<https://hackr.io/blog/angular-interview-questions>

## Angular Interview Questions:

So, ready to see how well your preparation is going? Without further ado, here we present you with some important Angular interview questions that you might expect in your interview:

**Question: What is Angular?**  
**Answer:**Angular is a TypeScript-based open-source web application framework, developed and maintained by Google. It offers an easy and powerful way of building front end web-based applications.

Angular integrates a range of features like declarative templates, dependency injection, end-to-end tooling, etc. that facilitates web application development.

**Question**: **Define the ng-content Directive?**  
**Answer**: Conventional HTML elements have some content between the tags. For instance:

<p>Put your paragraph here</p>

Now consider the following example of having custom text between angular tags:

<app-work>This won’t work like HTML until you use ng-content Directive</app-work>

However, doing so won’t work the way it worked for HTML elements. In order to make it work just like the HTML example mentioned above, we need to use the ng-content Directive. Moreover, it is helpful in building reusable components.

Know more about the [ng-content directive](https://blog.angular-university.io/angular-ng-content/).

**Question: Please explain the various features of Angular.**  
**Answer:** There are several features of Angular that makes it an ideal front end JavaScript framework. Most important of them are described as follows:

* **Accessibility Applications**

Angular allows creating accessible applications using ARIA-enabled components, built-in a11y test infrastructure, and developer guides.

* **Angular CLI**

Angular provides support for command line interface tools. These tools can be used for adding components, testing, instant deploying, etc.

* **Animation Support**

Angular’s intuitive API allows the creation of high-performance, complex animation timelines with very little code.

* **Cross-Platform App Development**

Angular can be used for building an efficient and powerful desktop, native, and progressive web apps. Angular provides support for building native mobile applications using Cordova, Ionic, or NativeScript.

Angular allows creating high performance, offline, and zero-step installation progressive web apps using modern web platform capabilities. The popular JS framework can also be used for building desktop apps for Linux, macOS, and Windows.

* **Code Generation**

Angular is able to convert templates into highly-optimized code for modern JavaScript virtual machines.

* **Code Splitting**

With the new Component Router, Angular apps load quickly. The Component Router offers automatic code-splitting so that only the code required to render the view that is requested by a user is loaded.

* **Synergy with Popular Code Editors and IDEs**

Angular offers code completion, instant errors, etc. with popular source code editors and IDEs.

* **Templates**

Allows creating UI views with a simple and powerful template syntax.

* **Testing**

Angular lets you carry out frequent unit tests using Karma. The Protractor allows running faster scenario tests in a stable way.

**Question**: **Demonstrate navigating between different routes in an Angular application.**  
**Answer**: Following code demonstrates how to navigate between different routes in an Angular app dubbed “Some Search App”:

import {Router} from "@angular/router";

.

.

.

@Component({

 selector: 'app-header',

 template: `

<nav class="navbar navbar-light bg-faded">

 <a class="navbar-brand" (click)="goHome()">Some Search App</a>

 <ul class="nav navbar-nav">

   <li class="nav-item">

     <a class="nav-link" (click)="goHome()">Home</a>

   </li>

   <li class="nav-item">

     <a class="nav-link" (click)="goSearch()">Search</a>

   </li>

 </ul>

</nav>

`

})

class HeaderComponent {

 constructor(private router: Router) {}

 goHome() {

   this.router.navigate(['']);

 }

 goSearch() {

   this.router.navigate(['search']);

 }

}

**Question: Could you explain services in Angular?**  
**Answer:** Singleton objects in Angular that get instantiated only once during the lifetime of an application are called services. An Angular service contains methods that maintain the data throughout the life of an application.

The primary intent of an Angular service is to organize as well as share business logic, models, or data and functions with various components of an Angular application.

The functions offered by an Angular service can be invoked from any Angular component, such as a controller or directive.

**Question: Discuss the advantages and disadvantages of using Angular?**  
**Answer:** Following are the various advantages of using Angular:

* Ability to add a custom directive
* Exceptional community support
* Facilitates client and server communication
* Features strong features, such as Animation and Event Handlers
* Follows the MVC pattern architecture
* Offers support for static template and Angular template
* Support for two-way data-binding
* Supports dependency injection, RESTful services, and validations

Disadvantages of using Angular are enumerated as follows:

* Complex SPAs can be inconvenient and laggy to use due to their size
* Dynamic applications do not always perform well
* Learning Angular requires a decent effort and time

**Question**: **Enumerate some salient features of Angular 7.**  
**Answer**: Unlike the previous versions of Angular, the 7th major release comes with splitting in @angular/core. This is done in order to reduce the size of the same. Typically, not each and every module is required by an Angular developer. Therefore, in Angular 7 each split of the @angular/core will have no more than 418 modules.

Also, Angular 7 brings drag-and-drop and virtual scrolling into play. The latter enables loading as well as unloading elements from the DOM. For virtual scrolling, the latest version of Angular comes with the <cdk-virtual-scroll-viewport> package. Furthermore, Angular 7 comes with a new and enhanced version of the ng-compiler.

**Question: What is string interpolation in Angular?**  
**Answer:** Also referred to as moustache syntax, string interpolation in Angular refers to a special type of syntax that makes use of template expressions in order to display the component data. These template expressions are enclosed within double curly braces i.e. {{ }}.

The JavaScript expressions that are to be executed by Angular are added within the curly braces and the corresponding output is embedded into the HTML code. Typically, these expressions are updated and registered like watches as a part of the digest cycle.

