**What is Angular 2? 13 Best Advantages for Angular 2!**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [12:30 PM](https://www.code-sample.com/2016/06/what-is-angular-2.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=6136586108537040539&from=pencil)

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

**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**](http://misko.hevery.com/about/).

By Angular Developer Guide book - “*Angular is a platform and framework for building client applications in HTML and TypeScript. Angular is itself written in TypeScript. It implements core and optional functionality as a set of TypeScript libraries that you import into your apps*.”

The “**Angular 2**” is focusing on **data-binding**, extensible **HTML** and on application test-ability but it is still in design and prototyping stage.

  Stayed Informed - [What Is Angular 4 or 5 or 6?](https://www.code-sample.com/2018/01/what-is-angular-4-or-5-or-6.html)

  Stayed Informed – [Angular 2 vs. ReactJs](https://www.code-sample.com/2016/07/angular-2-vs-react.html)  and [13 Best Advantages for Angular2](https://www.code-sample.com/2016/06/angular-2-vs-angular-1-performance.html)

Angular **framework** helps us to build **client** applications in **HTML** and **JavaScript**.

Angular 2 is so **simpler**, **faster**, **modular** and **instrumented** design.

Angular 2 targeting to modern browsers and it is developing using **ES6** (ES6 is called **ECMAScript** version 6). It also support to **ECMAScript** version 5(**ES5**).

You don’t worry about the versions of **ECMAScript**. The **ES6** compiler manages to the versioning related problems.

All the **Angular 2** framework code is already being written in ECMAScript 6.

The set of modern browsers are

1.              Chrome

2.              Firefox

3.              Opera

4.              Safari

5.              IE Version10, 11 and so on...

On mobiles, it is supporting to the list of Chrome on Android, iOS 6+, Windows Phone 8+ and Fire-Fox mobile and also trying to support to older versions of Android.

Angular 2 team working with Traceur compiler team to provide the support to build some extensions. This set of extensions called “**ES 6 +A**”.

The “**Angular 2**” is using “**Traceur**” compiler. **Traceur** is a compiler that takes “**ES6**” and compiles it down (**ES5**) to regular **JavaScript** that runs in your browsers. It is run everywhere you want to do.  
  
-

**Introduction to Angular 2 [A Most Popular JS Framework]**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [9:18 PM](https://www.code-sample.com/2016/06/introduction-to-angular-2-fundamentals.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=8500190761968704552&from=pencil)

**Angular 2 is a most popular framework for developing mobile apps**.  It is also for desktop as well mobile applications.    
  
Now, Angular 4 is development phase you know very well and hopefully you love.

|  |  |  |  |
| --- | --- | --- | --- |
| [Angular 2 vs. Angular 1](https://www.code-sample.com/2016/06/angular-2-vs-angular-1-performance.html) | [Angular 4 vs. Angular 2](https://www.code-sample.com/2017/03/angular-4-vs-angular-2-difference.html) | [Angular 5 vs. Angular 4](https://www.code-sample.com/2017/09/angular-5-vs-angular-4-whats-new-in.html) | [Angular 6 vs Angular 5](https://www.code-sample.com/2017/11/angular-5-vs-angular-6-new-features.html) |

The **Angular 2** is focusing on **data-binding**, extensible **HTML** and on application test-ability but it is still in design and prototyping stage.

**Angular framework** helps us to build client applications in **HTML** and **JavaScript**.

**Angular 2** is so **simpler**, **faster**, **modular** and instrumented design.

**Angular 2** targeting to modern browsers and it is developing using **ES6** (**The ES6 is called ECMAScript version 6**). It also support to **ECMAScript** version 5(**ES5**).

You don’t worry about the versions of ECMAScript. The compiler manages to the versioning related problems.

All the **Angular 2 framework** code is already being written in ECMAScript 6.

The set of modern browsers are

1.       Chrome

2.       Firefox

3.       Opera

4.       Safari

5.       IE Version10 and 11.

Stayed Informed – [13 Best Advantages for Angular 2](https://www.code-sample.com/2016/06/angular-2-vs-angular-1-performance.html)

On mobiles, it is supporting to the list of Chrome on **Android**, **iOS** 6+, **Windows Phone** 8+ and **Fire-Fox** mobile and also trying to support to older versions of **Android**.

The **Angular 2** is using **Traceur**compiler to generate the nice **ECMAScript** 5/6 that runs everywhere you want to do.

**Angular 2** team working with Traceur compiler team to provide the support to build some extensions. This set of extensions called “**ES 6 +A**”.

**What is ECMAScript ES5/ES6?**

The **ECMAScript** is a scripting language which is developed by **Ecma International Org.**

Currently **ECMAScript**available in multiple versions that are **ES5 and ES6** and both of versions fully supported to **Chrome**, **Firefox**, **Opera**, **Safari**, and **IE** etc.

