spring). Not only AngularJS, similar technologies like React JS and Vue JS are offering great scope in client development. Let us see what Angular has to offer in client development.

What is Angular and Why Angular?

Angular (2/4/5) is not a programming language like Javascript or Typescript. Angular is a front-end or client-side Framework, which needs a [programming](https://www.greycampus.com/programming) language like Typescript (developed by Microsoft). Now, what does it mean when one says, "Angular is a client-side framework"? It means that it runs on the client-side or user's browser and not on a Web Server (where java/python/ruby/vb.net run). Of course, there is the Angular Universal which allows Angular to run on a server; but, primarily, angular is a client-side framework. Angular is a product developed and maintained by the techie giant Google and has adopted the SPA (Single Page Application) principles.

Angular Interview Questions and Answers

With the introduction of these technologies, web development has catalyzed the generation of jobs like Front-end Developer or User Interface Developer. A newbie or an experienced developer can make a flourishing career by understanding any of the front-end technologies. Here are some of the questions that are frequently asked for any UI/Front-end Developer role.

FAQ on Angular

1. What’s new in Angular 7?

Certain tools are optimized in the new version of [Angular](https://www.greycampus.com/angularjs-training-instructor-led), let us see what the tools are:

* Angular 7 supports Typescript version 3.1
* Angular 5 supports RxJS 6.3.
* New ng compiler “Angular compatibility compiler” added in angular 7. Massive reduction in bundle size with ivy compliant format.
* CLI Prompt to choose routing and SCSS.
* Virtual scrolling and Drag drop module has been added into Angular Material CDK (Component Dev Kit)
* Angular “Do-Bootstrap” life cycle hook added
* Application performance become faster as removed polyfills file from production mode.

2. Name the building blocks of Angular.

The Angular application is made using the following:

* Modules
* Component
* Template
* Directives
* Data Binding
* Services
* Dependency Injection
* Routing

3. What is Transpiling in Angular?

Transpiling is the process of converting the typescript into javascript (using Traceur, a JS compiler). Though typescript is used to write code in the Angular applications, the code is internally transpiled into javascript.

4. Which of the Angular life cycle component execution happens when a data-bound input property value updates?

ngOnChanges is the life cycle hook that gets executed whenever a change happens to the data that was bound to an input.

5. Differentiate between Components and Directives in Angular 5.

Components break up the application into smaller parts; whereas, Directives add behavior to an existing DOM element.

6. What is the use of @Input and @Output?

When it comes to the communication of Angular Components, which are in Parent-Child Relationship; we use @Input in Child Component when we are passing data from Parent to Child Component and @Output is used in Child Component to receive an event from Child to Parent Component.

7. What is ng-content Directive / what is content projection in angular?

The HTML elements like p (paragraph) or h1 (heading) have some content between the tags. For example, <p>this is a paragraph</p> and <h1>this is a heading</h1>. Now, similar to this, what if we want to have some custom text or content between the angular tags like <app-tax>some tax-related content</app-tax> this will not work the way it worked for HTML elements.  Now, in such cases, the <ng-content> tag directive is used.

OR

Content projection allows you to insert a shadow DOM in your component. It means, if you want to insert HTML elements or other components in a component, then we can do that using the concept of content projection. In Angular, we achieve content projection using **<ng-content> </ng-content>.**We can make reusable components and scalable applications by properly using content projection.

Using the **@Input()** decorator, we can pass a simple string to the child components, but what if you need to pass different types of data to the child component then content projection is used such as:

1. Inner HTML
2. HTML Elements
3. Styled HTML
4. Another Component, etc.

**Note**: “select” option of ng-content is used for targeted content projection

8. What **Shadow DOM**?

**Shadow DOM** is like a parallel **DOM** tree hosted inside a component (an HTML element, not to be confused with **Angular** components), hidden away from the main **DOM** tree. It is the implementation of the **Shadow DOM** standard that allows view, style, and behavior encapsulation.

OR

**Shadow DOM** refers to a subtree of **DOM** elements that renders as part of the document, but not into the main document **DOM** tree.

**Note**: Normal content of the DOM tree are known as **LIGHT DOM**

9. What is **<ng-template>** in angular

The **<ng-template>** is an Angular element for rendering HTML. It is never displayed directly. In fact, before rendering the view, Angular *replaces* the <ng-template> and its contents with a comment.

9. What is **<ng-container>** in angular

The Angular <ng-container> is a grouping element that **doesn't interfere with styles** or layout because Angular doesn't put it in the DOM.

The <ng-container> is a syntax element recognized by the Angular parser. It's not a directive, component, class, or interface. It’s a logical container rendered as HTML comment.