**Question**: **Explain Angular Authentication and Authorization.**  
**Answer**: The user login credentials are passed to an authenticate API, which is present on the server. Post server-side validation of the credentials, a JWT (JSON Web Token) is returned. The JWT has information or attributes regarding the current user. The user is then identified with the given JWT. This is called authentication.

Post logging-in successfully, different users have a different level of access. While some may access everything, access for others might be restricted to only some resources. The level of access is authorization.

Here is a detailed post on Angular 7 – JWT Authentication Example & Tutorial: http://jasonwatmore.com/post/2018/11/16/angular-7-jwt-authentication-example-tutorial

**Question: Can you explain the concept of scope hierarchy in Angular?**  
**Answer:** Angular organizes the $scope objects into a hierarchy that is typically used by views. This is known as the scope hierarchy in Angular. It has a root scope that can further contain one or several scopes called child scopes.

In a scope hierarchy, each view has its own $scope. Hence, the variables set by a view’s view controller will remain hidden to other view controllers. Following is a typical representation of a Scope Hierarchy:

* Root $scope
  + $scope for Controller 1
  + $scope for Controller 2
  + …
  + ..
  + .
  + $scope for Controller n

**Question**: **How to generate a class in Angular 7 using CLI?**  
**Answer**:

ng generate class Dummy [options]

This will generate a class named Dummy.

**Question: Explain what is the difference between Angular and backbone.js?**  
**Answer:** Following are the various notable differences between Angular and Backbone.js

* **Architecture**

Backbone.js makes use of the MVP architecture and doesn’t offer any data binding process. Angular, on the contrary, works on the MVC architecture and makes use of two-way data binding for driving application activity.

* **Community Support**

Being backed by Google greatly ups the community support received by the Angular framework. Also, extensive documentation is available. Although Backbone.js has a good level of community support, it only documents on Underscore.js templates, not much else.

* **Data Binding**

Angular uses two-way data binding process and thus is a bit complex. Backbone.js, on the contrary, doesn’t have any data binding process and thus, has a simplistic API.

* **DOM**

The prime focus of Angular JS is upon valid HTML and dynamic elements that imitate the underlying data for rebuilding the DOM as per the specified rules and then works on the updated data records.

Backbone.js follows the direct DOM manipulation approach for representing data and application architecture changes.

* **Performance**

Thanks to its two-way data binding functionality, Angular offers an impactful performance for both small and large projects.

Backbone.js has a significant upper hand in performance over Angular in small data sets or small webpages. However, it is not recommended for larger webpages or large data sets due to the absence of any data binding process.

* **Templating**

Angular supports templating via dynamic HTML attributes. These are added to the document to develop an easy to understand application at a functional level. Unlike Angular, Backbone.js uses [Underscore.js](https://en.wikipedia.org/wiki/Underscore.js) templates that aren’t fully-featured as Angular templates.

* **The Approach to Testing**

The approach to testing varies greatly between Angular and Backbone.js due to the fact that while the former is preferred for building large applications the latter is ideal for developing smaller webpages or applications.

For Angular, unit testing is preferred and the testing process is smoother through the framework. In the case of Backbone.js, the absence of a data binding process allows for a swift testing experience for a single page and small applications.

* **Type**

Angular is an open-source JS-based front-end web application framework that extends HTML with new attributes. On the other hand, Backbone.js is a lightweight JavaScript library featuring a RESTful JSON interface and MVP framework.

**Question**: **How do Observables differ from Promises?**  
**Answer**: As soon as a [promise](http://andyshora.com/promises-angularjs-explained-as-cartoon.html) is made, the execution takes place. However, this is not the case with observables because they are lazy. This means that nothing happens until a subscription is made. While promises handle a single event, observable is a stream that allows passing of more than one event. A callback is made for each event in an observable.

**Question: Please explain the difference between Angular and AngularJS?**  
**Answer:** Various differences between Angular and AngularJS are stated as follows:

* **Architecture –**AngularJS supports the MVC design model. Angular relies on components and directives instead
* **Dependency Injection (DI) –**Angular supports a hierarchical Dependency Injection with unidirectional tree-based change detection. AngularJS doesn’t support DI
* **Expression Syntax –** In AngularJS, a specific ng directive is required for the image or property and an event. Angular, on the other hand, use () and [] for blinding an event and accomplishing property binding, respectively
* **Mobile Support –** AngularJS doesn’t have mobile support while Angular does have
* **Recommended Language –** While JavaScript is the recommended language for AngularJS, TypeScript is the recommended language for Angular
* **Routing –** For routing, AngularJS uses $routeprovider.when() whereas Angular uses @RouteConfig{(…)}
* **Speed –** The development effort and time are reduced significantly thanks to support for two-way data binding in AngularJS. Nonetheless, Angular is faster thanks to upgraded features
* **Structure –** With a simplified structure, Angular makes the development and maintenance of large applications easier. Comparatively, AngularJS has a less manageable structure
* **Support –** No official support or updates are available for the AngularJS. On the contrary, Angular has active support with updates rolling out every now and then

**Question**: **Observe the following image:**



**Question: Could you explain the concept of templates in Angular?**  
**Answer:** Written with HTML, templates in Angular contains Angular-specific attributes and elements. Combined with information coming from the controller and model, templates are then further rendered to cater the user with the dynamic view.

**Question: What should replace the “?”?**  
**Answer**: Directives. The image represents the types of directives in Angular; Attribute, structural, and custom.