**What is Traceur Compiler?**

The “**Traceur”**is a JavaScript**compiler**. The **Traceur**compiler is very popular now days use to allow use to use the features from the future. This compiler is fully supported to **ES5**, **ES6** and also to **vNext**. The main goal of **Traceur**compiler is to inform to design of new JavaScript features and wrote the programming code of new efficient and good manners.

**What is Advantages of Angular 2?**

1.       There is many more advantage of Angular 2.

2.       The Angular 2 has better performance.

3.       The Angular 2 has more powerful template system.

4.       The Angular 2 provide simpler APIs, lazy loading and easier to application debugging.

5.       The Angular 2 much more testable.

6.       The Angular 2 provides to nested level components.

7.       The Angular 2 execute run more than two programs at the same time.

The Angular 2 architecture diagram identifies the eight main building blocks as.

1.       [Module](https://www.code-sample.com/2017/04/angular-2-ngmodel-root-export-module.html)

2.       [Component](https://www.code-sample.com/2016/06/angular-2-template-components.html)

3.       [Template](https://www.code-sample.com/2016/06/angular-2-template-components.html)

4.       [Outpouts](https://www.code-sample.com/2016/06/angular-2-outputs.html)

5.       [Data Binding](https://www.code-sample.com/2016/06/angular-2-template-components.html)

6.       [Directive](https://www.code-sample.com/2016/06/angular-2-directives-components.html)

7.       [Service](https://www.code-sample.com/2017/05/angular-2-services-singleton-examples.html)

8.       [Dependency Injection](https://www.code-sample.com/2016/04/dependency-injection-in-angular-2.html)

The Angular 2 framework consists of several libraries, the some of them working as core and some are optional.

**What Are The New Features Of Angular 2? Why You Used Angular 2?**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [10:19 PM](https://www.code-sample.com/2017/06/new-features-of-angular-2.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=697489297963135113&from=pencil)

**Angular 2 Features –**

       Angular 2 is Entirely Component Based

       Directives

       Dependency Injection

       Used of TypeScript

       Used of Lambdas or Arrow functions

       Generics

       Forms and Validations

       [And So on.....](https://www.code-sample.com/2016/06/angular-2-vs-angular-1-performance.html)

**Component Based**- It is entirely component based. It is not used to scope and controllers and Angular 2 are fully replaced by components and directives.

**Directives**- The directive can be declared as @Directive annotation.

A component is a directive with a template and the @Component decorator is actually a @Directive decorator extended with template oriented features.

**Dependency Injection**- Dependency Injection is a powerful pattern for managing code dependencies. There are more opportunities for component and object based to improve the dependency injection.

**Use of TypeScript**- Type represents the different types of values which are using in the programming languages and it checks the validity of the supplied values before they are manipulated by your programs.

**Generics**- TypeScript has generics which can be used in the front-end development.

**Lambdas and Arrow functions** – In the TypeScript, lambdas/ arrow functions are available. The arrow function is additional feature in typescript and it is also known as a lambda function.

**Forms and Validations**- Angular 2 forms and validations are an important aspect of front-end development.

**Why You Used Angular 2?**

1.     It is entirely component based.

2.     Better change detection

3.     Angular2 has better performance.

4.     Angular2 has more powerful template system.

5.     Angular2 provide simpler APIs, lazy loading and easier to application debugging.

6.     Angular2 much more testable

7.     Angular2 provides to nested level components.

8.     Ahead of Time compilation (AOT) improves rendering speed

9.     Angular2 execute run more than two programs at the same time.

10.  Angular1 is controllers and $scope based but Angular2 is component based.

11.  The Angular2 structural directives syntax is changed like ng-repeat is replaced with \*ngFor etc.

12.  In Angular2, local variables are defined using prefix (#) hash. You can see the below \*ngFor loop Example.

13.  TypeScript can be used for developing Angular 2 applications

14.  Better syntax and application structure

**13 Best Advantages for Angular2 - [Angular 2 vs. Angular 1]**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [11:28 PM](https://www.code-sample.com/2016/06/angular-2-vs-angular-1-performance.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=5312211851424786865&from=pencil)

Why should you use Angular 2 ? What are the Advantages of Angular 2 ?

|  |  |  |  |
| --- | --- | --- | --- |
| [Angular 2 vs. Angular 1](https://www.code-sample.com/2016/06/angular-2-vs-angular-1-performance.html) | [Angular 4 vs. Angular 2](https://www.code-sample.com/2017/03/angular-4-vs-angular-2-difference.html) | [Angular 5 vs. Angular 4](https://www.code-sample.com/2017/09/angular-5-vs-angular-4-whats-new-in.html) | [Angular 6 vs Angular 5](https://www.code-sample.com/2017/11/angular-5-vs-angular-6-new-features.html) |

The core differences and many more advantages on Angular 2 vs. Angular 1 as following,

