**What are the Component in Angular?**

Component are the most basic building block of a UI in Angular Applications and its controls views. They also communicate with other components and services to bring functionality to your applications.

Angular component must have a root component that contains all other components.

Component are created using @Component decorator that is part of @angular/Core module.

Component configuration options (Metadata Properties):

1. Selector – to create an instance of this component.
2. templateUrl –
3. styleUrls –

**What is an Entry Component in Angular?**

The Entry Component is used to define components and created dynamically using the **ComponentFactoryResolver** (Provide Reusable Generation Service).

Angular creates a components factory for each of the bootstrap components with the help of ComponentFactoryResolver. And then, at run-time, it will use the factories to initiate the components.

There are two kind of entry components:

1. The Bootstrapping root component:
2. A component you specify in a root:

**What is Modules (@ngModule decorator)?**

The NgModule is a class and work with the @NgModule decorator function and also takes a metadata object that tells Angular how to compile and run module code.

The Angular module helps you to organize an application into associative block of functionality.

A module can import other modules and can expose its functionality to other modules.

**What are the @NgModule metadata properties?**

The @NgModule takes a metadata object that tells angular how to compile and launch the application.

The @NgModule metadata plays an important role in guiding the compilation process and also tells the compiler what component to compile for this module and how to link this module with other modules.

1. **Providers** – a list of DI providers.
2. **Declarations** – a list of declarable classes, components, directives and pipes that belongs to this module.
3. **Imports** – a list modules and its used to import the supporting modules (Forms, Routes).
4. **Exports** – a most of declarable components, directives, pipes and modules.
5. **entryComponents** – angular app has at least one entry component, the root component, app component.
6. **Bootstraps** – a list of components that should be compiled when this module is defined.
7. **Schemas** – that allow any non-angular elements and properties.
8. **Id** –

**What are the types of NgModules?**

1. Feature Module – the purpose of organizing an application code.
2. Routing Module – manage routes and also enables navigation from one view to another view.
3. Service Module – contains services and providers.
4. Widget Module – the third party UI component libraries.
5. Shared Module – organize your application code.

**What’s Angular Elements?**

Angular Elements resolve the problems of code reuse across multiple frameworks and provides a great way to use Angular components in non-angular environments like jQuery app or Vue.js app.

**What are Angular Directives?**

The Directives allow you attach behavior to DOM elements and the @directive decorator provide you an additional metadata that determines how directives should be processed, initiated and used at runtime.

**There are three type of Directives:**

1. Component :
2. Structural: change the DOM layout by adding, removing, and manipulating elements.
3. Attribute: change the behavior of a specified element or component.

**What are Decorators?**

The Decorators are functions that modify JavaScript classes and it also used for attaching metadata to classes.

**Difference between Component and Directive?**

The Component are used, when you want to create new elements in the DOM with their own HTML template.

The Attribute Directives are used, when you want to change or update the existing elements in the DOM.

**What is Dependency Injection (DI)?**

Managing code dependencies and way to create objects that depend upon other objects.

**What is Injector?**

A Service is just a class in Angular until you register with an Angular Dependency Injector.

The Injector is responsible for creating Angular Service Instances and injecting them into classes.

**What are @Injectable Providers?**

To inject the service into a Component, Angular provides an Injector decorator: @Injectable ().

A Provider defines the set of Injectable objects that are available in the Injector of this module.

**Why @Inject?**

The @Inject is a special technique for letting Angular knows that a parameter must be injected.

**What is Angular Service?**

Services are commonly used for storing data and making HTTP calls.

Service is provide an easy way to share the data between the components with help of DI.

Services use to fetch the data from the RESTful API.

**What is Pipe?**

@Pipes transform displayed values within a template. @Pipe class implements the “PipeTransform” interfaces **transform** method that accepts an input value and returns the transformed result.

**There are two categories of @Pipe:**

1. **Pure**: every pipe has been pure by default. When it detects pure change to the input value.
2. **Impure**: angular execute impure pipe during every component change detection cycle.

**What is Parametrizing Pipe?**

A pipe can accept any number of optional parameters to achieve output, follow the pipe name with colon (:).

**What is Changing Pipe?**

The changing pipe is used to perform the multiple operations within the single expression.

**What is AsyncPipe?**

Subscribe to an observable or promise and returns the latest value it has emitted.

**What is Angular Router?**

An Angular Router is a tool, library that configures navigations between states and views within your Angular app.

**What is Router Module?**

The Router module is a module that provides the necessary service providers and directives for navigating one view to other in application.

**What is Routes?**

Angular Routes is an array of route configurations. The **“RouteModule.forRoot”** method in the module import to configure the router.

**There are list of Routes Properties:**

1. **Path:**
2. **PathMatch:**
3. **Matcher:**
4. **Component:**
5. **redirectTo:**
6. **outlet:**
7. **canActivate:**
8. **canActivateChild:**
9. **canDeactivate:**
10. **canLoad:**
11. **data:**
12. **resolve:**
13. **runGaurdsAndResolvers:**
14. **children:**
15. **loadChildren:**

**What is PathLocationStratergy?**

A PathLocationStratergy used to configure the Location Service that represents its state in the path of the browser’s URL and PathLocationStratergy is a default routing strategy.

**What is HashLocationStrategy?**

Add the route path to the Hash (#) in the browser’s URL.

**What is Router Outlet?**

The Router-Link, RouterLink-Active and the Router outlet is directive provided by the Angular RouterModule package.

**Is it Possible have a multiple router-outlet in the same template?**

Yes! Simply adds the router-outlet name.

**What is RouterLinkActive?**

The RouterLinkActive is a directive. To add the active CSS class become the active.

**What is RouterState?**

RouterState is Interface and it represents the state of the router.

**What is ActivatedRoute?**

ActivatedRoute is an Interface and it contains the information about a route associated with a component loaded into an outlet and it can be used to traverse the router state tree.

**What is Router Events?**

1. **NavigationStart**
2. **RouteConfigLoadStart**
3. **RouteConfigLoadEnd**
4. **RoutesRecognized**
5. **GaurdsCheckStart**
6. **ChildActivationStart**
7. **ActivationStart**
8. **GaurdsCheckEnd**
9. **ResolveStart**
10. **ResolveEnd**
11. **ActivationEnd**
12. **ChildActivationEnd**
13. **NavigationEnd**
14. **NavigationCamcel**
15. **NavigationError**
16. **Scroll**

**What is HttpClient in Angular?**

HttpClient is performing HTTP requests and responses.

HttpClient also gives us advanced functionality like ability listen for progress events and interceptors to modify requests or responses.

HttpClient is use the XMLHttpRequest browser API to execute HTTP requests.

1. Get
2. Put
3. Post
4. Delete
5. Head
6. JSONP
7. Options
8. Patch

What is Angular HttpInterceptor?

Intercepts is an advanced feature that allow us to intercept each request/response and modify it before sending/receiving.

What are the HttpHeaders?

The Http Headers is immutable Map and each and every set () returns a new instance and applies the changes with lazy parsing.

What are the Validation Functions?

1. **Async Validator:** Async validator functions that take a control instance and return an observable that later emits a set of validation errors or null.
2. **Sync Validator:** Sync validator functions that take a control instance and return a set of validation error or null.