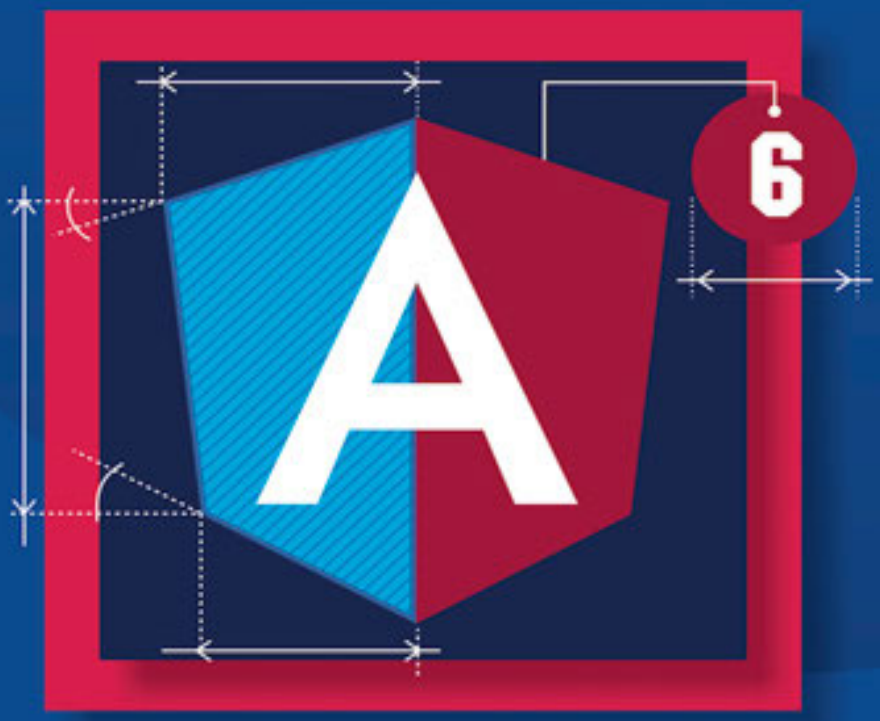

Angular

Interview Questions and Answers

Including Angular 6, 5, 4 and 2

ANIL SINGH

bpb

ANGULAR INTERVIEW

QUESTIONS AND

ANSWERS

INCLUDING ANGULAR 6,5, 4 AND 2

ANIL SINGH

FIRST EDITION 2018

ISBN:

Distributors:

BPB PUBLICATIONS

20, Ansari Road, Darya Ganj

New Delhi-110002

Ph: 23254990/23254991

BPB BOOK CENTRE

376 Old Lajpat Rai Market,

Delhi-110006

Ph: 23861747

DECCAN AGENCIES

4-3-329, Bank Street,

Hyderabad-500195

Ph: 24756967/24756400

MICRO MEDIA

Shop No. 5, Mahendra Chambers, 150

DN Rd. Next to Capital Cinema, V.T.

(C.S.T.) Station, MUMBAI-400 001 Ph:

22078296/22078297

Published by Manish Jain for BPB Publications, 20, Ansari Road, Darya Ganj, New Delhi-110002 and Printed by Repro India Ltd., Mumbai

Table of Contents

Preface

Changing job is one of the biggest challenges for any IT professional. When IT professional starts searching job, they realise that they need much more than experience. Working on a project is one thing and cracking an interview is another. This book will give you a bird's eye view of what is needed in an interview. It will help you in doing a quick revision so that you can be ready for the discussion faster.

Acknowledgements

I would like to thank my parents, wife, and daughter who patiently supported me in writing this book.

Thank you so much to my publisher (BPB), readers and reviewers for their feedbacks.

Chapter 1

The Basic Concepts of Angular

1.1 What is Angular?

Angular is a most popular web development framework for developing mobile apps and desktop applications.

Angular framework is also utilized in the cross-platform mobile development called IONIC and is not limited to web apps only.

Angular is an open source framework written and maintained by Angular team at Google and the Father of Angular is Misko Hevery.

