**Angular Interview questions part 1**

**Q: What is Angular?**

**A:** Angular is an open-source, front-end web development framework based on TypeScript. It is most suited for developing enterprise web applications because the code is reusable and maintainable.

**Q: What is Typescript?**

A: TypeScript is a primary language for Angular application development. It is a superset of JavaScript with design-time support for type safety

**Q: What are the angular essential building blocks of angular?**

A: **Components**:  encapsulates the logic of the view, data, and the HTML mark-up. Every app must have at least one part.

**Modules**: It is a container that groups related components

**Templates**: This is used to define the views of an Angular applications

**Directives**: It allow developers to add new HTML syntax, that is application-specific. The behaviour is essentially added to the existing DOM elements.

**Service** − This is used to create components which can be shared across the entire application.

### Q: Difference Between Angular and Angularjs?

### A:

|  |  |
| --- | --- |
| AngularJS | Angular |
| Based on JavaScript | Based on TypeScript |
| Based on the MVC design pattern | Based on components, modules, and directives |
| No support for mobile app | Supports mobile |
| Can’t build SEO friendly applications | SEO friendly applications can be easily created |
| Dependency Injection tokens can only be strings. Also, only the injector is present. | DI Tokens can be of any type, for example, strings or class. Angular follows a tree-hierarchy for injectors starting with the root injector, and a nozzle for each component henceforth. |
| No support or backward compatibility | Backward compatibility can be done without issues. Also, there is a lot of support documentation for the same. |
| Requires specific ng directive for each of property, event, and image | For event binding, () is used and for image or property binding [] is used |

### Q: Advantages of Angular

### A:

* Supports two-way data binding.
* Supports validations and template syntax (both angular and static).
* We can add custom animations, directives, and services.
* The event handling process is seamless.
* Hierarchical Dependency Injection structure (Parent-child).
* Provision to facilitate RESTful services and client-server communication.

### Q: Which Is the Latest Version of Angular? What Are the New Features in It?

### A: Angular 12 (released on 12th May 2021),

### Features

### Ivy Everywhere: View Engine has been deprecated

### Migrating from legacy i18n message IDs

### Protractor

### Nullish Coalscing : The nullish coalescing operator (??) has been assisting engineers with composing cleaner code in TypeScript classes for some time now

### Improvement is styling : In Angular v12, Components will have support for inline Sass in the styles field of the @Component decorator. Already, Sass was just accessible in outside resources because of the Angular compiler. You can empower this component in your current applications simply by adding "inlineStyleLanguage": "scss" to angular.json. Else, it will be accessible to new tasks utilizing SCSS.

### Deprecating support of E11:

### Strict by default : The strict mode of Angular has now been enabled by default in the CLI as one of the features of  Angular 12.

### Production by default : Up to this point, running the ng build command made an improvement towards a development build. In Angular version 12, ng build will default to a production build.

### Metadata for requests and interceptors: To start with, the HttpClient would now be able to be utilized to store and recover custom metadata for requests. This is especially helpful for HTTP interceptors. This capacity can be utilized through the new HttpContext.

### appendAll on HttpParams

### HttpStatusCode

### Ng API improvements

### Abstract Construct Signatures

### WebPack 5.37 support

### Q: Explain the Architecture of Angular?

### A:

### Q:What Is Angular CLI? How Do You Use It?

### A: Angular CLI automates the end-to-end development process. With a CLI (Command Line Interface), we can create a new project, add new features, and run tests (unit tests and end-to-end tests) by just typing a few simple commands. This way, development and testing processes both become faster.

ng new - generate new angular projects

ng build - compiles an Angular app

ng generate - generates components

ng serve - builds and serves your entire project.

ng test - runs unit tests in the existing project

**Q: What are Directives in Angular?**

**A:** It allow developers to add new HTML syntax, that is application-specific. The behaviour is essentially added to the existing DOM elements.

