

# ANGULAR4

## Contents

ANGULAR 4 BASICS.....	3
Installing And Setting Up Angular4 Application .....	3
DEFINITION.....	3
ENVIRONMENT SETUP .....	3
WORKING WITH ANGULARCLI.....	5
ANGULARCLI DEFINITION.....	5
INSTALL ANGULAR CLI.....	6
Cli Commands.....	7
Structure of angular application.....	8
STRUCTURE .....	9
Creating and Working With Component .....	17
Create new component .....	17
Type Script Basics .....	22
KeyWords .....	23
White Space and Line Breaks .....	23
TypeScript is Case Sensitive .....	23
Variables.....	23
Variable Declaration in Typescript.....	24
How To Work With Templates .....	25
Templates.....	25
Templates- NGIF CONDITION.....	25
Templates-NGIFELSECONDITION .....	27

# ANGULAR 4 BASICS

## INSTALLING AND SETTING UP ANGULAR4 APPLICATION

### DEFINITION

Angular 4 is a JavaScript framework for building web applications and apps in JavaScript, html, and Type Script, which is a superset of JavaScript. Angular provides built-in features for animation, http service, and materials which in turn have features such as auto-complete, navigation, toolbar, menus, etc. The code is written in Type Script, which compiles to JavaScript and displays the same in the browser.

### ENVIRONMENT SETUP

To install Angular 4, we require the following –

- Nodejs
- Npm
- Angular CLI
- IDE for writing your code

Nodejs has to be greater than 4 and npm has to be greater than 3.

### Nodejs Definition

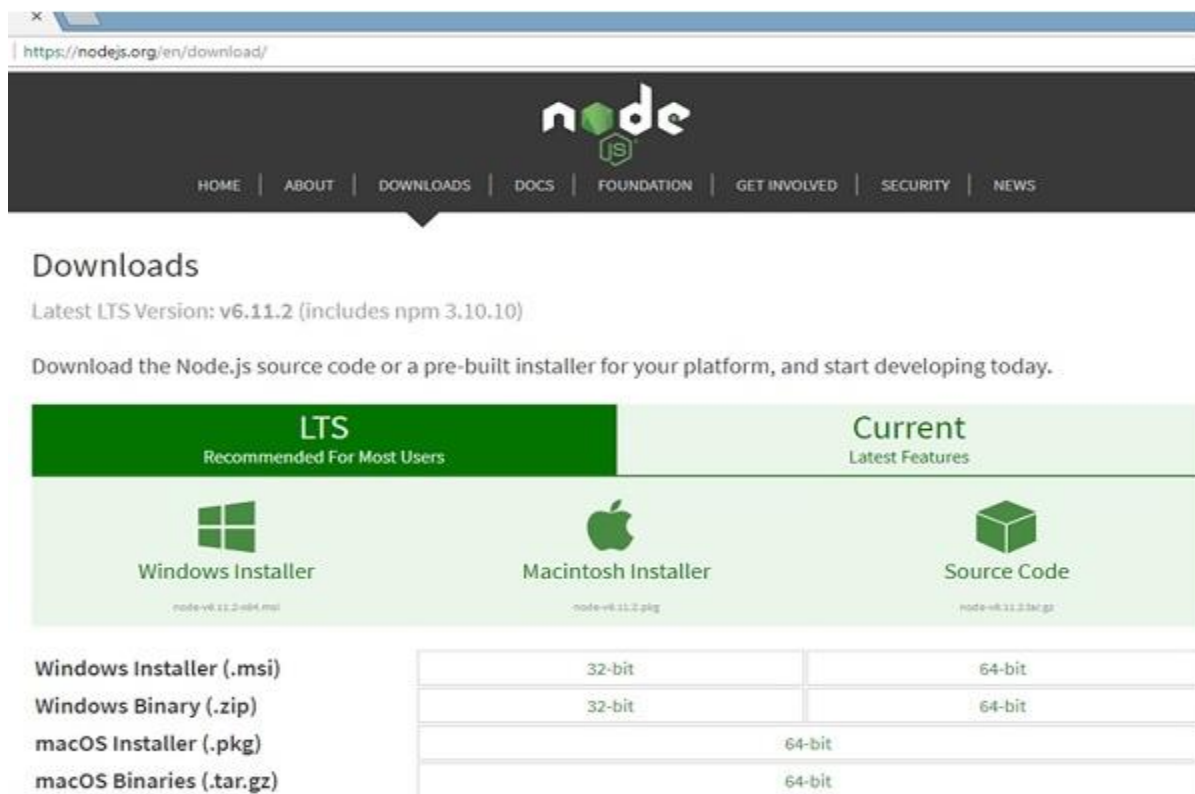
Node.js is a server-side platform built on Google Chrome's JavaScript Engine (V8 Engine). Node.js was developed by Ryan Dahl in 2009. Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

## Why used nodejs in angular4

Angular does not need node.js directly. Node js is used for all the build and development tools. Angular is a framework and you may use typescript or javascript or dart programming language to program using Angular.