Misko Hevery - Agile Coach at Google, Attended Santa Clara University and Lives in Saratoga, CA.

Angular is written in TypeScript and it comes with all the capabilities that typescript offers.

The core concepts of Angular:

You don't worry about the TypeScript versions. The compiler manages to the versioning related problems and Angular team working with Traceur compiler team to provide the support to build some extensions.

Angular's current target release dates and versions:

1.2 What are Angular Prerequisites?

Before you can install Angular 4 or higher versions, you must need to have some prerequisites.

You must have Node.js installed.

You must have NPM (Node Package Manager) installed.

```
node -v
```

Noted Points :

We need to setup our machine's local environments which are:

You can download the installer- https://nodejs.org/en/download/

1.3 What is Angular CLI?

The Angular CLI (Command Line Interface) is a tool to initialize, develop, scaffold and maintain Angular applications.

OR

The Angular CLI (Command Line Interface) is a set of tools that are used to initialize, develop, generate files, scaffold, and test & maintain Angular application.

You can use CLI commands to generate an app, the default AppModule is as follows –

```
ng new yourApp
```

The above CLI command is used to create a new Angular project and this CLI command will automatically creates several folders and files which are necessary for project development, testing, configuration and so on.

To use Angular CLI, we need to install it first (globally on your machine).

```
npm install -g @angular/cli
```

1.4 How to Update Angular CLI?

If you're using Angular CLI lesser version, uninstall an angular-cli package and install new versions of CLI.

```
npm uninstall -g angular-cli
npm uninstall –save-dev angular-cli
```

Global package :

```
npm uninstall -g @angular/cli
npm cache clean
npm install -g @angular/cli@latest
```

Local project package :

```
rm -rf node_modules dist # use rmdir /S/Q node_modules dist
in Windows Command Prompt; use rm -r -fo node_modules,dist
in Windows PowerShell
npm install –save-dev @angular/cli@latest
npm install
```

Some Additional CLI Commands :

1.5 What's New in Angular 2?

What are the differences between AngularJs and Angular 2?

Angular 2 is a TypeScript based open-source front-end web application platform led by the Angular Team at Google and by a community of individuals and corporations. Angular 2 is a complete rewrite from the same team that built AngularJs.

Angular 2 is a platform that makes it easy to build applications with the web. Angular 2 combines declarative templates, dependency injection, an end to end tooling, and integrated best practices to solve development challenges. Angular empowers developers to build applications that live on the web, mobile, or the desktop.

The first version of AngularJs was released in the year 2010 while Angular 2 was released in the year 2016.

Both AngularJs and Angular 2 are completely different. Actually, Angular 2 is entirely component based and AngularJs is both scope and controller based.

Angular 2 is a platform and framework for building client applications in HTML and TypeScript. Angular 2 is written in TypeScript. It implements core and optional functionality as a set of TypeScript libraries that you import into your apps.

Angular 2 is used on Cross-platform, modern browsers only. The core differences and many more advantages on Angular 2 vs. AngularJs are:

For Examples as:

AngularJs Controller :

```
var app = angular.module("userApp", []);
app.controller("productController", function($scope) {
   $scope.users = [{ name: "Anil Singh", Age:35, department
:"IT"},
        { name: "Aradhya Singh", Age:5, department :"MGMT"
},
        { name: "Reena Singh", Age:28, department :"HR" }];
});
```

Angular 2 Components using TypeScript:

Here the @Component annotation is used to add the metadata to the class. import {Component } from 'angular2/core';

```
@Component({
  selector: 'usersdata',
  template: '<h3>{{users.name}}</h3>'
})
export class UsersComponent {
```

```
  users = [{ name: "Anil Singh", Age:35, department :"IT"},
        { name: "Aradhya Singh", Age:5, department :"MGMT"
},
        { name: "Reena Singh", Age:28, department :"HR" }];
}
```

Bootstrapping in AngularJs using ng-app :

```
angular.element(document).ready(function() {
  angular.bootstrap(document, ['userApp']);
});

Bootstrapping in Angular 2 :
import { bootstrap } from 'angular2/platform/browser';
import { UsersComponent } from './product.component';
bootstrap(UserComponent);
```

The Angular 2 structural directives syntax is changed like ng-repeat is replaced with *ngFor and so many.