9. When to use <ng-template> and <ng-container>

* [<ng-container>](https://angular.io/guide/structural-directives#ngcontainer) if you need a multiple nested structural directives like [\*ngIf](https://angular.io/api/common/NgIf) or [\*ngFor](https://angular.io/api/common/NgForOf) or if you want to wrap more than one element inside such a structural directive;

If you want to apply more than one structural directive like ngFor and ngIf which is not supported, then you can use the explicit form for one of these.

Eg. Instead of this invalid syntax

<div \*ngFor="let item of items" \*ngIf="!item.hidden"></div>

you can use

<ng-container \*ngFor="let item of items">

<div \*ngIf="!itemHidden></div>

</ng-container>

* <ng-template> if you need a view "snippet" that you want to stamp at various places(reuse) using  [ngTemplateOutlet](https://angular.io/api/common/NgTemplateOutlet).

Note: The Angular <ng-container> is a grouping element that doesn't interfere with styles or layout because Angular *doesn't put it in the DOM*.

The <ng-template> is an Angular element for rendering HTML. It is never displayed directly. In fact, before rendering the view, Angular *replaces* the <ng-template> and its contents with a comment.

9. What does a router.navigate do?

When we want to route to a component we use router.navigate.  Syntax: this.router.navigate([‘/component\_name’]);

9. What is ViewEncapsulation?

ViewEncapsulation decides whether the styles defined in a component can affect the entire application or not. There are three ways to do this in Angular:

Emulated 0: Styles from other HTML/component/global (bootstrap css) spread/bleed to the current component and style from current component also used in current component, but style defined in current component does not spread/bleed to other html/components.

Native 1: styles from current HTML/component does not spread to the other HTML/Component where native is configured. E.g bootstrap style or other component style will not be inherit into the current component where native is configured

None 2: styles defined in an html/component are visible to all html/components. Style defined in component “A” and “B” applied to each other.

Note: Emulated & Native does not spread to other component.

ShadowDom 3:

10. What are Services in Angular and what command is used to create a service?

Services help us in not repeating the code. With the creation of services, we can use the same code from different components. Here is the command to create a service in angular, ng g service User (a UserService is created when this command is used).

11. What is Dependency Injection in Angular 4?

When a component is dependent on another component the dependency is injected/provided during runtime.

12. What is Routing in Angular 5?

Routing helps a user in navigating to different pages using links.

13. How to handle Events in Angular 5?

Any activity (button click, mouse click, mouse hover, mouse move, etc) of a user on a frontend/web screen is termed as an event. Such events are passed from the view (.HTML) page to a typescript component (.ts).

14. What is a RouterOutlet?

RouterOutlet is a substitution for templates rendering the components. In other words, it represents or renders the components on a template at a particular location.

15. Explain the usage of {{}}?

The set of brackets {{}} when used with an HTML tag, represent data from a component. For example, on a HTML page which has <h1>{{variableName}}</h1>, here the ‘variableName’ is actually typescript (component) data representing its value on the template; i.e., HTML. This entire concept is called String Interpolation.

16. In how many ways the Data Binding can be done?

Data Binding happens between the HTML (template) and typescript (component). Data binding can be done in 3 ways:

(i) Property Binding (ii) Event Binding (iii) Two-Way Data Binding.

17. What is the sequence of Angular Lifecycle Hooks?

OnChange()  -  OnInit()  -  DoCheck()  -  AfterContentInit()  -  AfterContentChecked()  -  AfterViewInit()  -  AfterViewChecked()  -  OnDestroy().

18. What is the purpose of using package.json in the angular project?

With the existence of package.json, it will be easy to manage the dependencies of the project. If we are using typescript in the angular project then we can mention the typescript package and version of typescript in package.json.

19. How is SPA (Single Page Application) technology different from the traditional web technology?

In traditional web technology, the client requests for a web page (HTML/JSP/asp) and the server sends the resource (or HTML page), and the client again requests for another page and the server responds with another resource. The problem here is a lot of time is consumed in the requesting/responding or due to a lot of reloading. Whereas, in the SPA technology, we maintain only one page (index.HTML) even though the URL keeps on changing.

20. What is Component in Angular Terminology?

A web page in Angular has many components involved in it. A Component is basically a block in which the data can be displayed on HTML using some logic usually written in typescript.

21. What are ngModel and how do we represent it?

ngModel is a directive which can be applied on a text field. This a two-way data binding. ngModel is represented by [()]

22. What does a Subscribe method do in Angular 4?

It is a method which is subscribed to an observable. Whenever the subscribe method is called, an independent execution of the observable happens.

