**Difference Among Different Versions of Angular**

* **AngularJs**
  + A JavaScript MVW Framework
  + Extends HTML support by adding tags, attributes, and expressions
  + Allows easy event Handling
  + Supports for Data Binding
  + Built-In Template Engine and Routing
  + Form Validations and Animations
  + Dependencies Injection
  + AngularJS is completely based on controllersand the View communicates using $scope.
* **Angular 2**
  + Modern, faster, and highly scalable framework
  + Equally useful framework for web, mobile, and desktop apps
  + Web components based architecture
  + Supports Hierarchical Dependency Injection
  + Angular 2 is completely rewritten from scratch and as a result, the ways we write in AngularJS and Angular 2 are completely different.
  + Angular 2 is completely based on the component. Components are the building blocks of an Angular 2 application. The advantage of the Component-based approach is that it facilitates greater code reuse.
  + From a performance standpoint, Angular 2 is 5 times faster compared to AngularJS.
  + AngularJS doesn't support for mobile devices, whereas Angular 2 is designed by keeping in mind to support mobile devices.
  + Angular 2 has more language choices (TypeScript, JavaScript, Dart, PureScript and Elm etc.)
  + In Angular 2 only TypeScript 1.8 version was supported
  + There is no animation feature offers in Angular 2.
  + The code generated using Angular 2 is bigger, and the file size is also larger.
* **Angular 4**
  + Angular 4 simply is the next version of Angular 2.
  + In Angular 4, a lot of improvements made to reduce the size of the AOT (Ahead-of-time) compiler generated code.
  + In Angular 4, it supports TypeScript 2.1 and TypeScript 2.2 compatibility, which means now we can use all new features supported in TypeScript 2.1 and TypeScript 2.2 can be used in Angular 4 application.
  + The Animation features are separated from @angular/core package and moved them to new packages. By this way, if we don't import animation packages into your application then the main bundle size will be reduced and gives the performance improvement. As I told you @angular/core package and moved them to new packages, so if you are using these packages in your Angular 2 application then at the time of moving from Angular 2 to Angular 4, we need to change the package reference as well.
  + Angular 4.0 has reduced the bundled file size by 60%. Thus code generated is reduced which helps to accelerate the application performance.
  + In Angular 4, else block newly introduced. I mean, along with \*ngif, we can use elseblock as well. Example : -
  + In Angular 2 we were writing like,

<div \*ngIf="yourCondition">

<h2>Condition **true**!</h2>

</div>

<div \*ngIf="!yourCondition">

<h2>Condition **false**!</h2>

</div>

* + Now you can rewrite the same in Angular 4,

<div \*ngIf="yourCondition; else myFalsyTemplate">

<h2>Condition **true**!</h2>

</div>

<ng-template #myFalsyTemplate>

<h2>Condition **false**!</h2>

</ng-template>

* **Angular 5**
  + Make AOT the default
  + Easier to build progressive web apps
  + Type checking in templates
  + Support for Internationalized Number, Date, and Currency Pipes
  + An update to Httpclient
  + Zone speed improvements
  + New Router Lifecycle Events

In Angular 5 few new life cycle events being added to the router and those are ActivationStart, ActivationEnd,  ChildActivationStart, ChildActivationEnd, GuardsCheckStart, GuardsCheckEnd, ResolveStart and ResolveEnd.

* + **Build optimizer**  
    It helps to removed unnecessary code from your application.
  + **Angular Universal State Transfer API and DOM Support**  
    By using this feature, we can now share the state of the application between the server side and client side very easily.
  + **Compiler Improvements**  
    This is one of the very nice features of Angular 5, which improved the support of incremental compilation of an application.
  + **Preserve Whitespace**  
    In earlier versions of Angular, unnecessary new lines, tabs and white spaces were created during the build. Now, in Angular 5, the decision is in your hands whether you need them or not. Angular 5 supports to restrict them (newlines, tabs, and white spaces) in both, the application level or you can restrict them individual component level where you wish to restrict.
  + Example If you want to restrict them only for TestComponent  then below is the sample code,

@Component({

            templateUrl: 'test.component.html',

            preserveWhitespaces: **false**

        }

**export** **class** TestComponent {}

* + If you want to restrict them throughout the application level then we have add the below lines of code in tsconfig.json file.