For example as:

```
//AngularJs
<div ng-repeat="user in users">
   Name: {{user.name}}
   Age : {{user.Age}}
   Dept: {{user.Department}}
</div>
```

And

```
//Angular 2,
<div *ngFor="let user of users">
     Name: {{user.name}}
     Age : {{user.Age}}
     Dept:  {{user.Department}}
</div>
```

1.6 What's New in Angular 4?

What are the differences between Angular 2 and Angular 4?

Off-course! Angular 4 being smaller, faster, easier to use and it will be making developer's life easier as compare to Angular 2.

Angular 2 is released in the year 2016 whereas Angular 4 is released in the year 2017.

Angular 2 and Angular 4 will use the same concept and patterns.

Angular 4 contains some additional enhancement and improvement. Consider the following enhancements:

```
<div *ngIf="user.length > 0; else empty"><h2>Users</h2></
div>
```

And

```
<ng-template #empty><h2>No users.</h2></ng-template>
```

Use of as keyword,

```
<div *ngFor="let user of users | slice:0:2 as total; index as
= i">
  {{i+1}}/{{total.length}}: {{user.name}}
</div>
```

To subscribe only once to a pipe "|" with "async" and If a user is observable, you can now use to write,

```
<div *ngIf="users | async as usersModel">
   <h2>{{ usersModel.name }}</h2> <small>{{ usersModel.age
}}</small>
</div>
```

The example as,

```
<h2>{{ 'anil singh' | titlecase }}</h2>
<!— OUPPUT - It will display 'Anil Singh' —>
```

```
//Angular 4 -
http.get('${baseUrl}/api/users', { params: { sort: 'ascending'
} });
```

And

```
//Angular 2-
const params = new URLSearchParams();
params.append('sort', 'ascending');
http.get('${baseUrl}/api/users', { search: params });
```

```
//Angular 4 -
TestBed.overrideTemplate(UsersComponent,'<h2>{{users.name}}</
h2>');
```

And

```
//Angular 2 -
TestBed.overrideComponent(UsersComponent, {
    set: { template: '<h2>{{users.name}}</h2>' }
});
```

```
@Component({
  selector: 'users-app',
  template: '<h1>Users</h1>'
})
export class UsersAppComponent {
  constructor(meta: Meta) {
      meta.addTag({ name: 'Blogger', content: 'Anil Singh'
});
  }
}
```

```
<select [compareWith]="byUId" [(ngModel)]="selectedUsers">
      <option *ngFor="let user of users"
[ngValue]="user.UId">{{user.name}}</option>
</select>
```

```
const uid = this.route.snapshot.paramMap.get('UId');
this.userService.get(uid).subscribe(user => this.name = name);
```

```
//Angular 4-
<div [ngPlural]="value">
      <ng-template ngPluralCase="0">there is nothing</ng-
template>
   <ng-template ngPluralCase="1">there is one</ng-template>
</div>
```

And

```
//Angular 2-
<div [ngPlural]="value">
     <ng-template ngPluralCase="=0">there is nothing</ng-
template>
   <ng-template ngPluralCase="=1">there is one</ng-template>
</div>
```

1.7 What's New in Angular 5?

What is the difference between Angular 4 and Angular 5?

Off-course! Angular 5 being smaller, faster, easier to use and it will be making developer's life easier as compare to Angular 2.

Angular 4 is released in the year 2017 while Angular 5 is released on 1st November 2017.

Angular 5 is going to be a much better Angular and you will be able to take advantage of it much easier.

Angular 5 contains a bunch of new features, performance improvements and a lot of bug fixes and also some surprises to Angular lovers:

The HttpClient communicate with backend services over the HTTP protocol and the Improvements is:

The Angular team recommends using HttpClientModule.

Angular 5 Added new router lifecycle events for Guards and Resolvers :

Some Bug Fixes in Angular 5 :

1.8 What's New in Angular 6?

What is the difference between Angular 5 and Angular 6?