**Question: Explain the difference between an Annotation and a Decorator in Angular?**  
**Answer:** In Angular, annotations are used for creating an annotation array. They are only metadata set of the class using the Reflect Metadata library.

Decorators in Angular are design patterns used for separating decoration or modification of some class without changing the original source code.

**Question: What are directives in Angular?**  
**Answer:** Directives are one of the core features of Angular. They allow an Angular developer to write new, application-specific HTML syntax. In actual, directives are functions that are executed by the Angular compiler when the same finds them in the DOM. Directives are of three types:

* Attribute Directives
* Component Directives
* Structural Directives

**Question**: **What are the building blocks of Angular?**  
**Answer**: There are essentially 9 building blocks of an Angular application. These are:

1. **Components –** A component controls one or more views. Each view is some specific section of the screen. Every Angular application has at least one component, known as the [root component](https://www.learnhowtoprogram.com/javascript/angular/angular-2-setup-root-component-root-module-and-more). It is bootstrapped inside the main module, known as the root module. A component contains application logic defined inside a class. This class is responsible for interacting with the view via an API of properties and methods.
2. **Data Binding –** The mechanism by which parts of a template coordinates with parts of a component is known as data binding. In order to let Angular know how to connect both sides (template and its component), the binding markup is added to the template HTML.
3. **Dependency Injection (DI) –** Angular makes use of DI to provide required dependencies to new components. Typically, dependencies required by a component are services. A component’s constructor parameters tell Angular about the services that a component requires. So, a dependency injection offers a way to supply fully-formed dependencies required by a new instance of a class.
4. **Directives –** The templates used by Angular are dynamic in nature. Directives are responsible for instructing Angular about how to transform the DOM when rendering a template. Actually, components are directives with a template. Other [types of directives](https://angular.io/guide/attribute-directives) are attribute and structural directives.
5. **Metadata –** In order to let Angular know how to process a class, metadata is attached to the class. For doing so decorators are used.
6. **Modules –** Also known as NgModules, a module is an organized block of code with a specific set of capabilities. It has a specific application domain or a workflow. Like components, any Angular application has at least one module. This is known as the root module. Typically, an Angular application has several modules.
7. **Routing –** An Angular router is responsible for interpreting a browser URL as an instruction to navigate to a client-generated view. The router is bound to links on a page to tell Angular to navigate the application view when a user clicks on it.
8. **Services –** A very broad category, a service can be anything ranging from a value and function to a feature that is required by an Angular app. Technically, a service is a class with a well-defined purpose.
9. **Template –** Each component’s view is associated with its companion template. A template in Angular is a form of HTML tags that lets Angular know that how it is meant to render the component.

**Question: Please explain the differences between Angular and jQuery?**  
**Answer:** The single biggest difference between Angular and jQuery is that while the former is a JS frontend framework, the latter is a JS library. Despite this, there are some similarities between the two, such as both features DOM manipulation and provides support for animation.

Nonetheless, notable differences between Angular and jQuery are:

* Angular has two-way data binding, jQuery does not
* Angular provides support for RESTful API while jQuery doesn’t
* jQuery doesn’t offer deep linking routing though Angular supports it
* There is no form validation in jQuery whereas it is present in Angular

**Question: Could you explain the difference between Angular expressions and JavaScript expressions?**  
**Answer:** Although both Angular expressions and JavaScript expressions can contain literals, operators, and variables, there are some notable dissimilarities between the two. Important differences between Angular expressions and JavaScript expressions are enlisted below:

* Angular expressions support filters while JavaScript expressions do not
* It is possible to write Angular expressions inside the HTML tags. JavaScript expressions, contrarily, can’t  be written inside the HTML tags
* While JavaScript expressions support conditionals, exceptions, and loops, Angular expressions don’t

**Question: Can you give us an overview of Angular architecture?**  
**Answer**: You can draw some like this:



Here is Angular Architecture in detail: https://angular.io/guide/architecture

**Question**: **What is Angular Material?**  
**Answer**: It is a UI component library. [Angular Material](https://material.angular.io/) helps in creating attractive, consistent, and fully functional web pages as well as web applications. It does so while following modern web design principles, including browser portability and graceful degradation.

**Question**: **What is AOT (Ahead-Of-Time) Compilation?**  
**Answer**: Each Angular app gets compiled internally. The Angular compiler takes in the JS code, compiles it and then produces some JS code. This happens only once per occasion per user. It is known as AOT (Ahead-Of-Time) compilation.

**Question**: **What is Data Binding? How many ways it can be done?**  
**Answer**: In order to connect application data with the DOM (Data Object Model), data binding is used. It happens between the template (HTML) and component (TypeScript). There are 3 ways to achieve data binding:

1. Event Binding – Enables the application to respond to user input in the target environment
2. Property Binding – Enables interpolation of values computed from application data into the HTML
3. Two-way Binding – Changes made in the application state gets automatically reflected in the view and vice-versa. The ngModel directive is used for achieving this type of data binding.

**Question**: **What is demonstrated by the arrow in the following image?**

  
**Answer**: This represents a dependency injection or DI.

**Question: Can you draw a comparison between the service() and the factory() functions?**  
**Answer:** Used for the business layer of the application, the *service()* function operates as a constructor function. The function is invoked at runtime using the *new* keyword.

Although the *factory()* function works in pretty much the same way as the *service()* function does, the former is more flexible and powerful. In actual, the *factory()* function are design patterns that help in creating objects.