## Nodejs install

To check if nodejs is installed on your system, type **node -v** in the terminal. This will help you see the version of nodejs currently installed on your system. If it does not print anything, install nodejs on your system. To install nodejs, go the homepage <https://nodejs.org/en/download/> of nodejs and install the package based on your OS






node

HOME | ABOUT | DOWNLOADS | DOCS | FOUNDATION | GET INVOLVED | SECURITY | NEWS

### Downloads

Latest LTS Version: v6.11.2 (includes npm 3.10.10)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

LTS Recommended For Most Users	Current Latest Features
 <b>Windows Installer</b> <small>node-v6.11.2-x86.msi</small>	 <b>Macintosh Installer</b> <small>node-v6.11.2.pkg</small>
 <b>Source Code</b> <small>node-v6.11.2.tar.gz</small>	

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit	
macOS Binaries (.tar.gz)	64-bit	

Based on your OS, install the required package. Once nodejs is installed, npm will also get installed along with it.

## Npm

The Angular CLI, Angular applications, and Angular itself depend upon features and functionality provided by libraries that are available as npm packages. To check if npm is installed or not, type `npm -v` in the terminal. It should display the version of the npm. You can download and install these npm packages with the npm client, which runs as a Node.js application. Node.js and npm are essential to Angular development.

### CHECK VERSIONS

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

D:\swethab\myang>node -v
v8.12.0

D:\swethab\myang>npm -v
6.4.1

D:\swethab\myang>
```

Here `node -v` command checks our pc for nodejs is available or not and check the nodejs version. After giving this command nothing is displayed our command prompt and our pc does not have nodejs. else nodejs version is displayed and our system having nodejs. and nodejs has to be greater than 4 then npm has to be greater than 3.

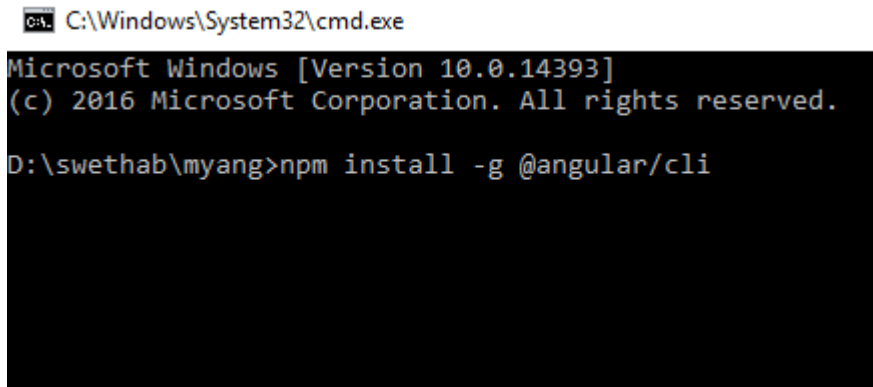
## WORKING WITH ANGULARCLI

## ANGULARCLI DEFINITION

The Angular CLI is a command-line interface tool that you use to initialize, develop and maintain Angular applications.

## INSTALL ANGULAR CLI

Enter the above command to install Angular 4. The installation process will start and will take a few minutes to complete.



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

D:\swethab\myang>npm install -g @angular/cli
```

We have also used -g to install Angular CLI globally. Now, you can create your Angular 4 project in any directory or folder and you don't have to install Angular CLI project wise, as it is installed on your system globally and you can make use of it from any directory.

## Cli Commands

```
ng add
ng build
ng config
ng doc
ng e2e
ng generate
ng help
ng lint
ng new
ng run
ng serve
ng test
ng update
ng version
ng xi18n
```

### Ng build

Compiles an Angular app into an output directory named dist/ at the given output path. Must be executed from within a workspace directory

### Ng new

Creates a new workspace and an initial Angular app

### Ng serve

Builds and serves your app, rebuilding on file changes

## Ng version

Outputs Angular CLI version

### CHECK CLI VERSION

ng v – this command used to check the cli version

```
C:\Windows\System32\cmd.exe
(c) 2016 Microsoft Corporation. All rights reserved.
D:\swethab\myang>ng v

Angular CLI

Angular CLI: 7.0.6
Node: 8.12.0
OS: win32 x64
Angular:
...
Package                                  Version
-----
@angular-devkit/architect                0.10.6
@angular-devkit/core                     7.0.6
@angular-devkit/schematics               7.0.6
@schematics/angular                     7.0.6
@schematics/update                       0.10.6
rxjs                                     6.3.3
typescript                              3.1.6
```

## IDE

You will get the above installation in your terminal, once Angular CLI is installed. You can use any IDE of your choice, i.e., WebStorm, Atom, Visual Studio Code, etc.



# Structure of angular application

## CREATE A NEWPROJECT

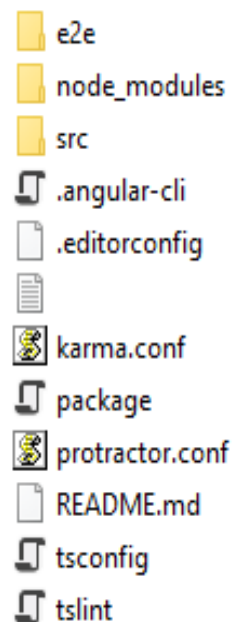
```
C:\Windows\System32\cmd.exe

Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

D:\swethab\myang>ng new myangularApp
```

Here ng is a angular and this command used to create a new project.The installation process will start and will take a few minutes to complete.