**Q: Types of directives in Angular?**

### A: Attribute directives: Used to add new attributes for the existing HTML elements to change its look and behaviour. Some of the examples are NgStyle, NgClass and NgModel. Whereas, NgModel is two-way attribute data binding

**Example** : <p [showToolTip]='Tips' />

### Structural directives: Used to add or remove DOM elements in the current HTML document.

### There are three pre-defined directives NgIf, NgFor and NgSwitch.

**Example :** <div \*ngIf="isNeeded">

Only render if the \*isNeeded\* value has true value.

</div>

### Component based directives : Component can be used as directives. Every component has Input and Output option to pass between component and its parent HTML elements.

### Component provides @Input and @Output decorator to send and receive information between parent and child components.

**Example** : <list-item [items]="fruits"> ... </list-item>

### Custom directives: Angular provides option to extend the angular directive with user defined directives and it is called Custom directives.

### Q: How Is Dependency Injection (DI) Done in Angular?

### A: DI, is a design pattern in which a class requests dependencies from external sources rather than creating them.

### DI framework provides dependencies to a class upon instantiation. You can use Angular DI to increase flexibility and modularity in your applications.

 Injectors can be configured in three different ways,

* @Injectable() : decorator for the service.
* @NgModule : for NgModule.
* @Component : for the component.

### Q: What Are Templates in Angular?

A:  The template is simply an HTML view where binding controls can display data to the properties of the Angular component. Templates can be defined inline using the template property as –

Template:

<div>My sample Template</div>

Or can be called from a different HTML file by using @Component decorator’s URL property –

templateUrl: 'app/app.component.html'

### Question: What Does the Representation [()] Mean?

**Answer:** This is a representation for ngModel used for two-way data binding. It is written as [(ngModel)] = “propertyvalue”.

### Question: What Does {{}} Represent? What Is It Used For? Show an Example

**Answer:** The double curly braces represent interpolation or expression. Angular converts it into property binding. The name of the component is written inside the inner curly braces. During execution, the name is replaced by the actual string value of the property. For example,

<h2> {{apptitle}} <img src="{{imgname}}" style="height:30px"> </h2>

Angular will evaluate and replace apptitle and imgname with their actual values.

### Q:  How Are Angular Expressions Different From Javascript Expressions?

A:

|  |  |
| --- | --- |
| **Angular expressions** | **JavaScript expressions** |
| Conditions, exceptions, and loops (control statements) cannot be used | All the control statements can be used |
| Regular expressions cannot be used | Regex is widely used |
| Filters can be used within the expression itself so that data is formatted before being displayed | Cannot be used |
| Functions cannot be declared | Any number of functions can be declared |

### Q: Difference between component and directives

### A:

|  |  |
| --- | --- |
| ****Component**** | ****Directive**** |
| To register component, Annotation used is @Component | @Directive is used to register a directive |
| The primary purpose of ingredients is to break the complex application into smaller, more manageable parts (components) | Purpose of the `directive is to create new custom components that are reusable |
| Each DOM element can have only one component | Any number of directives can be used in one DOM element |
| Component mandatorily requires @View decorator, template, or template URL to specify the view. | A directive has nothing to do with views |

### Q: What Is Data Binding, and What Are the Different Categories of Data Binding?

### A: Data binding is used to connect the application data and DOM i.e. components with the template. They can be categorized based on the direction of the data flow.

|  |  |  |
| --- | --- | --- |
| **Data flow Direction** | **Type** | **Description** |
| From source to view (one-way) | Interpolation - Attribute, style, class, property | Interpolates values calculated from application data into HTML |
| From lightview to the source (one-way) | Event | Enables applications to respond to users in the target environment. |
| View-source-view (two-way) | Two-way | Changes in the application state automatically get reflected in the view and vice-versa. For this type of binding, ngModel directive is used. |

### Q: Explain the Differences Between One-Way Binding and Two-Way Binding?

### A: Two way binding : UI or view is continuously updated automatically as the data model changes. The process is similar to the synchronization process.

### One Way Binding: The view doesn’t change or update automatically whenever there is a change in the data model. Custom code needs to be manually written to reflect changes.