**Question: Please explain the digest cycle in Angular?**  
**Answer:** The process of monitoring the watchlist in order to track changes in the value of the watch variable is termed the digest cycle in Angular. The previous and present versions of the scope model values are compared in each digest cycle.

Although the digest cycle process gets triggered implicitly, it is possible to start it manually by using the *$apply()* function.

**Question: Could you explain the various types of filters in Angular.**  
**Answer:** In order to format the value of expression so that it can be displayed to the user, Angular has filters. It is possible to add these filters to the controllers, directives, services, or templates. Angular also provides support for creating custom filters.

Organizing data in such a way so that it is displayed only when certain criteria are fulfilled is made possible using filters. Filters are added to the expressions using the pipe ‘|’ character. Various types of Angular filters are enumerated as follows:

* *currency* – Formats a number to the currency format
* *date* – Formats a data to some specific format
* *filter* – Selects a subset of items from an array
* *json* – Formats an object to a JSON string
* *limitTo* – Limits an array or string into a specified number of characters or elements
* *lowercase* – Formats a string to lowercase
* *number* – Formats a number to a string
* *orderBy* – Orders an array by an expression

**Question**: **What is new in Angular 6?**  
**Answer**: Here are some of the new aspects introduced in Angular 6:

* Angular Elements – It allows converting Angular components into web components and embeds the same in some non-Angular application
* Tree Shakeable Provider – Angular 6 introduces a new way of registering a provider directly inside the @Injectable() decorator. It is achieved by using the providedIn attribute
* RxJS 6 – Angular 6 makes use of RxJS 6 internally
* i18n (internationalization) – Without having to build the application once per locale, any Angular application can have “runtime i18n”

**Question**: **What is ngOnInit ()? How to define it?**  
**Answer**: ngOnInit () is a lifecycle hook that is called after Angular has finished initializing all data-bound properties of a directive. It is defined as:

Interface OnInit {

          ngOnInit () : void

     }

**Question**: **What is SPA** **(Single Page Application) in Angular? Contrast SPA technology with traditional web technology?**  
**Answer**: In the SPA technology, only a single page, which is index.HTML, is maintained although the URL keeps on changing. Unlike the traditional web technology, SPA technology is faster and easy to develop as well.

In the conventional web technology, as soon as a client requests a webpage, the server sends the resource. However, when again the client requests for another page, the server responds again with sending the requested resource. The problem with this technology is that it requires a lot of time.

**Question**: **What is the code for creating a decorator?**  
**Answer**: We create a decorator called Dummy:

    function Dummy(target) {

       dummy.log('This decorator is Dummy', target);

    }

**Question**: **What is the process called by which TypeScript code is converted into JavaScript code?**  
**Answer**: It is called Transpiling. Even though TypeScript is used for writing code in Angular applications, it gets internally transpiled into equivalent JavaScript.

**Question**: **What is ViewEncapsulation and how many ways are there do to do it in Angular?**  
**Answer**: To put simply, ViewEncapsulation determines whether the styles defined in a particular component will affect the entire application or not. Angular supports 3 types of ViewEncapsulation:

* Emulated – Styles used in other HTML spread to the component
* Native – Styles used in other HTML doesn’t spread to the component
* None – Styles defined in a component are visible to all components of the application

**Question**: **Why prioritize TypeScript over JavaScript in Angular?**  
**Answer**: TypeScript is developed by Microsoft and it is a superset of JavaScript. The issue with JS is that it isn’t a true OOP language. As the JS code doesn’t follow the Prototype Pattern, the bigger the size of the code the messier it gets. Hence, it leads to difficulties in maintainability as well as reusability. To offset this, TypeScript follows a strict OOP approach.

**Additional Tips to Win the Interview!**  
You need to be confident while giving the interview. Also, try avoiding hogwash in case you’re asked a question that you don’t know about. A simple no is better than giving some random non-answer and adding unnecessary details. Candidates that are straightforward and honest are preferred over those pretending to be know-it-alls!

Further, don’t disburse your personal details until asked for. The interviewer is more interested in knowing you as a technical person. So, all the very best! Do let us know the Angular questions you faced in the interview that are not covered here so that we can add those here for the benefit of the Angular community.

<https://www.edureka.co/blog/interview-questions/top-angularjs-interview-questions-2016/>

## ****Beginners Level – Angular Interview Questions****

## ****1. Differentiate between Angular and AngularJS.****

|  |  |  |
| --- | --- | --- |
| **Feature** | **AngularJS** | **Angular** |
| ***Architecture*** | Supports MVC design model | Uses components and directives |
| ***Language*** | Recommended Language: JavaScript | Recommended Language: TypeScript |
| ***Expression Syntax*** | Specific ng directive is required for the image/property and an event | Uses () to bind an event and [] for property binding |
| ***Mobile Support*** | Doesn’t provide any mobile support | Provides mobile support |
| ***Routing*** | $routeprovider.when() is used for routing configs | @RouteConfig{(…)} is used for routing config |
| ***Dependency Injection*** | Doesn’t supports the concept of Dependency Injection | Supports hierarchical Dependency Injection with a unidirectional tree-based change detection |
| ***Structure*** | Less manageable | Simplified structure and makes the development and maintenance of large applications easier |
| ***Speed*** | With two-way data binding development effort and time are reduced | Faster than AngularJS with upgraded features |
| ***Support*** | No support or new updates are provided anymore | Active support and frequent new updates are made |

## ****2. What is Angular?****

Angular is an open-source front-end web framework. It is one of the most popular JavaScript frameworks that is mainly maintained by Google. It provides a platform for easy development of web-based applications and empowers the front end developers in curating cross-platform applications. It integrates powerful features like declarative templates, an end to end tooling, dependency injection and various other best practices that smoothens the development path.