1.     It is entirely component based.

2.     Better change detection

3.     Angular2 has better performance.

4.     Angular2 has more powerful template system.

5.     Angular2 provide simpler APIs, lazy loading and easier to application debugging.

6.     Angular2 much more testable

7.     Angular2 provides to nested level components.

8.     Ahead of Time compilation (AOT) improves rendering speed

9.     Angular2 execute run more than two programs at the same time.

10.  Angular1 is controllers and $scope based but Angular2 is component based.

11.  The Angular2 structural directives syntax is changed like ng-repeat is replaced with \*ngFor etc.

12.  In Angular2, local variables are defined using prefix (#) hash. You can see the below \*ngFor loop Example.

13.  TypeScript can be used for developing Angular 2 applications

14.  Better syntax and application structure

There are more advantages over performance, template system, application debugging, testing, components and nested level components.

**For Examples as,**

**Angular 1 Controller:-**

**var** app = angular.module("userApp", []);

app.controller("productController", **function**($scope) {

$scope.users = [{ name: "Anil Singh", Age:**30**, department :"IT" },

{ name: "Aradhya Singh", Age:**3**, department :"MGMT" }];

});

**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:**30**, department :"IT" },

{ name: "Aradhya Singh", Age:**3**, department :"MGMT" }];

}

**Bootstrapping in Angular 1 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 Angular2 structural directives syntax is changed like **ng-repeat** is replaced with **\*ngFor** etc.

**For example as,**

//Angular 1,

<div ng-repeat="user in users">

Name: {{user.name}}, Age : {{user.Age}}, Dept: {{user.Department}}

</div>

//Angular2,

<div \*ngFor="let user of users">

Name: {{user.name}}, Age : {{user.Age}}, Dept: {{user.Department}}

</div>

**What is AOT Compilation? - Pros and Cons of Ahead-of-Time!**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [11:52 PM](https://www.code-sample.com/2017/06/angular-2-aot-compilation-advantages.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=398217492529267984&from=pencil)

**What is AOT compilation? Why Use in Angular 2?**

AOT compilation stands for “Ahead of Time compilation” and it are used to compiles the angular components and templates to native JavaScript and HTML during the build time instead of run-time.

The compiled HTML and JavaScript are deployed to the web server so that the compilation and render time can be saved by the browser. It is the big advantage to improve the performance of applications.

**Advantages of AOT -**

1.         **Faster download**: - The Angular 2 app is already compiled so it is faster.

2.         **Faster Rendering**: - If the app is not AOT compiled and the compilation process happens in the browser once the application is fully loaded. This has a wait time for all necessary components to be downloaded and then the time taken by the compiler to compile the app. With AOT compilation, this is optimized.

3.         **Lesser Http Requests**: - It is supporting to the lazy loading. Actually, lazy loading is great concepts for sending HTTP request to the server. It is minimise the multiple requests for each associated html and css, there is a separate request goes to the server.

4.         **Detect error at build time**: - In Angular 2, the compilation happens beforehand and most of the errors can be detected at the compile time and this process providing us a better application’s stability.

**Disadvantages of AOT -**

1.         AOT only works only with HTML and CSS and not for other file types. If required other file types that time we will need to follow the previous build step.

2.         We need to maintain AOT version of bootstrap file.

3.         We need to clean-up step before compiling.

**Angular 2 - What is Lazy Loading and How to enable Lazy Loading?**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [11:30 PM](https://www.code-sample.com/2017/06/angular-2-lazy-loading.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=3502726505318566376&from=pencil)

**What is lazy loading and How to enable lazy loading in angular 2?**

**Lazy Loading** - Lazy loading enables us to load only the module user is interacting and keep the rest to be loaded at run-time on demand.

Lazy loading speeds up the application initial load time by splitting the code into multiple bundles and loading them on demand.

1.         Each and every Angular2 application must have one main module that is called “AppModule” and your code should be splitted into various child modules based on your applications.

2.         We do not require to import or declare lazily loading module in root module.

3.         Add the route to top level routing and takes routes array and configures the router.

4.         Import module specific routing in the child module.

5.         And so on.

**How would you Optimize the Angular 2 Application for Better Performance?**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [11:43 PM](https://www.code-sample.com/2017/06/optimize-angular-2-application.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=2994117979166560331&from=pencil)

The optimizations are depends on the size of applications, type and other factors but normally we consider following optimizing points i.e.

1.         Consider AOT compilation.

2.         Consider lazy loading instead of fully bundled app if the app size is more.

3.         Keep in mind, your application is bundled and disfeatured.

4.         Keep in mind, your application doesn’t have un-necessary import statements.

5.         Keep in mind, your application’s 3rd party unused library. If exist and not used, removed from your application.