## STRUCTURE



**e2e –**

end to end test folder. Mainly e2e is used for integration testing and help ensure application works fine.

**node\_modules –**

The npm package installed is node\_modules. You can open the folder and see the packages available.

**src –**

This folder is where we will work on the project using Angular 4.

**.angular-cli.json –**

It basically holds the project name, version of cli, etc.

**.editorconfig –**

This is the config file for the editor.

**.gitignore –**

A .gitignore file should be committed into the repository, in order to share the ignore rules with any other users that clone the repository.

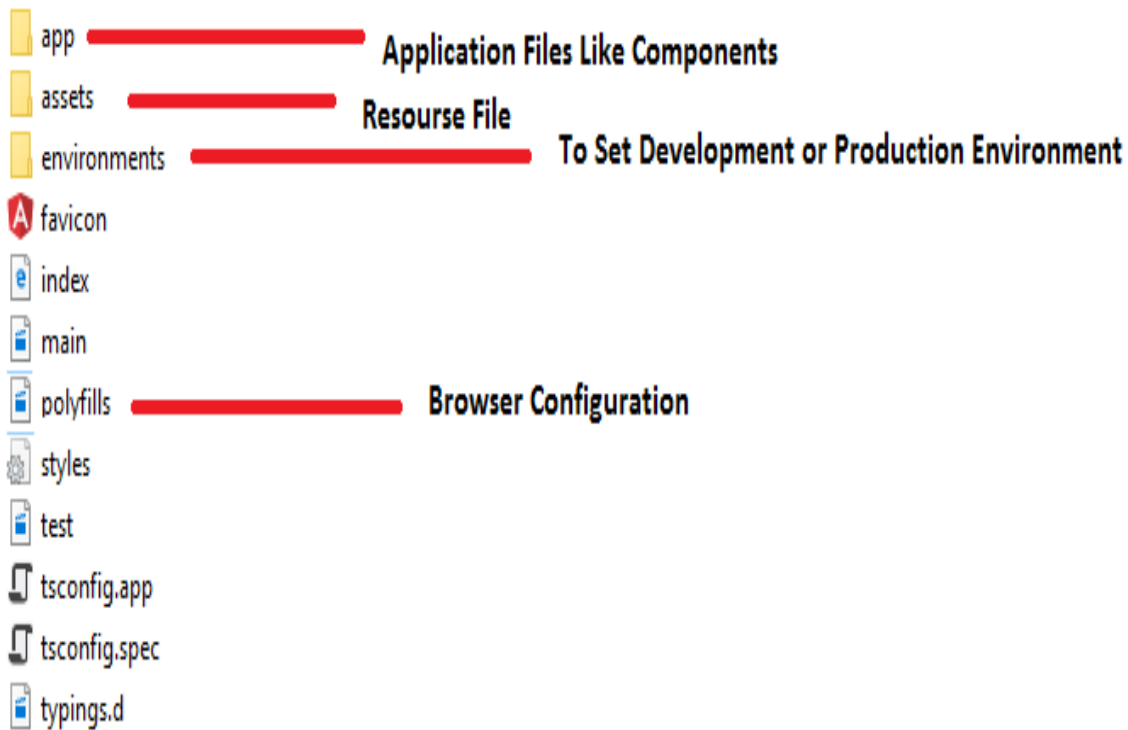
**karma.conf.js –**

This is used for unit testing via the protractor. All the information required for the project is provided in karma.conf.js file.

**package.json –**

The package.json file tells which libraries will be installed into node\_modules when you run npm install.

## APPLICATION STRUCTURE



## INDEX.HTML

This is the file which is displayed in the browser.

```
1 <!doctype html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>MyangularApp</title>
6   <base href="/">
7
8   <meta name="viewport" content="width=device-width, initial-scale=1">
9   <link rel="icon" type="image/x-icon" href="favicon.ico">
10 </head>
11 <body>
12   <app-root></app-root>
13 </body>
14 </html>
15
```

The body has `<app-root></app-root>`. This is the selector which is used in **app.component.ts** file and will display the details from app.component.html file.

## MAIN.TS

main.ts is the file from where we start our project development. It starts with importing the basic module which we need. Right now if you see angular/core, angular/platform-browser-dynamic, app.module and environment is imported by default during angular-cli installation and project setup.

