**Angular 2**

**Overview:**

The whole concept of application structure has been changed in angular2. Previously, it was the MVC framework that allowed us to create applications in the pattern of rather tightly coupled entities like controller, views and services.

Now, the concept of directives has been moved further closer to the Web components standard . The whole application is now a component which contains another set of components which ends in tree-like structure. The purpose of angular 2 application is to create components that doesn’t depend on each other and decoupled with DOM elements.

Angular 2 uses Typescript compiler which converts typescript file to javascript file when compiled. The *TypeScript* is a super set of JavaScript which is migrated to TypeScript and code written in TypeScript makes less prone to run time errors.

To work with Angular 2, Node js installation is necessary and npm packages.

**Steps to set Environment:**

* Download quickstart example from <https://angular.io/docs/ts/latest/guide/setup.html>
* Make sure you have node js and npm installed.
* Install npm packages
* Run npm start to launch the sample application downloaded.

****

**Index.html**

<!DOCTYPE html>

<html>

<head>

<title>Angular Quickstart</title>

<base href="/">

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<style>

body {color:#369;font-family: Arial,Helvetica,sans-serif;}

</style>

<!-- Polyfills -->

<script src="node\_modules/core-js/client/shim.min.js"></script>

<script src="node\_modules/zone.js/dist/zone.js"></script>

<script src="node\_modules/systemjs/dist/system.src.js"></script>

<script src="systemjs.config.js"></script>

<script>

System.import('main.js').catch(function(err){ console.error(err); });

</script>

</head>

<body>

<my-app>Loading AppComponent content here ...</my-app>

</body>

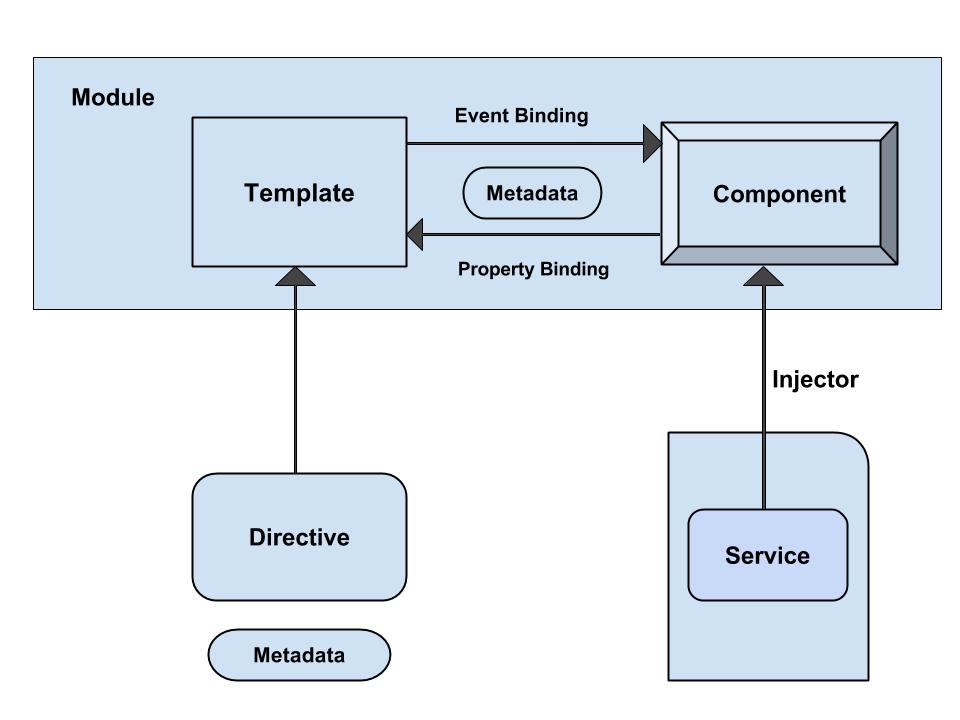
</html>

When the html is loaded, **main.js** javascript file is loaded and Angular name is displayed which is defined in **app.component.ts.**

|  |  |
| --- | --- |
| **File** | **Purpose** |
| app/app.component.ts | Defines the same AppComponent as the one in the QuickStart playground. It is the **root** component of what will become a tree of nested components as the application evolves. |
| app/app.module.ts | Defines AppModule, the root module that tells Angular how to assemble the application. Right now it declares only the AppComponent. Soon there will be more components to declare. |
| main.ts | Compiles the application with the JIT compiler and bootstraps the application's main module (AppModule) to run in the browser. The JIT compiler is a reasonable choice during the development of most projects. |

**The Basic Building blocks:**

* Modules
* Components
* Templates
* Metadata
* Data Binding
* Directives
* Services
* Dependency Injection



**Modules:**

Every angular app has atleast one angular module called root-module. This root-module is a class with @NgModule decorator. Decorators are functions that modify JavaScript classes. Angular has many decorators that attach metadata to classes so that it knows what those classes mean and how they should work.

NgModule is a single metadata object whose properties constitute the module. The properties include:

* Declarations

These are the view classes which belong to the module.

* Exports

The declarations which should be visible and reusable in other components modules.

* Imports

other modules whose exported classes are needed by component templates declared in this module.

* Providers

creators of [services](https://angular.io/docs/ts/latest/guide/architecture.html#services) that this module contributes to the global collection of services; they become accessible in all parts of the app.

* Bootstrap

Only the root module should set this bootstrap property which hosts all other apps.

**Sample Code of a Module:**

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

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

@NgModule({

imports: [ BrowserModule ],

providers: [ Logger ],

declarations: [ AppComponent ],

exports: [ AppComponent ],

bootstrap: [ AppComponent ]

})

export class AppModule { }

**Components:**

Views are controlled by components. It binds the template behavior. You define a component’s application logic like what does it support to view inside a class. The class interacts with the view through an API of properties and methods.

Components split the entire task into different views and contains multiple components inside a particular one. These components can also be reused in any other applications depending on its usage. The component knows how to render itself and configure dependency injection.

To register component, we use *@Component* annotation and can be used to break up the application into smaller parts. There will be only one component per DOM element.

**Template:**

The component’s view can be defined by using the template. For instance:

<div>

Your name is : {{name}}

</div>

**Metadata:**

Metadata is a way of processing the class. Consider we have one component called MyComponent which will be a class until we tell Angular that it's a component.You can use  metadata to the class to tell Angular that MyComponent is a component. The metadata can be attached to TypeScript by using the *decorator*.

For instance:

@Component({

selector : 'mylist',

template : '<h2>Name is Harry</h2>'

directives : [MyComponentDetails]

})

export class ListComponent{...}

The *@Component* is a decorator which uses configuration object to create the component and its view. The *selector* creates an instance of the component where it finds <mylist> tag in parent HTML. The *template* tells Angular how to display the component. The *directive* decorator is used to represent the array of components or directives.

**References**

* [**https://www.tutorialspoint.com/angular2/index.htm**](https://www.tutorialspoint.com/angular2/index.htm)
* [**https://angular.io/docs/ts/latest/guide/setup.html**](https://angular.io/docs/ts/latest/guide/setup.html)
* [**https://docs.npmjs.com/getting-started/installing-node**](https://docs.npmjs.com/getting-started/installing-node)