6.         Remove your application dependencies if not required.

**What are the Securities Threats should we be Aware of in Angular 2 Applications?**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [11:34 PM](https://www.code-sample.com/2017/06/angular-2-securities-threats.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=2188088907972071580&from=pencil)

As like your other web applications, you should flow in angular 2 applications also.

There are some basic guidelines to mitigate the security risks.

1.     Consider using AOT compilation.

2.     Try to avoid using or injecting dynamic HTML content to your component.

3.     Try to avoid using external URLs if not trusted.

4.     Try to prevent XSRF attack by restricting the REST APIs.

If you are using external resources like HTML, CSS, which is coming from outside the application in case you follow best practice/cleanly your apps.

**Stayed Informed** – [Angular 2 Tutorials | Angular 2 Quick Start Docs](https://www.code-sample.com/2015/07/angularjs-2-documentation-with-example.html)

**13 Best Advantages for Angular2 - [Angular 2 vs. Angular 1]**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [11:28 PM](https://www.code-sample.com/2016/06/angular-2-vs-angular-1-performance.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=5312211851424786865&from=pencil)

Why should you use Angular 2 ? What are the Advantages of Angular 2 ?

|  |  |  |  |
| --- | --- | --- | --- |
| [Angular 2 vs. Angular 1](https://www.code-sample.com/2016/06/angular-2-vs-angular-1-performance.html) | [Angular 4 vs. Angular 2](https://www.code-sample.com/2017/03/angular-4-vs-angular-2-difference.html) | [Angular 5 vs. Angular 4](https://www.code-sample.com/2017/09/angular-5-vs-angular-4-whats-new-in.html) | [Angular 6 vs Angular 5](https://www.code-sample.com/2017/11/angular-5-vs-angular-6-new-features.html) |

The core differences and many more advantages on Angular 2 vs. Angular 1 as following,

1.     It is entirely component based.

2.     Better change detection

3.     Angular2 has better performance.

4.     Angular2 has more powerful template system.

5.     Angular2 provide simpler APIs, lazy loading and easier to application debugging.

6.     Angular2 much more testable

7.     Angular2 provides to nested level components.

8.     Ahead of Time compilation (AOT) improves rendering speed

9.     Angular2 execute run more than two programs at the same time.

10.  Angular1 is controllers and $scope based but Angular2 is component based.

11.  The Angular2 structural directives syntax is changed like ng-repeat is replaced with \*ngFor etc.

12.  In Angular2, local variables are defined using prefix (#) hash. You can see the below \*ngFor loop Example.

13.  TypeScript can be used for developing Angular 2 applications

14.  Better syntax and application structure

There are more advantages over performance, template system, application debugging, testing, components and nested level components.

**For Examples as,**

**Angular 1 Controller:-**

**var** app = angular.module("userApp", []);

app.controller("productController", **function**($scope) {

$scope.users = [{ name: "Anil Singh", Age:**30**, department :"IT" },

{ name: "Aradhya Singh", Age:**3**, department :"MGMT" }];

});

**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:**30**, department :"IT" },

{ name: "Aradhya Singh", Age:**3**, department :"MGMT" }];

}

**Bootstrapping in Angular 1 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 Angular2 structural directives syntax is changed like **ng-repeat** is replaced with **\*ngFor** etc.

**For example as,**

//Angular 1,

<div ng-repeat="user in users">

Name: {{user.name}}, Age : {{user.Age}}, Dept: {{user.Department}}

</div>

//Angular2,

<div \*ngFor="let user of users">

Name: {{user.name}}, Age : {{user.Age}}, Dept: {{user.Department}}

</div>

**7 Best Key Differences - Constructor Vs. ngOnInit [Angular 2]**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [11:21 AM](https://www.code-sample.com/2017/02/angular-2-constructor-vs-oninit.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=4585252370315772382&from=pencil)

**Angular 2 Constructors:-**

1.      The **constructor** is a default method runs when component is being constructed.

2.      The constructor is a typescript **feature** and it is used only for a class **instantiations** and nothing to do with Angular 2.

3.      The constructor called first time before the **ngOnInit**().

**Stayed Informed** - [13 Best Advantages for Angular 2](https://www.code-sample.com/2016/06/angular-2-vs-angular-1-performance.html)

**Angular 2 ngOnInit**:-

1.      The **ngOnInit** event is an Angular 2 life-cycle event method that is called after the first ngOnChanges and the ngOnInit method is use to parameters defined with @**Input** otherwise the constructor is **OK**.

2.      The **ngOnInit** is called after the constructor and ngOnInit is called after the first ngOnChanges.

3.      The **ngOnChanges** is called when an input or output binding value changes.

**Example as,**

**import** {Component, OnInit} from '@angular/core';

**export** **class** App **implements** OnInit{

**constructor**(){

}

ngOnInit(){

}

}

[When will ngInit be called? How would you make use of onNgInit()?](https://www.code-sample.com/2017/02/angular-2-ngoninit-and-ng-init.html)

I hope you are enjoying with this post! Please share with you friends!! Thank you!!!