Index.html internally refers to main.ts

```
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();
}

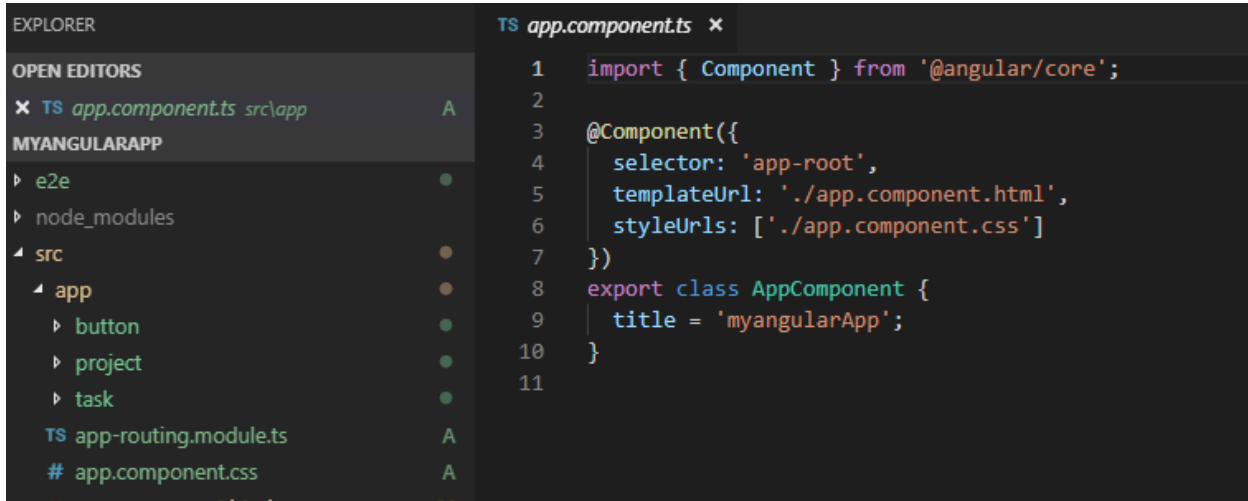
platformBrowserDynamic().bootstrapModule(AppModule)
  .catch(err => console.error(err));
```

## COMPONENTS

Components are basically classes that interact with the .html file of the component, which gets displayed on the browser. it consists of the following files

- app.component.css
- app.component.html
- app.component.spec.ts
- app.component.ts

## STRUCTURE OF COMPONENT



```
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-root',
5   templateUrl: './app.component.html',
6   styleUrls: ['./app.component.css']
7 })
8 export class AppComponent {
9   title = 'myangularApp';
10 }
11
```

Here selector tag is which is placed in the app.component.html file because index.html internally refer app.component.html. and templateUrl have the html file and styleUrls refer the css file.

## MODULE

**Module** in Angular refers to a place where you can group the components, directives, pipes, and services, which are related to the application. To define module, we can use the **NgModule**. When you create a new project using the Angular –cli command, the ngmodule is created in the app.module.ts file by default

## STRUCTURE OF MODULE

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

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';

@NgModule({
  declarations: [
    AppComponent,
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

### BROWSER MODULE

It starts with `BrowserModule` exports `CommonModule` and provides a few services specific to the browser platform

### NGMODULE

`@NgModule` and contains an object which has declarations, imports, providers and bootstrap.

### ROUTINGMODULE

*Routes* tell the router which view to display when a user clicks a link or pastes a URL into the browser address bar. A typical Angular Route has two properties:

1. `path`: a string that matches the URL in the browser address bar.

2. component: the component that the router should create when navigating to this route.

## Declaration

It is an array of components created. If any new component gets created, it will be imported first and the reference will be included in declarations as shown below –

```
declarations: [  
  AppComponent,  
  NewCmpComponent  
]
```

## Import

It is an array of modules required to be used in the application. It can also be used by the components in the Declaration array. For example, right now in the @NgModule we see the Browser Module imported. In case your application needs forms, you can include the module as follows –

```
import { FormsModule } from '@angular/forms';
```

The import in the **@NgModule** will be like the following –

```
imports: [  
  BrowserModule,  
  FormsModule  
]
```

## Providers

This will include the services created. services is Components shouldn't fetch or save data directly and they certainly shouldn't knowingly present fake data. They should focus on presenting data and delegate data access to a service.

## Bootstrap

This includes the main app component for starting the execution.

## APP.COMPONENT.HTML

```
app.component.html x TS app.component.ts material.component.html
1 <!--The content below is only a placeholder and can be replaced.-->
2 <div style="text-align:center">
3   <h1>
4     Welcome to {{ title }}!
5   </h1>
6   Tour of Heroes</a></h2>
12  </li>
13  <li>
14    <h2><a target="_blank" rel="noopener" href="https://github.com/angular/angular-cli/wiki">CLI Documentati
15  </li>
16  <li>
17    <h2><a target="_blank" rel="noopener" href="https://blog.angular.io/">Angular blog</a></h2>
18  </li>
19 </ul>
20 <router-outlet></router-outlet>
21
```

## RUN THE PROGRAM

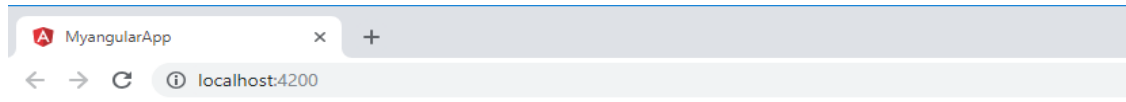
```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

D:\swethab\myang\myangularApp>ng serve --open
```

Ng serve --open command used to Builds and serves your app Opens the url in default browser.