23. Differentiate between Observables and Promises.

Observables are lazy, which means nothing happens until a subscription is made. Whereas Promises are eager; which means as soon as a promise is created, the execution takes place.

Observable is a stream in which passing of zero or more events is possible and the callback is called for each event. Whereas, promise handles a single event.

**Promises**

1. Definition : Helps you run functions asynchronously, and use their return values (or exceptions) **only once** when executed.
2. Not Lazy
3. Not cancellable. The two possible decisions are
   * Reject
   * Resolve
4. Cannot be **retried**(Promises should have access to the original function that returned the promise in order to have a retry capability, which is a bad practice)

**Observables**

1. Definition : Helps you run functions asynchronously, and use their return values in a continuous sequence(**multiple times**) when executed.
2. By default, it is Lazy as it emits values when time progresses.
3. Has a lot of operator which simplifies coding effort.
4. One operator **retry** can be used to retry whenever needed, also if we need to retry the observable based on some conditions **retryWhen** can be used.

24. What is an AsyncPipe in Angular?

When an observable or promise returns something, we use a temporary property to hold the content. Later, we bind the same content to the template.

But with the usage of AsyncPipe, the promise or observable can be directly used in a template and a temporary property is not required.

25. Explain Authentication and Authorization.

Authentication: The user login credentials are passed to an authenticate API (on the server). On the server side validation of the credentials happens and a JSON Web Token (JWT) is returned. JWT is a JSON object that has some information or attributes about the current user.  Once the JWT is given to the client, the client or the user will be identified with that JWT.

Authorization: After logging in successfully, the authenticated or genuine user does not have access to everything. The user is not authorized to access someone else’s data, he/she is authorized to access some data.

26. What is AOT Compilation?

Every angular application gets compiled internally. The angular compiler takes type script code, compiles it and produces javascript code again. Ahead-of-Time Compilation does not happen every time or for every user, as is the case with Just-In-Time (JIT) Compilation.

OR

The Angular Ahead-of-Time (AOT) compiler converts your Angular HTML and TypeScript code into efficient JavaScript code during the build phase *before* the browser downloads and runs that code. Compiling your application during the build process provides a faster rendering in the browser.

Angular offers two ways to compile your application:

1. *Just-in-Time* (JIT), which compiles your app in the browser at runtime.
2. *Ahead-of-Time* (AOT), which compiles your app at build time.

JIT compilation is the default when you run the [ng build](https://angular.io/cli/build) (build only) or [ng serve](https://angular.io/cli/serve) (build and serve locally) CLI commands:

For AOT compilation, include the --aot option with the ng build or ng serve command:

27. What is Redux?

It is a library which helps us maintain the state of the application. Redux is not required in applications that are simple with the simple data flow, it is used in Single Page Applications that have complex data flow.

28. What are Pipes?

This feature is used to change the output on the template; something like changing the string into uppercase and displaying it on the template. It can also change Date format accordingly.

29. Differentiate between ng-Class and ng-Style.

In ng-Class, loading of CSS class is possible; whereas, in ng-Style we can set the CSS style.

30. Why Typescript with Angular?

Typescript is a superset of Javascript. Earlier, Javascript was the only client side language supported by all browsers. But, the problem with Javascript is, it is not a pure Object Oriented Programming Language. The code written in JS without following any pattern so code becomes messy and finally leading to difficulties in maintainability and reusability. Instead of learning concepts (like patterns) to maintain code, programmers prefer to maintain the code in a OOP approach and is made available with a programming language like Typescript was thus developed by Microsoft in a way that it can work as Javascript and also offer what javascript cannot ie;

* Pure OOPS as Typescript offers concepts like Generics, Interfaces and Types (a Static Typed Language) which makes it is easier to catch incorrect data types passing to variables.
* TS provides flexibility to programmers experienced in java, .net as it offers encapsulation through classes and interfaces.
* JS version ES5 offers features like Constructor Function, Dynamic Types, Prototypes.
* The next version of Javascript ie ES6 introduced new feature like Class keyword but not supported by many browsers.
* TS offers Arrow Functions (=>) which is an ES6 feature not supported by many browsers directly but when used in TS, gets compiled into JS ES5 and runs in any browser.
* TS is not the only alternative to JS, we have CoffeScript, Dart(Google).
* Finally, it is like, TS makes life easier when compared to JS.

31. What is pure and impure pipe in Angular?

A pure pipe is only called when Angular detects a change in the value or the parameters passed to a pipe.

An impure pipe is called for every change detection cycle no matter whether the value of the data bind property or parameters is changed.