**When will ngInit be called? How would you make use of ngOnInit()?**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [10:39 AM](https://www.code-sample.com/2017/02/angular-2-ngoninit-and-ng-init.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=6466839795068590699&from=pencil)

In Angular 1.x, **ngInit** is called when template is re-rendered. In other words “**ng-init**” is called, when I take turns back to a page.

In Angular2, there is no “**ng-init**” but we can create a ways like this using the directive and ngOnInit class. Angular 2 provides life cycle hook **ngOnInit** by default.

The **ngOnInit** is invoked when the component is initialized and invoked only once when the directive is instantiated. It is a best practice to implement these life-cycle interfaces.  
  
**Stayed Informed** - [13 Best Advantages for Angular 2](https://www.code-sample.com/2016/06/angular-2-vs-angular-1-performance.html)

According to Angular2 Doc, “The **ngOnInit** is called right after the directive's data-bound properties have been checked for the first time, and before any of its children have been checked. It is invoked only once when the directive is instantiated.”

For example as,

**import** { Directive, Input } from '@angular/core';

@Directive({

selector: '[ngInit]'

})

**class** NgInit {

@Input() ngInit;

ngOnInit() {

**if**(**this**.ngInit) { **this**.ngInit(); }

}

}

In template as following,

<div \*ngIf="Timer.dateTime === currentDateTime">

<div \*ngIf="Timer.checked" [ngInit]="Start"></div>

<div \*ngIf="!Timer.checked" [ngInit]="Stop"></div>

</div>

I hope you are enjoying with this post! Please share with you friends!! Thank you!!!

**Angular 2  Component Lifecycle Hooks [Examples Also]**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [11:14 PM](https://www.code-sample.com/2017/02/angular-2-component-lifecycle-hooks.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=9107682600641554920&from=pencil)

The common questions ask bye most of Angular 2 lovers,

“**Could anyone tell me about the usage of ngOnInit if we already have a constructor?**” but Angular 2 provides life cycle hook ngOnInit by default.

Angular 2 Components and Directives has multiple life-time hooks where we custom logic can be executed.

**Stayed Informed** - [7 Best Key Differences [Constructor Vs. ngOnInit]](https://www.code-sample.com/2017/02/angular-2-constructor-vs-oninit.html)

**Angular 2 Constructors**:-

The constructor is a default method runs when component is being constructed.

The constructor is a typescript feature and it is used only for a class instantiations and nothing to do with Angular 2.

The constructor called first time before the ngOnInit().

**Example as**,

**import** {Component} **from** 'angular2/core';

**import** {UserService} **from** './userService';

**@Component**({

selector: ‘list-user’,

template: `<ul><li \*ngFor="#user of users">{{user.name}}</li></ul>`

})

**class** **App\_Component** {

users:Array<any>;

constructor(private \_userService: UserService) {

this.users = \_userService.getUsers();

}

}

**Angular 2 ngOnInit and ngOnChanges:-**

The ngOnInit event is an Angular 2 life-cycle event method that is called after the first ngOnChanges and the ngOnInit method is use to parameters defined with @Input otherwise the constructor is OK.

The ngOnInit is called after the constructor and ngOnInit is called after the first ngOnChanges.

The ngOnChanges is called when an input or output binding value changes.

**Examples as**,

**import** {Component, OnInit} **from** '@angular/core';

export **class** **App** implements OnInit{

constructor(){

}

ngOnInit(){

}

}

**Angular 2 ngOnDestroy** :-

The ngDestroy directive is called in a component lifecycle just before the instance of the component is finally destroyed.

**Example as,**

@Directive({

selector: '[destroyDirective]'

})

**export** **class** OnDestroyDirective **implements** OnDestroy {

//Call Constructor and set hello Msg.

**constructor**() {

**this**.helloMsg = window.setInterval(() => alert('Hello, I am Anil'), **2000**);

}

//Destroy to the component

ngOnDestroy() {

window.clearInterval(**this**.helloMsg);

}

}

**Angular 2 Complete lifecycle hook interface inventory**:-

1.      ngOnChanges - called when an input binding value changes.

2.      ngOnInit - after the first ngOnChanges.

3.      ngDoCheck - after every run of change detection.

4.      ngAfterContentInit - after component content initialized.

5.      ngAfterContentChecked - after every check of component content.

6.      ngAfterViewInit - after component's view(s) are initialized.

7.      ngAfterViewChecked - after every check of a component's view(s).

8.      ngOnDestroy - just before the component is destroyed.