## OUTPUT



**Welcome to myangularApp!**




Here are some links to help you start:

- [Tour of Heroes](#)
- [CLI Documentation](#)
- [Angular blog](#)

## Creating and Working With Component

### Create new component

To create new component given the command in the terminal

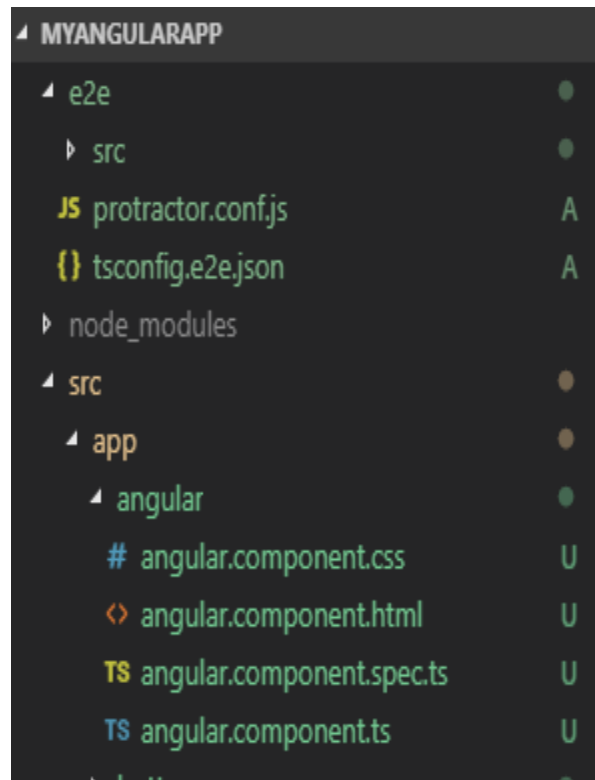
 C:\Windows\System32\cmd.exe

```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

D:\swethab\myang\myangularApp>ng g component angular
```

Create new component at globally and above component name angular

## AFTER CREATING COMPONENT



The following files are created in the angular component folder –

- angular.component.css – css file for the new component is created.
- angular.component.html – html file is created.
- angular.component.spec.ts – this can be used for unit testing.
- angular.component.ts – here, we can define the module, properties, etc.

## APP.MODULE.TS

```

TS app.module.ts • <> app.component.html <> material.component.html
1 import { BrowserModule } from '@angular/platform-browser';
2 import { NgModule } from '@angular/core';
3 import { AppRoutingModule } from './app-routing.module';
4
5 import { AppComponent } from './app.component';
6
7 import { AngularComponent } from './angular/angular.component';
8
9
10 @NgModule({
11   declarations: [
12     AppComponent,
13
14     AngularComponent,
15   ],
16   imports: [
17     BrowserModule,
18     AppRoutingModule,
19
20
21
22   ],
23   providers: [],
24   bootstrap: [AppComponent]
25 })
26 export class AppModule { }
27
28

```

In this above figure imported the AngularComponent from angular folder and declare angularcomponent @NgModule declarations part.

## APP.COMPONENT.HTML

```

11 <h2><a target="_blank" rel="noopener" href="https://angular.io/...
12 </li>
13 <li>
14   <h2><a target="_blank" rel="noopener" href="https://github.com/a
15 </li>
16 <li>
17   <h2><a target="_blank" rel="noopener" href="https://blog.angular
18 </li>
19 </ul>-->
20 <app-angular></app-angular>

```

<app-angular> is a selector which is used in angular.component.ts file.

Will get details and displayed in the browser

### angular.component.ts

```
<img alt="Screenshot of angular.component.ts code" data-bbox="142 200 853 587"/>A screenshot of a code editor showing the angular.component.ts file. The editor has a dark background with light blue and orange text. The code is as follows:  
1  import { Component, OnInit } from '@angular/core';  
2  
3  @Component({  
4    selector: 'app-angular',  
5    templateUrl: './angular.component.html',  
6    styleUrls: ['./angular.component.css']  
7  })  
8  export class AngularComponent implements OnInit {  
9  
10   constructor() { }  
11  
12   ngOnInit() {  
13   }  
14  
15 }  
16
```

Here above this figure app-angular is a selector name. and in this selector name given to the app.component.html.