## ****3. What are the advantages of using Angular?****

A few of the major advantages of using Angular framework are listed below:

* It supports two-way data-binding
* It follows MVC pattern architecture
* It supports static template and Angular template
* You can add a custom directive
* It also supports RESTfull services
* Validations are supported
* Client and server communication is facilitated
* Support for dependency injection
* Has strong features like Event Handlers, Animation, etc.

## ****4. What is Angular mainly used for?****

Angular is typically used for the development of SPA which stands for Single Page Applications. **Angular** provides a set of ready-to-use modules that simplify the development of single page applications. Not only this, with features like built-in data streaming, type safety, and a modular CLI,  Angular is regarded as a full-fledged web framework.

## ****5. What are Angular expressions?****

Angular expressions are code snippets that are usually placed in binding such as {{ expression }}similar to JavaScript. These expressions are used to bind application data to HTML

Syntax: **{{ expression }}**

## ****6. What are templates in Angular?****

Templates in Angular are written with HTML that contains Angular-specific elements and attributes. These templates are combined with information coming from the model and controller which are further rendered to provide the dynamic view to the user.

## ****7. In Angular what is string interpolation?****

String interpolation in Angular is a special syntax that uses template expressions within double curly**{{ }}** braces for displaying the component data. It is also known as **moustache syntax.**The JavaScript expressions are included within the curly braces to be executed by Angular and the relative output is then embedded into the HTML code. These expressions are usually updated and registered like watches, as a part of the digest cycle.

## ****8. What is the difference between an Annotation and a Decorator in Angular?****

Annotations in angular are “only” metadata set of the class using the Reflect Metadata library. They are used to create an “annotation” array. On the other hand, decorators are the design patterns that are used for separating decoration or modification of a class without actually altering the original source code.

## ****9. What do you understand by controllers in Angular?****

Controllers are JavaScript functions which provide data and logic to HTML UI. As the name suggests, they control how data flows from the server to HTML UI.

## ****10. What is scope in Angular?****

Scope in Angular is an object that refers to the application model. It is an execution context for expressions. Scopes are arranged in a hierarchical structure which mimics the DOM structure of the application. Scopes can watch expressions and propagate events.

## ****11. What are directives in Angular?****

A core feature of Angular, directives are attributes that allow you to write new HTML syntax, specific to your application. They are essentially functions that execute when the Angular compiler finds them in the DOM.  The Angular directives are segregated into 3 parts:

1. Component Directives
2. Structural Directives
3. Attribute Directives

## ****12. What is data binding?****

In Angular, data binding is one of the most powerful and important features that allow you to define the communication between the component and DOM(Document Object Model). It basically simplifies the process of defining interactive applications without having to worry about pushing and pulling data between your view or template and component. In Angular, there are four forms of data binding:

1. String Interpolation
2. Property Binding
3. Event Binding
4. Two-Way Data Binding

## ****13. What is the purpose of a filter in Angular?****

Filters in Angular are used for formatting the value of an expression in order to display it to the user. These filters can be added to the templates, directives, controllers or services. Not just this, you can create your own custom filters. Using them, you can easily organize data in such a way that the data is displayed only if it fulfills certain criteria. Filters are added to the expressions by using the pipe character |, followed by a filter.

## ****14. What are the differences between Angular and jQuery?****

|  |  |  |
| --- | --- | --- |
| **Features** | **jQuery** | **Angular** |
| ***DOM Manipulation*** | **Yes** | **Yes** |
| ***RESTful API*** | **No** | **Yes** |
| ***Animation Support*** | **Yes** | **Yes** |
| ***Deep Linking Routing*** | **No** | **Yes** |
| ***Form Validation*** | **No** | **Yes** |
| ***Two Way Data Binding*** | **No** | **Yes** |
| ***AJAX/JSONP*** | **Yes** | **Yes** |

## ****15. What is a provider in Angular?****

A provider is a configurable service in Angular. It is an instruction to the Dependency Injection system that provides information about the way to obtain a value for a dependency. It is an object that has a $get() method which is called to create a new instance of a service. A Provider can also contain additional methods and uses $provide in order to register new providers.

## ****Intermediate Level – Angular Interview Questions****

## ****16.**** ****Does Angular support nested controllers?****

Yes, Angular does support the concept of nested controllers. The nested controllers are needed to be defined in a hierarchical manner for using it in the View.

## ****17. How can you differentiate between Angular expressions and JavaScript expressions?****

|  |  |
| --- | --- |
| **Angular Expressions** | **JavaScript Expressions** |
| 1. They can contain literals, operators, and variables. | 1. They can contain literals, operators, and variables. |
| 2. They can be written inside the HTML tags. | 2. They can’t be written inside the HTML tags. |
| 3. They do not support conditionals, loops, and exceptions. | 3. They do support conditionals, loops, and exceptions. |
| 4.  They support filters. | 4.  They do not support filters. |

## ****18. List at down the ways in which you can communicate between applications modules using core Angular functionality.****

Below are the most general ways for communicating between application modules using core Angular functionality :

* Using events
* Using services
* By assigning models on **$rootScope**
* Directly between controllers [**$parent**, **$$childHead**, **$$nextSibling**, etc.]
* Directly between controllers [**ControllerAs**, or other forms of inheritance]

## ****19. What is the difference between a service() and a factory()?****

A service() in Angular is a function that is used for the business layer of the application. It operates as a constructor function and is invoked once at the runtime using the ‘new’ keyword. Whereas factory() is a function which works similar to the service() but is much more powerful and flexible. factory() are design patterns which help in creating Objects.