**Angular 2 Lifecycle Events Log**:-

1.      onChanges

2.      onInit

3.      doCheck

4.      afterContentInit

5.      afterContentChecked

6.      afterViewInit

7.      afterViewChecked

8.      doCheck

9.      afterContentChecked

10. afterViewChecked

11. onChanges

12. doCheck

13. afterContentChecked

14. afterViewChecked

15. onDestroy

**What is the Best way to Declare and Access a Global Variable in Angular 2?**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [2:08 AM](https://www.code-sample.com/2017/06/angular-2-declare-global-variable.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=6660490743279003859&from=pencil)

This post helps us to learn “Declare and Access a Global Variable in Angular 2” using “Typescript” and also share the steps to create and use of this global variables.

**Stayed Informed -** [Angular 2 Tutorials and Examples](https://www.code-sample.com/2015/07/angularjs-2-documentation-with-example.html)

**Steps –**

1.     Create Global Variables.

2.     Import and Use the Global Variables in the Component.

3.     Result

**Create Global Variables :- “app.global.ts”**

import { Injectable } from '@angular/core';

@Injectable()

export class AppGlobals {

    readonly baseAppUrl: string = 'http://localhost:57431/';

    readonly baseAPIUrl: string = 'https://api.github.com/';

}

**Import and Use the Global Variables in the Component:- “user.component.ts”**

import { Component, Injectable} from '@angular/core';

import { CommonModule } from '@angular/common';

import { HttpModule, Http } from '@angular/http';

import { UserService } from '../service/user.service';

**import { AppGlobals } from '../shared/app.globals';**

@Component({

    selector: 'user',

    templateUrl: './user.component.html',

    styleUrls: ['./user.component.css'],

**providers**: [UserService, **AppGlobals**]

})

export class UserComponent {

    //USERS DECLARATIONS.

    users = [];

    //HOME COMPONENT CONSTRUCTOR

    constructor(private userService: UserService, **private \_global: AppGlobals**) { }

    //GET USERS SERVICE ON PAGE LOAD.

    ngOnInit() {

        this.userService.getAPIUsers(**this.\_global.baseAPIUrl** + 'users/hadley/orgs').subscribe(data => this.users = data);

        this.userService.getAppUsers(**this.\_global.baseAppUrl** + 'api/User/GetUsers').subscribe(data => console.log(data));

    }

}

//END BEGIN – USERCOMPONENT

**“user.server.ts” :-**

import { Injectable, InjectionToken } from '@angular/core';

import { Http, Response } from '@angular/http';

import 'rxjs/add/operator/map';

//BEGIN-REGION - USERSERVICE

@Injectable()

export class UserService {

    constructor(private \_http: Http) {

    }

    getAPIUsers(apiUrl) {

        return this.\_http.get(apiUrl).map((data: Response) => data.json());

    }

    getAppUsers(apiUrl) {

        return this.\_http.get(apiUrl).map((data: Response) => data);

    }

}

//END BEGIN – USERSERVICE

**Result:-**

**What's New in Angular 4? [Angular 4 New Features]**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [8:45 PM](https://www.code-sample.com/2017/04/whats-new-in-angular-4-new-features.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=4140705373180602777&from=pencil)

Angular 4 contains some additional Enhancement and Improvement. Consider the following enhancements.

1.      Smaller & Faster Apps

2.      View Engine Size Reduce

3.      Animation Package

4.      NgIf and ngFor Improvement

5.      Template

6.      NgIf with Else

7.      Use of AS keyword

8.      Pipes

9.      HTTP Request Simplified

10.   Apps Testing Simplified

11.   Introduce Meta Tags

12.   Added some Forms Validators Attributes

13.   Added Compare Select Options

14.   Enhancement in Router

15.   Added Optional Parameter

16.   Improvement Internationalization

**1. Smaller & Faster Apps** - [Angular 4](https://www.code-sample.com/2017/04/angular-4-interview-questions-and.html) applications is smaller & faster in comparison with Angular 2.

**2. View Engine Size Reduce**- Some changes under to hood to what **AOT** generated code compilation that means in Angular 4, improved the compilation time. These changes reduce around 60% size in most cases.

**3. Animation Package**- Animations now have their own package i.e. @angular/platform-browser/animations

**4. Improvement** - Some Improvement on \***ngIf** and \***ngFor**.

**5. Template** - The template is now **ng-template**. You should use the “ng-template” tag instead of “**template**”. Now Angular has its own template tag that is called “ng-template”.

**6. NgIf with Else** – Now in Angular 4, possible to use an else syntax as,

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

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

**7. AS keyword** – A new addition to the template syntax is the “**as** keyword” is use to simplify to the “**let**” syntax.

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 an observable, you can now use to write,

<div \*ngIf="users | async as usersModel">

<h2>{{ usersModel.name }}</h2> <small>{{ usersModel.age }}</small>

</div>

**8. Pipes** - Angular 4 introduced a new “**titlecase**” pipe “|” and use to changes the first letter of each word into the uppercase.   
  