## angular.component.html

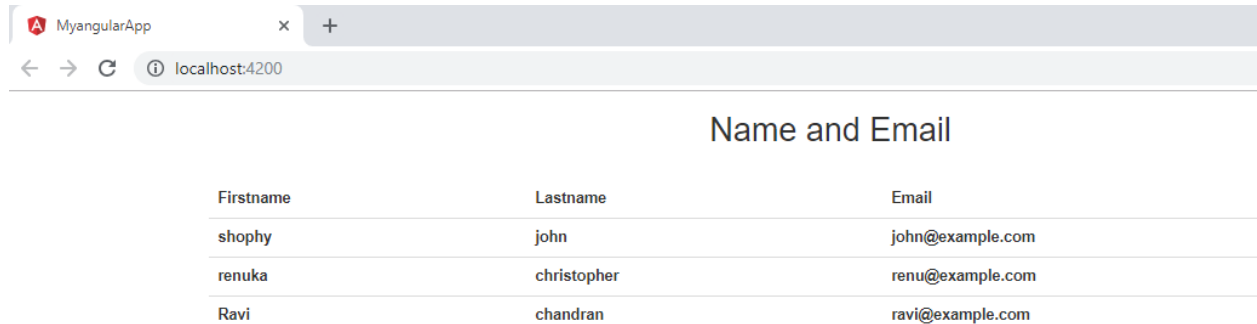
```
<div class="container">  
  <h2 style=text-align:center>Name and Email</h2><br>  
  <table class="table table-hover">  
    <thead>  
      <tr>  
        <th>Firstname</th>  
        <th>Lastname</th>  
        <th>Email</th>  
      </tr>  
    </thead>  
    <tbody>  
      <tr>  
        <td>shophy</td>  
        <td>john</td>  
        <td>john@example.com</td>  
      </tr>  
      <tr>  
        <td>renuka</td>  
        <td>christopher</td>  
        <td>renu@example.com</td>  
      </tr>  
      <tr>  
        <td>Ravi</td>  
        <td>chandran</td>  
        <td>ravi@example.com</td>  
      </tr>  
    </tbody>  
  </table>  
</div>
```

## RUN THE APPLICATION

```
C:\Windows\System32\cmd.exe  
Microsoft Windows [Version 10.0.14393]  
(c) 2016 Microsoft Corporation. All rights reserved.  
  
D:\swethab\myang\myangularApp>ng serve --open
```

Once you run `http://localhost:4200/` in the browser, you will be directed to the following screen

## OUTPUT



Name and Email		
Firstname	Lastname	Email
shopfy	john	john@example.com
renuka	christopher	renu@example.com
Ravi	chandran	ravi@example.com

## Type Script Basics

The popular JavaScript framework **Angular 2.0** is written in TypeScript. JavaScript was introduced as a language for the client side.

## KeyWords

break	as	any	switch
case	if	throw	else
var	number	string	get
module	type	instanceof	typeof
public	private	enum	export
finally	for	while	void
null	super	this	new
in	return	true	false
any	extends	static	let
package	implements	interface	function
new	try	yield	const
continue	do	catch	

## White Space and Line Breaks

TypeScript ignores spaces, tabs, and newlines that appear in programs. You can use spaces, tabs, and newlines freely in your program and you are free to format and indent your programs in a neat and consistent way that makes the code easy to read and understand

## TypeScript is Case Sensitive

TypeScript is case-sensitive. This means that TypeScript differentiates between uppercase and lowercase characters.

## Variables

Variable, by definition, is “a named space in the memory” that stores values. In other words, it acts as a container for values in a program.

## TypeScript variables must follow the JavaScript naming rules –

- Variable names can contain alphabets and numeric digits.
- They cannot contain spaces and special characters, except the underscore (\_) and the dollar (\$) sign.
- Variable names cannot begin with a digit.

A variable must be declared before it is used. Use the var keyword to declare variables.

### Variable Declaration in Typescript

The type syntax for declaring a variable in TypeScript is to include a colon (:) after the variable name, followed by its type.

**var name:string = "swetha"**

The variable stores a value of type string

**var name:string;**

The variable is a string variable. The variable's value is set to undefined by default

**var name = "swetha"**

The variable's type is inferred from the data type of the value. Here, the variable is of the type string

**var name;**

The variable's data type is any. Its value is set to undefined by default.