## ****20. What is the difference between $scope and scope in Angular?****

* $**scope** in Angular is used for implementing the concept of dependency injection (D.I) on the other hand **scope** is used for directive linking.
* $**scope** is the service provided by $scopeProviderwhich can be injected into controllers, directives or other services whereas **Scope** can be anything such as a function parameter name, etc.

## ****21. Explain the concept of scope hierarchy?****

The $scope objects in Angular are organized into a hierarchy and are majorly used by views. It contains a root scope which can further contain scopes known as child scopes. One root scope can contain more than one child scopes. Here each view has its own $scope thus the variables set by its view controller will remain hidden to the other controllers. The Scope hierarchy generally looks like:

* Root $scope
  + $scope for Controller 1
  + $scope for Controller 2
  + ..
  + $scope for Controller ‘n’

## ****22. What is AOT?****

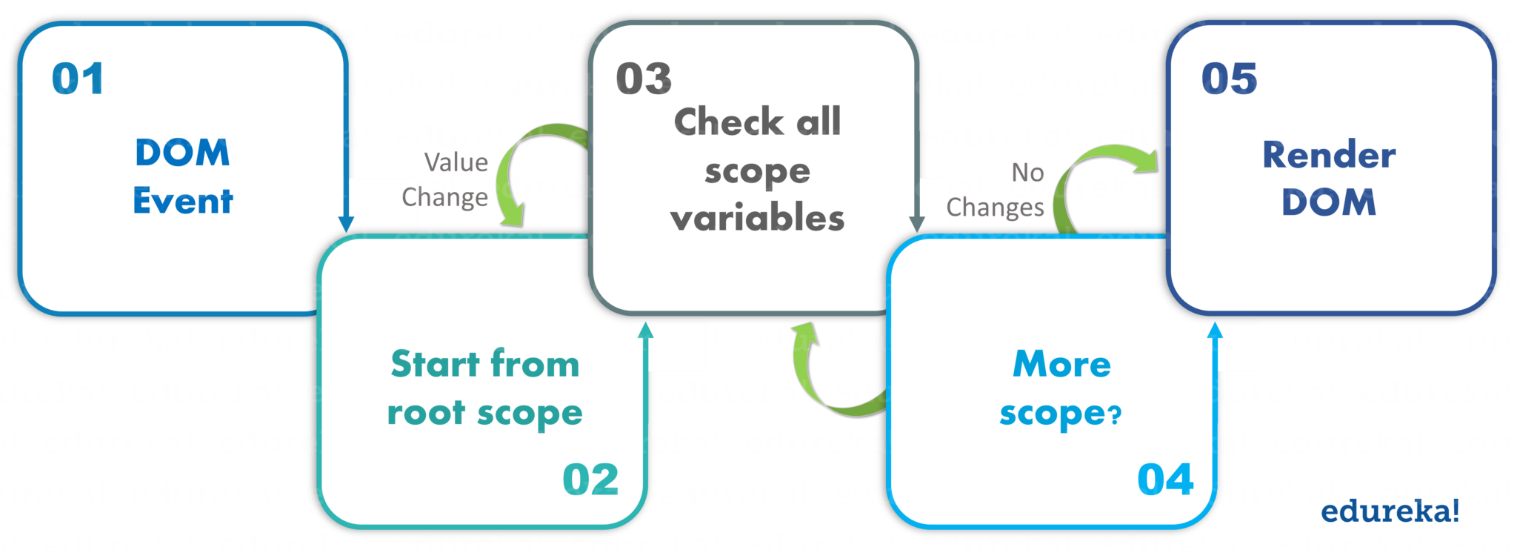
AOT stands for Angular Ahead-of-Time compiler. It is used for pre-compiling the application components and along with their templates during the build process. Angular applications which are compiled with AOT has a smaller launching time. Also, components of these applications can execute immediately, without needing any client-side compilation. Templates in these applications are embedded as code within their components. It reduces the need for downloading the Angular compiler which saves you from a cumbersome task. AOT compiler can discard the unused directives which are further thrown out using a tree-shaking tool.

## ****23. Explain jQLite.****

jQlite is also known as **jQuery lite** is a subset of jQuery and contains all its features. It is packaged within Angular, by default. It helps Angular to manipulate the DOM in a way that is compatible cross-browser. **jQLite** basically implements only the most commonly needed functionality which results in having a small footprint.

## ****24. Explain the process of digest cycle in Angular?****

The digest cycle in Angular is a process of monitoring the watchlist for keeping a track of changes in the value of the watch variable. In each digest cycle, Angular compares the previous and the new version of the scope model values. Generally, this process is triggered implicitly but you can activate it manually as well by using **$apply()**.



## ****25. What are the Angular Modules?****

All the Angular apps are modular and follow a modularity system known as NgModules. These are the containers which hold a cohesive block of code dedicated specifically to an application domain, a workflow, or some closely related set of capabilities. These modules generally contain components, service providers, and other code files whose scope is defined by the containing NgModule.  With modules makes the code becomes more maintainable, testable, and readable. Also, all the dependencies of your applications are generally defined in modules only.