The example as,

<h2>{{ 'anil singh' | titlecase }}</h2>

<!-- OUPPUT - It will display 'Anil Singh' -->

**9. Http** - Adding search parameters to an “**HTTP** request” has been simplified as,

//Angular 4 -

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

//Angular 2-

**const** params = **new** URLSearchParams();

params.append('sort', 'ascending');

http.get(`${baseUrl}/api/users`, { search: **params** });

**10. Test**- Angular 4, overriding a template in a test has also been simplified as,

//Angular 4 -

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

//Angular 2 -

TestBed.overrideComponent(UsersComponent, {

set: { template: '<h2>{{users.name}}</h2>' }

});

**11. Service**- A new service has been introduced to easily get or update “**Meta Tags**” i.e.

@Component({

selector: 'users-app',

template: `<h1>Users</h1>`

})

**export** **class** UsersAppComponent {

**constructor**(meta: **Meta**) {

meta.addTag({ name: 'Blogger', content: 'Anil Singh' });

}

}

**12. Forms Validators** - One new validator joins the existing “required”, “minLength”, “maxLength” and “pattern”. An email helps you validate that the input is a valid email.

**13. Compare Select Options** - A new “**compareWith**” directive has been added and it used to help you compare options from a select.

<select [compareWith]="byUId" [(ngModel)]="selectedUsers">

<option \*ngFor="let user of users" [ngValue]="user.UId">{{user.name}}</option>

</select>

**14. Router** - A new interface “**paramMap**” and “**queryParamMap**” has been added and it introduced to represent the parameters of a URL.

**const** uid = **this**.route.snapshot.paramMap.get('UId');

**this**.userService.get(uid).subscribe(user => **this**.name = name);

**15. CanDeactivate** - This “**CanDeactivate**” interface now has an extra (optional) parameter and it is containing the next state.

**16. I18n**- The internationalization is tiny improvement.

//Angular 4-

<div [ngPlural]="value">

<ng-template ngPluralCase="0">there is nothing</ng-template>

<ng-template ngPluralCase="1">there is one</ng-template>

</div>

//Angular 2-

<div [ngPlural]="value">

<ng-template ngPluralCase="=0">there is nothing</ng-template>

<ng-template ngPluralCase="=1">there is one</ng-template>

</div>

**What's New In Angular 5? [Angular 4 vs. Angular 5]**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [4:19 AM](https://www.code-sample.com/2017/08/whats-new-in-angular-5-angular-4-vs.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=887709382458219460&from=pencil)

Angular 5 is going to be a much better Angular and you will be able to take advantage of it much easier and Version 5 will be fully released on September/October 2017.

Stayed Informed – [Angular 4](https://www.code-sample.com/2017/04/angular-4-interview-questions-and.html) and [Angular 2](https://www.code-sample.com/2016/06/angular-2-interview-questions-and.html)

The **Angular 5 Contains** bunch of new features, performance improvements and lot of bug fixes and also some surprises to Angular lovers.

1.     Make AOT the default

2.     Watch mode

3.     Type checking in templates

4.     More flexible metadata

5.     Remove \*.ngfactory.ts files

6.     Better error messages

7.     Smooth upgrades

8.     Tree-Shakeable components

9.     Hybrid Upgrade Application

10. And so on...

**Angular 5 Performance Improvements - Angular 5**

1.     Use of addEventListener for the faster rendering and it is the core functionality.

2.     Update to new version of build-optimizer.

3.     Added some Improvements on the abstract class methods and interfaces

4.     Remove decorator DSL which depends on Reflect for Improve the Performance of Apps and This is the core functionality.

5.     Added an option to remove blank text nodes from compiled templates

6.     Switch Angular to use Static-Injector instead of Reflective-Injector.

7.     Improve the applications testing.

8.     Improve the performance of hybrid applications

9.     Improvements on Lazy loading for Angular

10. And so on...

**Some Improvement on HttpClient** – This is used for Applications communicate with backend services over the HTTP protocol!