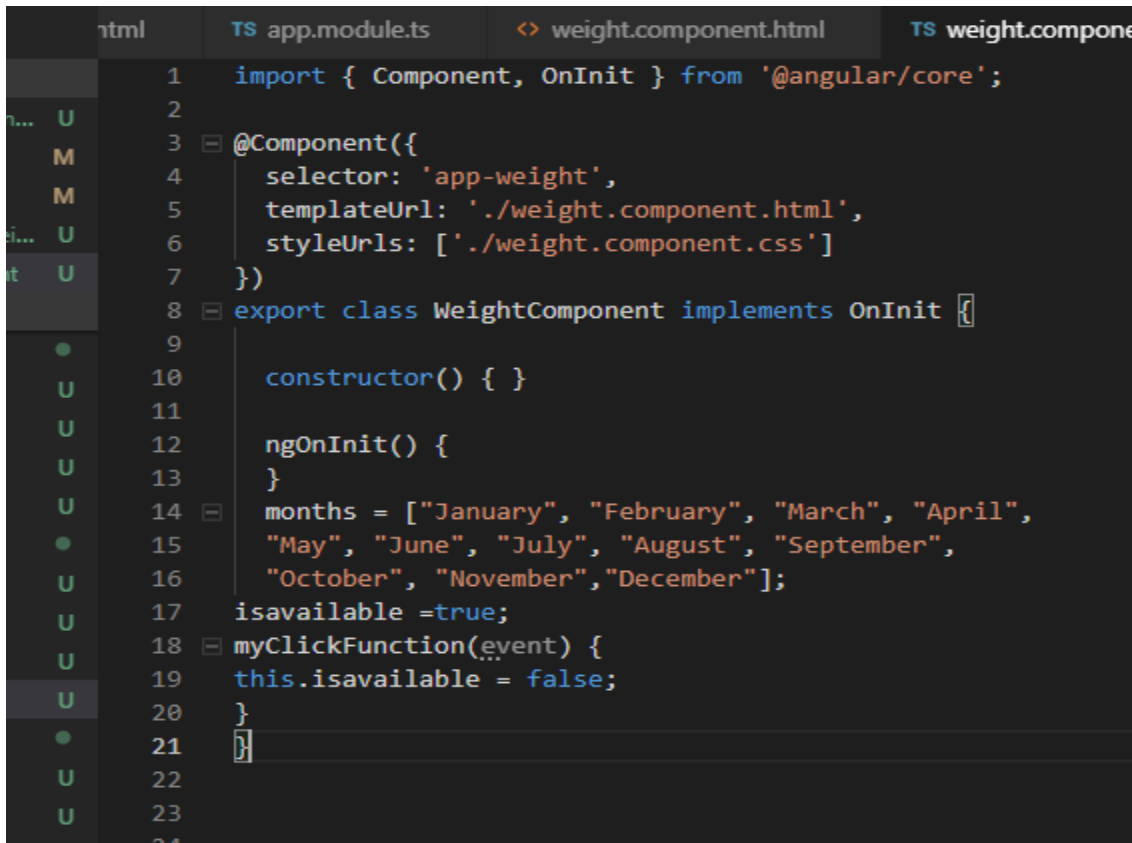
## How To Work With Templates

### Templates

The template is a fully responsive with a neat and clean design that looks beautiful on any device. Angular 4 uses the `<ng-template>` as the tag instead of `<template>` which is used in Angular2.

The reason Angular 4 changed `<template>` to `<ng-template>` is because there is a name conflict between the `<template>` tag and the html `<template>` standard tag. `<ng-template>` is an angular element for rendering HTML. It is never displayed directly. It can be displayed using structural directive,

### templates ngif condition



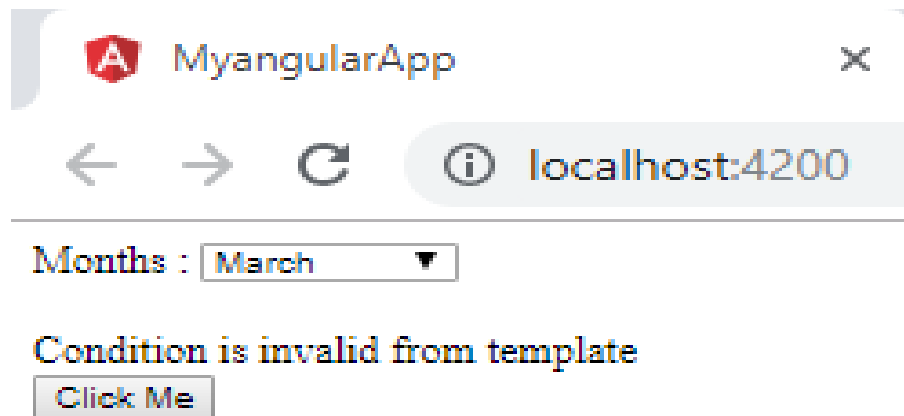
```
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-weight',
5   templateUrl: './weight.component.html',
6   styleUrls: ['./weight.component.css']
7 })
8 export class WeightComponent implements OnInit {
9
10  constructor() { }
11
12  ngOnInit() {
13
14    months = ["January", "February", "March", "April",
15             "May", "June", "July", "August", "September",
16             "October", "November", "December"];
17    isavailable = true;
18    myClickFunction(event) {
19      this.isavailable = false;
20    }
21  }
22
23
24
```

- 1) Array initialization-initializing months January to December
- 2) Isavailable is true and condition Is displayed in the browser

## Customer.component.html

```
<> customer.component.html x
1  <div> Months :
2    <select (change) = "changemonths($event)" name = "month">
3      <option *ngFor = "let i of months">{{i}}</option>
4    </select>
5  </div>
6  <br/>
7
8  <div>
9    <span *ngIf = "isavailable;else condition"></span>
10   <ng-template #condition>Condition is invalid from template</ng-template>
11 </div>
12 <button (click) = "myClickFunction($event)">Click Me</button>
13
14
```

## Ngif output



The variable **isavailable** is true so nothing is printed. If you click the button, the respective template will be called. It means after clicking the button condition is invalid was displayed in the browser.

