**Interview Questions Importance**

1. **Single page application?**

* Spa's are web applications that run entirely in a web browser and do not need an any page reloads.
* In single page application that will be loaded at initially once, when the user accesses the web application that is why the name is single page application. ex: Facebook and Gmail
* In Facebook, when we click on one page to another it will not refreshing.

**Advantages**:

* It provides a great user experience
* It is very fast
* It's very easy to develop, because we separate out the backend and frontend
* spa's can work offline, once all the data is fetched.

**Disadvantages**:

* Spa's are less secure, because we need to interact backend side, we need to enable cross-site scripting
* Chances for memory leak, we can handle it JavaScript memory leaks properly.

1. **Traditional page application?**

* when a user when using our application, user click on a button or a link, request will send to a server, and server-side code is responsible for generating brand new UI or a html page which will be send back to the browser, result in a page refresh.

1. **What Angular js is and why?**

* Angular js is a spa's development framework or platform, that is using HTML CSS and TypeScript.
* Architectural feature:
* Using Angular js we can quickly and easily build a single page application.
* The first architectural feature is component.

1. **Component:**

* Every angular application is made up of one component, a component is a combination of data and logic associated with HTML templates that renders with the using logics on to the web browser
* Every application will have root component followed by the several other components renders the different parts of our application on to the browser.
* For every application will have the root component followed by the several other components we must render different parts of our application on to the browser.

1. **HTML template:**

The html template than we can use two other cool features as to offer namely:

1. Angular Directives
2. Data Binding

1)Angular Directives - Using angular directives, we can define logic in our HTML page

2)Data Binding - Using data binding we can bind the data to the HTML DOM element.

There are two types of data binding are:

* Event Binding and Property Binding
* Event Binding - this is button click and or when the user places a mouse to text or input field etc.
* property binding - which will bind the HTML input properties to the model in our application component

1. **Service:**

service is a class are used to fetch the data from the backend, Angular knows how to use services whatever  
services we write we can use dependency injection to inject that service into a component and then the component in use the service to fetch the data and render it onto the UI So, Dependency injection and services are other cool features of Angular

1. **Routing:**

As the user clicks on the button or link are the browser back button, we need to be routing to the appropriate view. Angular provides routing module which can use to easily map URL paths to views instead of pages

**Versions of Angular:**

* The major change of angular version is Angular 2, its completely different on Angular 1.0 Angular 4 5 and 6 is the minor updates

1. **How to install and how to use the latest version of Angular CLI:**

**->** install a node-js

**->** install the angular CLI using the command - npm uninstall -g @angular/cli and use npm install @angular/cli

**-**> ng version - This command is showing an angular version.

**-**> ng new my-app - This is the command we used to create a project. asking stricter type checking - yes, it will be very beneficial, our application code strictly checked for type of data we deal with.

**-**> Would you like angular routing - ignore add it later, when we required.

**-**> Go to the inside of the project folder ex: cd my-app

**-**> ng serve -o - This is command we run our application into the browser.

**-**> yarn installation on googles, just type yarn installation or npm install --global yarn uses this command

**Practical:**

Component - A component is a key feature of an angular using which we will create our own HTML like elements are:<navbar> <mymenu> <footer> These are all custom HTML tag or custom elements which are their own look and behavior, and we create this using existing html elements are:<div> <h1> <a> <from> etc.

1. **Ng Modules or Router**

The basic building block of an angular application are: **ngModule or AngularModules** So ngModule defines a compilation context for a set of components or a service that address a particular functionality of our application.

**Advantages**:

* Easy maintainable or code and then reusability
* Angular supports two way of binding?1) **Input way of binding** - [hidden]=true or false, or else add a property for component, the pass property names hide or anything ex: [hidden]='hide'  
  2) **output way of binding** - It is used to when we hide the Input binding, it supports show that binding to the users ex: <input type="button" value="Toggle" (click)="hide=!hide"/>

1. **Directives:**

Directives means in html, we added to attribute in our HTML elements, which will dynamically affect the HTML DOM.