"angularCompilerOptions": {

    "preserveWhitespaces": **false**

}

* + **Increased the standardization across all browsers**  
    In earlier versions of Angular, we were depending on i18n whenever we wanted to support internationalization in our application.  In Angular 5 now no need to depend on i18n, it provides a new date, number, and currency pipes which increases the internationalization across all the browsers and eliminates the need of i18n polyfills.
  + **HttpClient**  
    **The HttpClient**  
    One of the greatest anticipated changes from version 4.3 was being able to say goodbye to the Http library.

Instead, they introduced the HttpClient API, which is faster, more secure and efficient than its predecessor. Although this API came with the 4.3 version update, as of Angular 5, Http library was depreciated.

**Some Great Benefits of the HTTPCLIENT API**

* + Response body access included support for JSON types and was typed synchronous.
  + JSON became an assumed default and no longer had to be parsed explicitly.
  + With the use of Interceptors, you could use middleware logic and insert it in the pipeline.
  + Request/response objects were immutable.
  + Request upload and response download could utilize progress events.
  + **Improved Decorator Support**  
    In Angular 5, we can use lambda expressions instead of naming functions. Sample code snippet as below,
  + Before,

Component({

  provider: [{

        provide: 'my-service',

        useValue: testMethod()

    }]

})

**export** **class** CustomClass {}

* + Now in Angular 5,

Component({

            provider: [{

                provide: ''

                my - service ', useFactory: () => **null**}]

            })

**export** **class** CustomClass {}

* + Angular 5 supports TypeScript 2.3 version.
* **Angular 6**
  + The highlights of Angular 6 include the Angular Command Line Interface (CLI), The Component Development KIT (CDK) and the Angular Material package update.
  + **The Angular Material Design Library**  
    A new Tree component is now added in the Angular Material Design Package and the Component Dev Kit. It allows you to visualize tree structures in a more hierarchical order, like a list of files, for example. These new tree components come in both styled and un-styled versions, (Material’s mat-tree) and (CDK’s cdk-tree) respectively.
  + **Angular Elements**  
    Remember the Elements package? Angular 6 fully supports it now. What it did was allow us to use Angular components outside of Angular like in JQuery or VueJS apps.
  + **Component Dev Kit (CDK)**

With the CDK you can now build your own library of UI components without using the Angular Material library. It also supports Responsive Web Design layouts so you don't have to use other libraries like Flex Layout or even learn using the CSS Grid. It covers them all.

* + **Command Line Interface (CLI)**  
    The Angular command-line interface is now equipped with new commands such as ng-update , which updates dependencies and code, and ng-add , which helps quickly add application features and also supports turning applications into progressive web apps.
  + **Multiple Validators For Your Forms**  
    Those of you who had to fuss about passing more than one validator in your Formbuilders, your prayers have been answered because Angular 6 now allows you to pass multiple validators to the formBuilder.
* **Angular 7**
  + **CLI prompts**

In Angular 7, the CLI prompts have been updated to v7.0.2 with new features. For instance, it will now prompt users when typing commands like @angular/material, ng-new, and ng-add to help them discover the in-built SCSS support, routing, and more.

### **Angular material & component dev kit (CDK)**

The Angular 7 introduced minor visual updates & improvements in Material Design that earlier received a [major update](https://www.youtube.com/watch?v=1Dh8ZBQp9jo) this year only.

In addition, refresh, virtual scrolling, large lists of data, dynamic loading and unloading of parts of the DOM also were the part of improvements in CDK and Angular Material.

### **Drag & drop**

The new drag-drop module basically provides a better way to easily create drag & drop interfaces, which is backed by sorting within a list, support for free dragging, animations, custom drag handles, transferring items between lists, previews, and placeholders.

### **Virtual scrolling**

Like mentioned earlier, the new Virtual Scrolling in Angular 7 basically loads and unloads items from the DOM depending upon visible parts of lists, resulting into a much faster experiences for users having huge scrollable lists.

## **Upgrading requirements**

The Upgrading process is really simple. For most Angular apps out there that are running on Angular 6 and RxJS 6, you only need to run a single command stated below to upgrade to Angular 7.



* + If, however, you’re using Angular Material, use following single line command and you’re good to go!