Off-course! Angular 6 being smaller, faster, easier to use and it will be making developer's life easier.

The Angular Team are working on lots of bug fixes, new features to be added/update/remove/ re-introduce/ and many more things.

Let's start to explore all the changes of Angular 6 step by step!

```
ng update
```

Both require you to update your existing code.

To update to RxJS 6, you simply run:

```
npm install –save rxjs@6
```

Simply run the below command and update your existing Angular project:

```
npm install –save rxjs-compat
```

Alternatively, you can use the command - ng update rxjs to update RxJS and install the rxjs-compat package automatically.

RxJS 6 Related import paths:

Instead of:

```
import { Observable } from 'rxjs/Observable';
import { Subject } from 'rxjs/Subject';
```

Use a single import:

```
import { Observable, Subject } from 'rxjs';
```

So all from rxjs/Something imports become from one 'rxjs'

Operator imports have to change:

Instead of:

```
import 'rxjs/add/operator/map';
import 'rxjs/add/operator/throttle';
```

Now you can use:

```
import { map, throttle } from 'rxjs/operators';
```

And

Instead of:

```
import 'rxjs/add/observable/of';
```

Now you can use:

```
import { of } from 'rxjs';
```

RxJS 6 Changes - Changed Operator Usage -

Instead of:

```
import 'rxjs/add/operator/map';
import 'rxjs/add/operator/throttle';
yourObservable.map(data => data * 2)
.throttle(...)
.subscribe(...);
```

You can use the new pipe () method:

```
import { map, throttle } from 'rxjs/operators';
yourObservable
  .pipe(map(data => data * 2), throttle(...))
  .subscribe(...);
```

Now in Angular 6 new projects use “angular.json” file instead of “.angular-cli.json” file.

```
ng update @angular/cli –from=1 –migrate-only
```

The above command will help you to update your existing “.angular-cli.json” file to the new “angular.json” file.

The “angular.json” file contains the Properties:

For example, previously you are using

```
<template [ngIf]=”IsAdmin”>
  <p>This template renders only if IsAdmin is true.</p>
</template>
```

Now in Angular 6, you should use instead of

```
<ng-template [ngIf]=”IsAdmin”>
  <p>This template renders only if IsAdmin is true.</p>
</ng-template>
```

Example for marking a service as global:

Instead of

```
//my.service.ts
export class MyService { }
//In app.module.ts
//JavaScript imports services
import { MyService } from './my-serice.service';

//AppModule class with the @NgModule decorator
@NgModule({
  declarations: [],
   providers: [MyService] //My services instances are now
available across the entire app.
})
export class AppModule {
   //exporting app module
}
```

Use with Angular 6 released-

```
//my.service.ts
@Injectable({providedIn: 'root'})
export class MyService { }
@NgModule({
  declarations: [],
  providers: [] // Service does not need to be added here
})
export class AppModule {}
```

The second one obviously saves some lines of code as compared to the previous code.

Angular 6 introduces Angular Elements

The elements are a feature that allows you to compile Angular components to native web components which you can use in your Angular application.

An angular element is a package which is part of the Angular framework @angular/elements.

Angular 6 introduces new Ivy Renderer

The new Ivy renders and it's not stable for now and it's only in beta version. It will stable in future for production.

The main goal of Ivy render is to speed up its loading time and reduce the bundle size of your applications. Also for uses a different approach for rendering Angular components.

Ivy Renderer is new rendering engine which is designed to be backward compatible with existing render and focused to improve the speed of rendering and it optimizes the size of the final package.

For Angular, this will not be default renderer, but you can manually enable it in compiler options.

Bazel Compiler

The Bazel Complier is a build system used for nearly all software built at Google. From Angular 6 release, will start using the Bazel compiler support and when you compile the code with Bazel Compiler, you will recompile the entire code base, but it compiles only with necessary code.

The Bazel Complier uses advanced local and distributed caching, optimized dependency analysis and parallel execution.