## templates-ngifelsecondition

### Customer.component.ts

```
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-weight',
5   templateUrl: './weight.component.html',
6   styleUrls: ['./weight.component.css']
7 })
8 export class WeightComponent implements OnInit {
9
10   constructor() { }
11
12   ngOnInit() {
13   }
14   months = ["January", "February", "March", "April",
15     "May", "June", "July", "August", "September",
16     "October", "November", "December"];
17   isavailable = false;
18   myClickFunction(event) {
19     this.isavailable = true;
20   }
21 }
22
```

Isavailable is true then condition 1 is displayed

Isavailable is false then condition 2 is displayed

### Customer.component.html

```
<? customer.component.html x
1 <div> Months :
2   <select (change) = "changemonths($event)" name = "month">
3     <option *ngFor = "let i of months">{{i}}</option>
4   </select>
5 </div>
6 <br/>
7
8 <div>
9   <span *ngIf = "isavailable;then condition1 else condition2">Condition is valid.</span>
10   <ng-template #condition1>Condition is valid from template</ng-template>
11   <ng-template #condition2>Condition is invalid from template</ng-template>
12 </div>
13 <button (click) = "myClickFunction($event)">Click Me</button>
14
```

Span tag, we have added the if statement with the else condition and will call template condition1, else condition2.

## Ng if else output

---

Months :

Condition is valid from template

[Click Me](#)

The variable **isavailable** is false so the condition2 template is printed. If you click the button, the respective template will be called. Means after clicking the button condition1 is displayed

## Routing Concepts

Routing basically means navigating between pages. You have seen many sites with links that direct you to a new page. This can be achieved using routing.

### App.Module.ts

```
TS app.module.ts ●
1  import { BrowserModule } from '@angular/platform-browser';
2  import { NgModule } from '@angular/core';
3  import { AppRoutingModule } from './app-routing.module';
4  import { RouterModule } from '@angular/router';
5  import { AppComponent } from './app.component';
6  import { NewCmpComponent } from './new-cmp/new-cmp.component';
7  @NgModule({
8    declarations: [
9      AppComponent,
10
11      NewCmpComponent,
12    ],
13    imports: [
14      BrowserModule,
15      AppRoutingModule,
16      RouterModule.forRoot([
17        {
18          path: 'new-cmp',
19          component: NewCmpComponent
20        }
21      ])
22    ],
23    providers: [],
24    bootstrap: [AppComponent]
25  })
26  export class AppModule { }
27
28
29
30
```

above RouterModule is imported from angular/router and RouterModule refers to the forRoot which takes an input as an array, which in turn has the object of the path and the component. Path is the name of the router and component is the name of the class, i.e., the component created.

### NewCmp.Component.ts

```
TS new-cmp.component.ts x
1  import { Component, OnInit } from '@angular/core';
2
3  @Component({
4    selector: 'app-new-cmp',
5    templateUrl: './new-cmp.component.html',
6    styleUrls: ['./new-cmp.component.css']
7  })
8  export class NewCmpComponent implements OnInit {
9
10     newcomponent = "Have Good Day";
11
12     constructor() { }
13
14     ngOnInit() {
15     }
16
17 }
18
```

## NewCmp.Component.Html

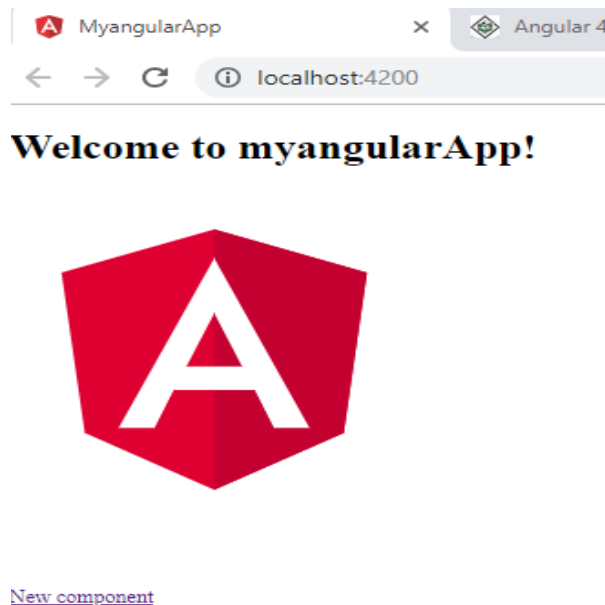
```
<> new-cmp.component.html x
1
2  <p>
3    |   {{newcomponent}}
4  </p>
5
6  <p>
7    |   new-cmp works!
8  </p>
9  |
```

## App.Component.Html

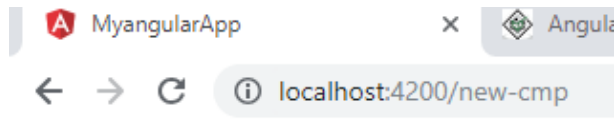
```
app.component.html x
1
2 <div style="text-align:left">
3 <h1>
4 Welcome to {{ title }}!
5 </h1>
6 New component</a>
12 <br />
13 <br />
14 <router-outlet></router-outlet>
15
```

Above Anchor tag and given routerLink as "new-cmp" This is referred in app.module.ts as the path. When a user clicks new component, the page should display the content

## Output



When a user clicks New component, you will see the following in the browser.



**Welcome to myangularApp!**



[New component](#)

Have Good Day

new-cmp works!

For above Output, When a user clicks New component, the page is not refreshed and the contents are shown to the user without any reloading. Only a particular piece of the site code will be reloaded when clicked. This feature helps when we have heavy content on the page and needs to be loaded based on the user interaction. The feature also gives a good user experience as the page is not reloaded.