1. **ngIf** - This directive is used to display a html conditionally  
    ex: <b \*ngIf='true'>My text</b> if it’s true directly added to the HTML DOM, if false not added
2. **ngSwitch** - When we use ngSwitch the elements are matches with the ngSwitch case then will be displayed.  
    ex: <div [ngSwitch] = 'mycase'>  
    <div \*ngSwitchCase = '''one'''>  
   <div \*ngSwitchCase = '''two'''>  
    <div \*ngDefaultCase = '''one'''>
3. **ngFor** - As the name itself says it is used to repeat given DOM element based on each element in an array ex: <div \*ngFor='let c of courses'>  
    <b>c</b> </div>
4. **ngNonBindable** - It is used to tell angular not to Bind or compiler part of our page  
   ex: <div>{{myData}} </div>
5. **ngstyle and ngClass** - This will used to styling purpose   
   ngStyle - [style. <cssproperty>] ='values'  
   ngClass - it is directly assigning CSS to the html elements  
    ex: [ngClass] = "{tdata: true}"
6. **Dependency injection?**

When we build huge software application or single page applications, we don't put all the code in one single component, that code will be spread across the component to do dependency injection on angular the two steps are:

1. **Step-1** - First Register any component or any dependency inside the ngModule, that is app.module.ts
2. **Step-2** - Is to use or inject itself we do that simply provided a constructor.  
    <https://restcountries.com/> - This website is exposes to RESTful API's we can consume.

**-> click All -> we got an all links https://restcountries.com/v3.1/all**

import {HttpClient} from '@angular/common/http' - This import is support for all RESTful calls GET, POST, PUT, DELETE etc.

1. **Pipes:**

Pipes are used to transform or format the data on the HTML template files. There are several other inbuilt pipes are available in angular are:

1. titlecase 2) uppercase 3) lowercase 4) date 5) currency 6) Json 7) number 8) percentage
2. **titlecase** - It converts the first letter in the given text to capital**.**
3. **uppercase** - It converts the entire text to uppercase.
4. **lowercase** - It converts the entire text to lowercase.
5. **date** - It is used to format the date we are dealing with**.**
6. **currency** - it used to display the different countries of currency.
7. **Json** - it converts the JavaScript objects into a Json string.
8. **number** - it formats a number **number:"{minIntererDigits}.{minDecimalDigits}-{maxDecimalDigits}"**
9. **percentages** - it converts a given number to a percentage.
10. **async** - it gives a promise a given processing and slice that will work with arrays slice [10,20,30,40,50] | slice:2:4

**How to create a custom pipe**

* we use the command is (ng g p pipes/reversePipe)

1. **Forms:**

Forms are a key part of any web application

**There are two approaches of forms are**:

1. **Model Driven or Reactive Forms** - It defined as we create a model or component first and then will create the HTML form and we will bind or map to the html forms to the model using the directives that angular provides. We create Reactive forms using the ReactiveFormsModule gives us two important objects or classes are:

-> **FormGroup** - it represents a HTML <form> type element within a elements form

-> **FormControl** - it represents the <input> type element within a elements form

2) **Template Driven Forms** - Template driven forms are internally MODEL DRIVEN FORMS there are driven in the html instead of the code in the model. Just by adding the FormsModule to the ngModule configuration we will get a ngForm object thatwe can use in our HTML template, internally creates a FormGroup objects.

**14) custom Directives** - Steps to creating a custom Directives are**:**

**->** Create a Directive **->** Use TemplateRef and ViewContainerRef **->** Implement the logic  **->** createEmbeddedView () - This is the method will create a DOM dynamically

**15) Flow of the angular application:**

1) **package.json** – Here we are doing our project configuration, project script for build, test and deploy, and dependencies management.

2) **angular.json** – Project structure and root folder, prefix configuration, CSS and JavaScript, and static file in assets folder. This is an entry point file for the ng app (By default this is the bootstrap file.)   
 “index”: “src/index.html”,  
“main”: “src/main.ts”

3) **main.ts** – It is returning the AppModule and the place where the appmodule is **app** folder where we have **app.module.ts** – This is the root module by default created our angular project.