Replace Context, Record, and Injectors:

Replace ngOutletContext with ngTemplateOutletContext

Replace CollectionChangeRecord with IterableChangeRecord

Now use Renderer2, Instead of Renderer

Now use StaticInjector, Instead of ReflectiveInjector

Angular 6 Renamed Operators

The lists of renamed operators are:

Angular 6 introduces multiple validators for array method of FormBuilder:

```
import { Component } from '@angular/core';
import {FormsModule, FormBuilder, FormGroup} from '@angular/
forms';
constructor(private fb: FormBuilder) {}
myForm: FormGroup;
ngOnInit() {
  this.myForm = this.fb.group({
      text: ['', Validators.required],
        options: this.fb.array([], [MyValidators.minCount,
MyValidators.maxCount])
  });
}
```

Previously -

```
<input [(ngModel)]="name" (ngModelChange)="onChange($event)">
```

And

```
onChange(value) {
  console.log(value);   // would log the updated value, not
old value
}
```

Now Use:

```
<input   #modelDir="ngModel"   [(ngModel)]="name"
(ngModelChange)="onChange(modelDir)">
```

And

```
onChange(NgModel: NgModel) {
   console.log(NgModel.value);// would log old value, not
updated value
}
```

Let's see in depths:

```
@ViewChild('your-element') yourElement:ElementRef;
```

Where to download the Angular 6?

For download Angular 6, kindly refer below link.

https://github.com/angular/angular/releases/

1.9 What's New in Angular 7?

What's New In Angular 7?

Angular 7 being smaller, faster and easier to use and it will be making developers life easier.

Angular 7 is a major release and expanding to the entire platform including core framework, Angular Material, and Angular CLI (stands for Command Line Interface).

Let's introduce added new features of Angular 7 -

Angular Compatibility Compiler (NGCC)

The Angular Compatibility Compiler (ngcc) is a tool which "upgrades" node_module compiled with non-ivy ngc into ivy compliant format.

This compiler will convert node*modules compiled with ngcc, into node*modules which appear to have been compiled with TSC compiler transformer (ngtsc) and these compiler conversions will allow such "legacy" packages to be used by the Ivy rendering engine.

TSC transformer which removes and converts @Pipe, @Component, @Directive and @NgModule to the corresponding definePipe, defineComponent, defineDirective and defineInjector.

Ivy rendering engine -

The Ivy rendering engine is a new backwards-compatible Angular renderer main focused on the following.

The template functions for creating dynamically views are no longer nested functions inside each other.

Now we use for loops that are nested inside other loops.

Example:

```
functionAppComponent(rf: RenderFlags, ctx: AppComponent) {
functionulTemplateFun(rf1: RenderFlags, ctx0: any) {
functionliTemplateFun(rf1: RenderFlags, ctx1: any) {...}
  }
}
```

No longer create multiple functions instances for loops that are nested inside other loops.

Example:

```
<ul *ngFor="let student of students">
<li *ngFor="let subject of student"> {{subject}} </li>
</ul>
```

To enabling Ivy by adding the following lines to the tsconfig.json file in the new project folder:

```
"angularCompilerOptions": {
"enableIvy": true
}
```

UrlSegment interface

The UrlSegment interface represents a single URL segment and the constructor, properties, and methods look like below UrlSegment class i.e.

```
classUrlSegment {
constructor(path: string, parameters: {...})
path: string
parameters: {...}
toString(): string
}
```

The UrlSegment is a part of a URL between the two slashes and it contains a path and matrix parameters associated with the segment.

Example:

```
@Component({
templateUrl:'./user.component.html',
styleUrls: ['./user.component.css']
})
classUserComponent {
constructor(router: Router) {
consturlTree: UrlTree = router.parseUrl('/user;id=101');
consturlSGroup:          UrlSegmentGroup          =
urlTree.root.children[PRIMARY_OUTLET];
consturlSegment: UrlSegment[] = urlSGroup.segments;

urlSegment[0].path; // It will returns 'user'
urlSegment[0].parameters; //It will returns {id: 101}
  }
}
```

DoBootstrap interface

Angular 7 added a new lifecycle hook that is called ngDoBootstrap and an interface that is called DoBootstrap.