<p> {{'Hello' | translate }}<p>

<p> {{'World' | translate }}<p>

* If pipe is pure: there will be only one instance of the pipe. The transform method will be called twice but on the same instance.
* If pipe is impure: there will be two instances of the pipe.

@Pipe({

name: 'sort',

pure: true | false // true makes it pure and false makes it impure

})

export class myPipe implements PipeTransform {

transform(value: any, args?: any): any {

// your logic here and return the result

}

}

32. What is decorators in angular

A **decorator** is a function that adds metadata to a class, property, method or parameter. **Component** decorator allows you to mark a class as an Angular component and provide additional metadata that determines how the component should be processed, instantiated and used at runtime.

* Class decorators, e.g. @Component and @NgModule
* Property decorators for properties inside classes, e.g. @Input, @Output, @ViewChield
* Method decorators for methods inside classes, e.g. @HostListener, @HostBinder
* Parameter decorators for parameters inside class constructors, e.g. @Inject, @Optional, @Self

33. What is components in angular

Components are simply **classes** that are designated as a component with the help of a component **decorator**. Every component has a defined **template** which can communicate with the code defined in the component class.

34. What is Module in angular

Module in **Angular** is a place where you can group the components, directives, pipes, and services, which are related to the application. In case you are developing a website, the header, footer, left, center and the right section become part of a module.

35. What is NgModule in angular

An **NgModule** is a **class** marked by the @**NgModule** decorator. @**NgModule** takes a metadata object that describes how to compile a component's template and how to create an injector at runtime.

36. What is ngModle in angular

ngModule is built in **directive** belongs to FormsModule, that implement two way data binding in angular, It creates a FormControl instance from a domain model and binds it to a form control element.

37. How to fix/solve error “Can't bind to 'ngModel' since it isn't a known property of 'input'” in angular

Import the FormsModule package in your Angular module

38. What are Services in angular

Services are simple classes with **@Injectatble decorator**. In an Angular application components are dependent on services to display data on view. Fetching data for the components is handled by Angular services. Since the data fetching portion is handled separately in services, because it **makes** easier to mock test services.

39. What are RxJS in angular

RxJS (Reactive Extensions for JavaScript) is a library for **reactive programming using observables** that makes it easier to compose asynchronous or callback-based code.

RxJS provides an implementation of the Observable type, which is needed until the type becomes part of the language and until browsers support it. The library also **provides utility functions** **for creating and working with observables**.

These utility functions can be used for:

* Converting existing code for async operations into observables
* **Iterating through the values** in a stream
* **Mapping values to different types**
* **Filtering streams**
* Composing multiple streams

40. Example of Promise

void test()

{

let promise = new Promise((success,reject)=>{

httpClient.get('Url').toPromise().then(

res=>{

this.result = res.result.map(item=> {

return new Test(

a = res.result.a

)

}

success();

},

mes=>{

reject(mes);

}

);

});

return promise;

}

OR

return this.httpClient.get<Employee[]>("http://localhost:4444/api/Employee").toPromise();

41. Example of Observable

getEmployees(){

return this.httpClient.get<Employee[]>("http://localhost:4444/api/Employee").pipe(

map(R=> {

console.log(R);

return R;

}),

catchError((err,emp)=> {

console.log("Error :"+ err);

return emp

})

)

}

42. Observable vs Subject vs BehaviorSubject

--------------Observable Method 1--------------

const simpleObservable = new Observable((observer) => {

observer.next("bla bla bla")

obs.error("error data");

observer.complete()

})

simpleObservable.Subscribe();

--------------Observable Method 2--------------

let obser = Observable.create((obs)=>

{

obs.next("resposne data")

obs.error("error data");

obs.complete();

})

obser.subscribe();

//OR

obser.subscribe(res=>{ },err=>{},()=>{})

--------------

--------------Subject --------------

//Publisher itself is subscriber

let obser = new Subject(1);

obser.next(34);

obser.subscribe();

--------------Behavior Subject--------------

//Publisher itself is subscriber like Subject

let obser = new BehaviorSubject(1);

obser.next(34);

obser.value;

obser.subscribe();

43. Event Emitter Example

@Component({

selector: 'chield'

template: `<button (click)="sendNotification()">Notifi to parent</button>

})

export class chieldComponent

{

@Output notifyParent : EventEmiter<any> = new EventEmiter();

sendNotification()

{

this.notifyParent.emit("I am chield, Hi parent how are you.");

}

}

@Component({

selector: 'parent'

template: `<chield (notifyParent)="getNotification($event)">{{data}}</chield>

})

export class chieldComponent

{

data: string

getNotification(emitValue)

{

data = emitValue;

}

}