## ****26. On which types of the component can we create a custom directive?****

Angular provides support to create custom directives for the following:

* **Element directives** − Directive activates when a matching element is encountered.
* **Attribute** − Directive activates when a matching attribute is encountered.
* **CSS** − Directive activates when a matching CSS style is encountered.
* **Comment** − Directive activates when a matching comment is encountered

## ****27. What are the different types of filters in Angular?****

Below are the various filters supported by Angular:

* **currency:** Format a number to a currency format.
* **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.
* **limit:**To Limits an array/string, into a specified number of elements/characters.
* **lowercase:** Format a string to lower case.
* **number:** Format a number to a string.
* **orderBy:** Orders an array by an expression.
* **uppercase:** Format a string to upper case.

## ****28. What is Dependency Injection in Angular?****

Dependency Injection (DI) is a software design pattern where the objects are passed as dependencies rather than hard-coding them within the component. The concept of Dependency Injection comes in handy when you are trying to separate the logic of object creation to that of its consumption. The ‘config’ operation makes use of DI that must be configured beforehand while the module gets loaded to retrieve the elements of the application. With this feature, a user can change dependencies as per his requirements.

## ****29. Differentiate between one-way binding and two-way data binding.****

In **One-Way** data binding, the View or the UI part does not update automatically whenever the data model changes. You need to manually write custom code in order to update it every time the view changes.

## 1 way data binding - Angular Interview Questions - Edureka

Whereas, in **Two-way** data binding, the View or the UI part is updated implicitly as soon as the data model changes. It is a synchronization process, unlike One-way data binding.

## 2 way data binding - Angular Interview Questions - Edureka

## ****30. What are the lifecycle hooks for components and directives?****

An Angular component has a discrete life-cycle which contains different phases as it transits through birth till death. In order to gain better control of these phases, we can hook into them using the following:

* **constructor:** It is invoked when a component or directive is created by calling new on the class.
* **ngOnChanges:** It is invoked whenever there is a change or update in any of the input properties of the component.
* **ngOnInit:** It is invoked every time a given component is initialized. This hook is only once called in its lifetime after the first ngOnChanges.
* **ngDoCheck:** It is invoked whenever the change detector of the given component is called. This allows you to implement your own change detection algorithm for the provided component.
* **ngOnDestroy:** It is invoked right before the component is destroyed by Angular. You can use this hook in order to unsubscribe observables and detach event handlers for avoiding any kind of memory leaks.

## ****31. What do you understand by dirty checking in Angular?****

In Angular, the digest process is known as **dirty checking**. It is called so as it scans the entire scope for changes. In other words, it compares all the new scope model values with the previous scope values. Since all the watched variables are contained in a single loop, any change/update in any of the variable leads to reassigning of rest of the watched variables present inside the DOM. A watched variable is in a single loop(digest cycle), any value change of any variable forces to reassign values of other watched variables in DOM

## ****32. Differentiate between DOM and BOM.****

|  |  |
| --- | --- |
| **DOM** | **BOM** |
| 1. Stands for Document Object Model | 1. Stands for Browser Object Model |
| 2. Represents the contents of a web page | 2. Works a level above web page and includes browser attributes |
| 3. All the Objects are arranged in a tree structure and the document can be manipulated & accessed via provided APIs only | 3. All global JavaScript objects, variables & functions become members of the window object implicitly |
| 4. Manipulates HTML documents | 4. Access and manipulate the browser window |
| 5. W3C Recommended standard specifications | 5. Each browser has its own implementation |

**33. What is Transpiling in Angular?**  
Transpiling in Angular refers to the process of conversion of the source code from one programming language to another. In Angular, generally, this conversion is done from TypeScript to JavaScript. It is an implicit process and happens internally.

## ****34. How to perform animation in Angular?****

In order to perform animation in an Angular application, you need to include a special Angular library known as Animate Library and then refer to the ngAnimate module into your application or add the ngAnimate as a dependency inside your application module.

## ****35. What is transclusion in Angular?****

The transclusion in Angular allows you to shift the original children of a directive into a specific location within a new template. The ng directive indicates the insertion point for a transcluded DOM of the nearest parent directive that is using transclusion. Attribute directives like **ng-transclude** or **ng-transclude-slot** are mainly used for transclusion.

## ****36. What are events in Angular?****

Events in Angular are specific directives that help in customizing the behavior of various DOM events. Few of the events supported by Angular are listed below:

* ng-click
* ng-copy
* ng-cut
* ng-dblclick
* ng-keydown
* ng-keypress
* ng-keyup
* ng-mousedown
* ng-mouseenter
* ng-mouseleave
* ng-mousemove
* ng-mouseover
* ng-mouseup
* ng-blur

## ****37. List some tools for testing angular applications?****

1. Karma
2. Angular Mocks
3. Mocha
4. Browserify
5. Sion

## ****38. How to create a service in Angular?****

In Angular, a service is a substitutable object that is wired together using dependency injection. A service is created by registering it in the module it is going to be executed within. There are basically three ways in which you can create an angular service. They are basically three ways in which a service can be created in Angular:

* Factory
* Service
* Provider

## ****39. What is a singleton pattern and where we can find it in Angular?****

Singleton pattern in Angular is a great pattern which restricts a class from being used more than once. Singleton pattern in Angular is majorly implemented on dependency injection and in the services. Thus, if you use ‘new Object()’ without making it a singleton, then two different memory locations will be allocated for the same object. Whereas, if the object is declared as a singleton, in case it already exists in the memory then simply it will be reused.