### Q: What Are the Various Filters Supported by Angular?

### A:

|  |  |
| --- | --- |
| **Filter name** | **Description** |
| Uppercase | Convert string to uppercase |
| Lowercase | Convert string to lowercase |
| Date | Convert date to the specified format |
| Currency | Convert the number to currency format |
| Number | Format number into a string |
| Orderby | Orders an array by specific expression |
| limitTo | Limits array into the specified number of elements; string to specified number of characters |
| JSON | Format object to JSON string |
| Filter | A select a subset of items from the array |

### Example : <p>Amount: {{ amount | currency }}</p>

### Q: What Are Ngmodules?

### A:  Ngmodules enable developers to declare all the relationships in one place with metadata. Thus, in short, NgModules are built from metadata that describes components, services, directives, pipes etc… Angular then creates a component factory, a class that creates components.

### Q: Difference Between Javascript Modules and Ngmodules?

### A:

|  |  |
| --- | --- |
| **JS modules** | **NgModules** |
| Bounds all the classes | Bounds only declarable classes |
| All the member classes are defined in a single file | The module’s classes are listed in the @NgModule.declarations list |
| Cant extend the entire application with services | The entire application can be extended with services using @NgModules.providers list to add providers |
| Can import or export any kind of classes | It can import or export only those declarable classes that it owns or imports from other modules. |

### Q: What Are ngIf and ngFor?

**A:** ngif and ngFor are used as control statements. Below are the examples

**ngIf**

<p \*ngIf="display">Show this only if the Boolean "display" is true</p>

Where the display is a boolean with the value true or false.

**ngFor**

ngFor is used to loop through and display elements of an array (set of data).

<tr \*ngFor="let student of students; let i = index"> <td>{{student.name}}

</td> <td>{{i}}</td> </tr>

The second part (i=index) is optional and only needed if you want to display the index.

### Q: Is There a Way to Convert the Typescript Code Into Javascript Code? How Is It Done?

### A: Yes, converting TypeScript into JavaScript is called as transpilation.

### Q: What Is the Digest Cycle?

### A: Digest cycle is the process of monitoring watchlist to track the changes in the value of the watch variable. The digest cycle is implicitly triggered, but we can also trigger it manually using $apply() function.

### Q: What Is a Pipe? Write a Simple Code to Demonstrate.

A: Pipe (|) is used to transform input data into desired format. For example,

<p>Price is {{ price | currency }}</p>

### Q: Can You Create a Parameterized Pipe in the Above Example?

**Answer:** Yes,

<p>Price is {{ price | currency : “USD$” : 0.00 }}</p>

### Q: Explain How You Can Chain Pipes

**A:** We can add any number of filters using pipes -

<p>Average is {{ average | uppercase | number}}</p>

**Q: Is It Possible to Create a Custom Pipe? How?**

**A:** Yes, we can create custom pipes.

* Pipe metadata @Pipe decorator can be imported from core Angular library
* Pipe is a class that is decorated with the above metadata (@Pipe({name: 'myCustomPipe'}))
* The next step is to define the transformation. For this, the pipe class should implement the method transform() of the PipeTransform class.
* Specify the pipe name in the main code

<p>Size: {{number | myCustomPipe: 'Error'}}</p>

### Q: What Is the Purpose of an Async Pipe?

**A:** Async pipe subscribes to a promise or an observable, and returns the latest value. If a new value is emitted, the pipe marks the component that needs to be checked for any changes.

<code>observable|async</code>

### Q: What Is the Difference Between Pure and Impure Pipe?

A:

|  |  |
| --- | --- |
| **Pure pipe** | **Impure pipe** |
| Doesn’t get affected by internal state | Can produce different output for the same input based on the internal state |
| Can be shared with many different instances | It cannot be shared because the internal state can be affected by any factors. |

### Q: Explain the Importance of HttpClient.

A: HttpClient is a simplified Http API for Angular applications. It gives better observable APIs, better error handling mechanisms, testability, request and response interception, typed request and response objects. The HttpClientAPI rests on the XMLHttpRequest interface exposed by the browsers.