Example:

```
//ngDoBootstrap - Life-Cycle Hook Interface
classAppModuleimplementsDoBootstrap {
ngDoBootstrap(appRef: ApplicationRef) {
appRef.bootstrap(AppComponent);
  }
}
```

KeyValuePipe

Chang, you object into an array of key-value pairs that output array will be ordered by keys.

By default, it will be by Unicode point value.

Syntax:

```
{{your_input_expression | keyvalue [:compareFn] }}
```

Example:

```
@Component({
selector:'key-value-pipe',
template:`<div>
<p>your custom Object</p>
<div *ngFor="let cust of customerObject | keyvalue">
      {{cust.key}}:{{cust.value}}
</div>
</div>`
})
exportclassKeyValuePipeComponent {
customerObject: {[key: number]: string} =
   {
      1:'Anil Singh',
      2:'Aradhaya Singh',
      3:'Reena Singh'
   };
 }
```

1.10 What is Bootstrapping (bootstrap) in Angular?

The Bootstrap is the root AppComponent that Angular creates and inserts into the “index.html” host web page.

```
<body>
  <app-root></app-root>
</body>
```

You can put more than one component tree on a host web page, that’s not typical. Most of the applications have only one component tree and they bootstrap a single root component and you can call the one root component you want but most developers call it AppComponent.

The bootstrapping process creates the components listed in the bootstrap array and inserts each one into the browser (DOM).

The Angular Module (NgModules) helps us to organize an application into connected blocks of functionality.

The NgModule properties for the minimum “AppModule” generated by the CLI which follows as:

app.module.ts -

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppComponent } from './app.component';
import { LoginComponent } from './login/login.component';
import { SignupComponent } from './signup/signup.component';

@NgModule({
  //declarations is used for configure the selectors.
  declarations: [
    AppComponent,
    LoginComponent,
    SignupComponent
  ],
  //Composability and Grouping
  //imports used for composing NgModules together.
  imports: [
    BrowserModule
  ],
  //Runtime or injector configuration
  //providers is used for runtime injector configuration.
  providers: [],

  bootstrap: [AppComponent]
})
export class AppModule { }
```

By default Bootstrap file is created in the folder “src/main.ts” and “main.ts” file is very stable. Once you have set it up, you may never change it again and its looks like -

```
import { enableProdMode } from '@angular/core';
import { platformBrowserDynamic } from '@angular/platform-
browser-dynamic';
import { AppModule } from './app/app.module';
import { environment } from './environments/environment';

if (environment.production) {
  enableProdMode();
}

platformBrowserDynamic().bootstrapModule(AppModule)
  .catch(err => console.log(err));
```

1.11 What Is Architecture Overview of Angular?

Angular is a most popular web development framework for developing mobile apps as well as desktop applications.

The Angular framework is also utilized in the cross-platform mobile development called IONIC and so it is not limited to web apps only.

Angular is an open source framework written and maintained by Angular team at Google and the Father of Angular is Misko Hevery .

The bootstrapping process creates the components listed in the bootstrap array and inserts each one into the browser (DOM).

With bits of the help of above pic, you can identify the seven main building blocks of an Angular Application, which are:

The basic building blocks of an Angular application are NgModules, which provide a compilation context for components.

Angular app is defined by a set of NgModules and it always has at least a root module that enables bootstrapping, and many more feature modules.

Introduction to Modules

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 purpose of the module is to declare everything you create in Angular and group them together.

The NgModule is used to simplify the ways you define and manage the dependencies in your applications and also you can consolidate different components and services into associative blocks of functionality.

Introduction to Components

Components are the most basic building block of a UI in Angular applications and it controls views (HTML/CSS). They also communicate with other components and services to bring functionality to your applications.

Introduction to Templates

The Template is just a subset of HTML, which tells Angular how to display the HTML view. Templates are created by using the normal HTML tags and also use the Angular-specific markup like interpolation or Property binding.

Most of all HTML syntax is valid template syntax.

A template expression produces a value. Angular executes the expression and assigns it to a property of a binding target. The target might be HTML elements, components, or directives.

The