1.     Improvement on Type-checking the response

2.     Improvement on Reading the full response

3.     Improvement on Error handling and fetching error details

4.     Improvement on Intercepting all requests or responses

5.     Improvement on Logging

6.     Improvement on Caching

7.     Improvement on XSRF Protection

**Added Features - Angular 5**

1.     Added Representation of Placeholders to xliff and xmb in the compiler

2.     Added an Options Arg to Abstract Controls in the forms controls

3.     Added add default updateOn values for groups and arrays to form controls

4.     Added updateOn blur option to form controls

5.     Added updateOn submit option to form controls

6.     Added an Events Tracking Activation of Individual Routes

7.     Added NgTemplateOutlet API as stable in the common controls

8.     Create StaticInjector which does not depend on Reflect polyfill

9.     Added [@.disabled] attribute to disable animation children in the animations

10. And so on..

**Router Life Cycle Events – Angular 5**

Added new router life cycle events for Guards and Resolvers -

1.     GuardsCheckStart,

2.     GuardsCheckEnd,

3.     ResolveStart and

4.     ResolveEnd

**Angular 5 Bug Fixes - Angular 5**

1.     Fixed compilation error by using the correct type for providers

2.     Skip PWA test when redeploying non-public commit

3.     Don't strip CSS source maps. This is the compiler related fix

4.     Remove tsickle (language-service) dependency

5.     Support persisting dynamic styles within animation states

6.     Ignore @import in multi-line css

7.     Fix platform-browser-dynamic

8.     Forbid destroyed views to be inserted or moved in VC

9.     Support persisting dynamic styles within animation states

10. And so on...

**What's New In Angular 6? What Are Improvements In Angular 6?**

[Anil Singh](https://www.blogger.com/profile/09359926778482233933) [10:49 PM](https://www.code-sample.com/2018/01/whats-new-in-angular-6.html) [**Edit**](https://www.blogger.com/post-edit.g?blogID=9195201046271844871&postID=6936790131628922773&from=pencil)

What's New In Angular 6? And What Are the Improvements In Angular 6?

**Off-course! Angular 6 being smaller, faster and easier to use and it will making developers life easier**. Recently **Angular 6.0.0-beta.7** is **released**and production release on end of **March 2018**.

The [**Angular Team**](https://www.code-sample.com/2018/01/whats-new-in-angular-6.html) are working on lots of bug fixes, new features and added/update/remove/ re-introduce/ and may more things.

Let’s see the about Angular 6 in details -

**1)** Typescript 2.6.x supports

**2)** Added Angular Material and CDK Stable

**3)** Component Dev Kit (CDK) - CDK allows you to build your own library of UI components using Angular Material.

**4)** Improved decorator error messages

**5)** Fix platform-detection example for Universal

**6)** Ivy Renderer - It is a new backward compatible and main focused area - speed improvements, size reduction, and increased flexibility.

**7)** Add afterContentInit and afterContentChecked to render

**8)** Added to supports of nativeElement

**9)** Added Optional generic type for ElementRef

The Example looks like -

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

**10)** Bazel Compiler - Bazel only rebuilds what is necessary.

**11)** Added Test Comment

**12)** Add missing lifecycle tests for projected components

**13)** Closure Compiler - Closure Compiler consistently generates smaller bundles.

**14)** Rename QueryPredicate to LQuery and LQuery to LQueries

**15)** Service Worker - Service worker is a script that runs in the web browser. It also manages caching for an application.

**16)** Added multiple validators for array method of FormBuilder

The Example looks like -

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])

  });

}

**17)** Handle string with and without line boundary - Now Handle string with and without line boundary (^ & $) on pattern validators. Previously, it works with string not boundaries.

**18)** AbstractControl statusChanges - Previous version, not emits an event when you called “markAsPending” but now emits an event of "PENDING" when we call AbstractControl markAsPending.

**19)** Updates on NgModelChange - Now emitted after value and validity is updated on its control. Previously, it was emitted before updated.

Previously it looks like -

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

And

onChange(value) {

  console.log(value);   // would log the updated value, not old value

}

Now, its looks like -

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

And

onChange(NgModel: NgModel) {

  console.log(NgModel.value);// would log old value, not updated value

}

**20)** Allow HttpInterceptors to inject HttpClient –

Previously, an interceptor attempting to inject HttpClient directly would receive a circular dependency error, as HttpClient was constructed via a factory which injected the interceptor instances. Users want to inject HttpClient into interceptors to make supporting.

Either HttpClient or the user has to deal specially with the circular Dependency. This change moves that responsibility into HttpClient itself. By utilizing a new class HttpInterceptingHandler which lazily Loads the set of interceptors at request time, it's possible to inject HttpClient directly into interceptors as construction of HttpClient no longer requires the interceptor chain to be constructed.

**21)** Add navigationSource and restoredState to NavigationStart – Currently, NavigationStart there is no way to know if navigation was triggered imperatively or via the location change. These two use cases should be handled differently for a variety of use cases (e.g., scroll position restoration). This PR adds a navigation source field and restored navigation id (passed to navigations triggered by a URL change).

**22)** Add type and hooks to directive def

**23)** Enable size tracking of a minimal CLI render3 application

**24)** Add canonical view query

**25)** Language Service – The 2.6 version of Typescript’s “resolveModuleName” started to require paths passed to be separated by '/' instead of being able to handle '\'.

Ref: <https://www.code-sample.com/2016/06/angular-2-interview-questions-and.html>

JS Ref: <https://www.code-sample.com/2017/06/typescript-interview-questions-and.html>

TS: <https://www.code-sample.com/2017/06/typescript-interview-questions-and.html>

<https://www.onlineinterviewquestions.com/typescript-interview-questions/#.WtiIHBWn_cv>