**40. What do you understand by REST in Angular?**

REST stands for **RE**presentational **S**tate **T**ransfer. REST is an API (Application Programming Interface) style that works on the HTTP request. In this, the requested URL pinpoints the data that needs to be processed. Further ahead, an HTTP method then identifies the specific operation that needs to be performed on that requested data. Thus, the APIs which follows this approach are known as RESTful APIs.

**41. What is bootstrapping in Angular?**

Bootstrapping in Angular is nothing but initializing, or starting the Angular app. Angular supports automatic and manual bootstrapping.

* ***Automatic Bootstrapping:*** this is done by adding the ng-app directive to the root of the application, typically on the tag or tag if you want angular to bootstrap your application automatically. When Angular finds ng-app directive, it loads the module associated with it and then compiles the DOM.
* **Manual Bootstrapping:**Manual bootstrapping provides you more control on how and when to initialize your Angular application. It is useful where you want to perform any other operation before Angular wakes up and compile the page.

## ****42. What is the difference between a link and compile in Angular?****

* Compile function is used for template DOM Manipulation and to collect all the directives.
* Link function is used for registering DOM listeners as well as instance DOM manipulation and is executed once the template has been cloned.

**43.** **What do you understand by constants in Angular?**

In Angular, constants are similar to the services which are used to define the global data. Constants are declared using the keyword “constant”. They are created using constant dependency and can be injected anywhere in controller or services.

## ****44. What is the difference between a provider, a service and a factory in Angular?****

|  |  |  |
| --- | --- | --- |
| **Provider** | **Service** | **Factory** |
| A provider is a method using which you can pass a portion of your application into app.config | A service is a method that is used to create a service instantiated with the ‘new’ keyword. | It is a method that is used for creating and configuring services. Here you create an object, add properties to it and then return the same object and pass the factory method into your controller. |

## ****45.**** ****What are Angular Global APIs?****

Angular Global API is a combination of global JavaScript functions for performing various common tasks like:

* Comparing objects
* Iterating objects
* Converting data

There are some common Angular Global API functions like:

* **angular. lowercase:** Converts a string to lowercase string.
* **angular. uppercase:** Converts a string to uppercase string.
* **angular. isString:**Returns true if the current reference is a string.
* **angular. isNumber:** Returns true if the current reference is a number.

## ****Advanced Level – Angular Interview Questions****

## ****46. In Angular, describe how will you set, get and clear cookies?****

For using cookies in Angular, you need to include a  module called ngCookies angular-cookies.js.

**To set Cookies** – For setting the cookies in a key-value format ‘put’ method is used.

cookie.set('nameOfCookie',"cookieValue");

**To get Cookies –** For retrieving the cookies ‘get’ method is used.

cookie.get(‘nameOfCookie’);

**To clear Cookies –** For removing cookies ‘remove’ method is used.

cookie.delete(‘nameOfCookie’);

## ****47.**** If your data model is updated outside the ‘Zone’, explain the process how will you the view?

You can update your view using any of the following:

1. **ApplicationRef.prototype.tick()**: It will perform change detection on the complete component tree.
2. **NgZone.prototype.run():** It will perform the change detection on the entire component tree. Here, the run() under the hood will call the tick itself and then parameter will take the function before tick and executes it.
3. **ChangeDetectorRef.prototype.detectChanges():**It will launch the change detection on the current component and its children.

## ****48. Explain ng-app directive in Angular.****

ng-app directive is used to define Angular applications which let us use the auto-bootstrap in an Angular application. It represents the root element of an Angular application and is generally declared near <html> or <body> tag. Any number of ng-app directives can be defined within an HTML document but just a single Angular application can be officially bootstrapped implicitly. Rest of the applications must be manually bootstrapped.

**Example**

<div ng-app=“myApp” ng-controller=“myCtrl”>  
First Name :  
<input type=“text” ng-model=“firstName”>  
<br />  
Last Name :  
<input type=“text” ng-model=“lastName”>  
<br>  
Full Name: {{firstName + ” ” + lastName }}  
</div>

## ****49. What is the process of inserting an embedded view from a prepared TemplateRef?****

@Component({

selector: 'app-root',

template: `

<ng-template #template let-name='fromContext'><div>{{name}}</ng-template>

`

})

export class AppComponent implements AfterViewChecked {

@ViewChild('template', { read: TemplateRef }) \_template: TemplateRef<any>;

constructor() { }

ngAfterViewChecked() {

this.vc.createEmbeddedView(this.\_template, {fromContext: 'John'});

}

}

## ****50. How can you hide an HTML element just by a button click in angular?****

An HTML element can be easily hidden using the ng-hide directive in conjunction along with a controller to hide an HTML element on button click.

**View**

<div ng-controller="MyController">

<button ng-click="hide()">Hide element</button>

<p ng-hide="isHide">Hello World!</p>

</div>

**Controller**

controller: function() {

this.isHide = false;

this.hide = function(){

this.isHide = true; }; }

So this brings us to the end of the Angular interview questions article. The topics that you learned in this Angular Interview Questions article are the most sought-after skill sets that recruiters look for in an Angular Professional. These set of Angular Interview Questions will definitely help you ace your job interview. **Good luck with your interview!**