**What is an Angular Component**

The Component is the main building block of an Angular Application.

The Component contains the data & user interaction logic that defines how the View i.e template (HTML

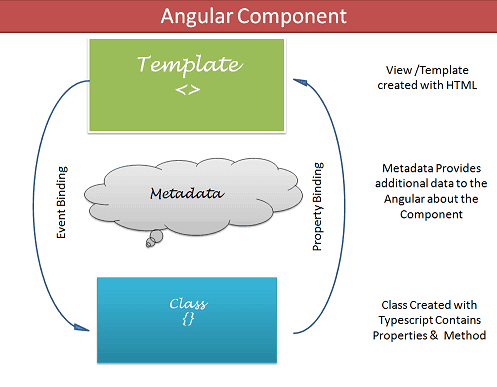
The Angular Components are plain [JavaScript](https://www.tektutorialshub.com/javascript-tutorial/) classes and defined using **@Component Decorator**. It contains Metadata about the Component

The Component is responsible to provide the data to the view by using [data binding](https://www.tektutorialshub.com/angular/angular-data-binding/)  This is done using the special HTML markup knows as the Angular Template Syntax. The Component can also get notified when the View Changes.

The Angular applications will have lots of components. Each component handles a small part of UI. These components work together to produce the complete user interface of the application

The Components consists of three main building blocks

* Template
* Class
* MetaData



### Template (View)

The template defines the layout and content of the View. Without the template,  there is nothing for Angular to render to the DOM.

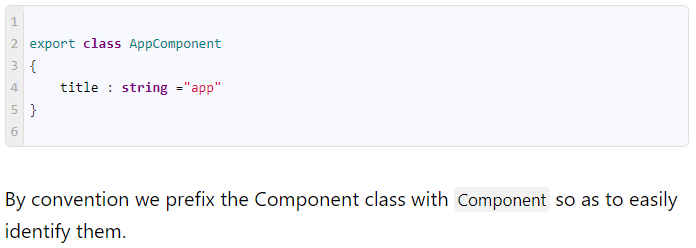
There are two ways you can specify the Template in Angular.

1. Defining the Template Inline
2. Provide an external Template

### Class

The Class provides the data & logic to the View. It contains the JavaScript code associated with Template (View). We use [TypeScript](https://www.tektutorialshub.com/typescript-tutorial/) to create the class, but you can also use JavaScript directly in the class.

Class Contains the Properties & Methods. The Properties of a class can be bind to the view using [Data Binding](https://www.tektutorialshub.com/angular/angular-data-binding/).



### Metadata

Metadata Provides additional information about the component to the Angular. Angular uses this information to process the class. We use the @Component decorator to provide the Metadata to the Component.

#### @Component decorator

A decorator is a function that adds metadata to class, its methods & to its properties. The Components are defined with a @Component class decorator.

When Angular sees a class with @Component decorator, it treats the class as Component.

A Decorator is always prefixed with @. We must place the Decorator immediately before the class definition. We can also build our own decorators.

#### Important Component metadata properties

##### Selector

Selector specifies the simple CSS selector. The Angular looks for the CSS selector in the template and renders the component there.

##### Providers

The Providers are the [Angular Services](https://www.tektutorialshub.com/angular/angular-services/), that our component going to use. The Services provide service to the Components or to the other Services.

##### Directives

The[directives](https://www.tektutorialshub.com/angular/angular-directives/) that this component going to use are listed here.

##### Styles/styleUrls

The CSS styles or style sheets, that this component needs. Here we can use either external stylesheet (using styleUrls) or inline styles (using Styles). The styles used here are specific to the component

##### template/templateUrl

The HTML template that defines our View. It tells Angular how to render the Component’s view. The templates can be inline (using a template) or we can use an external template (using a templateUrl). The Component can have only one template. You can either use inline template or external template and not both

## Creating the Component

The Angular CLI has automatically created the root component **app.component.ts**.( The extension used is **ts** indicating that this is a typescript module file.)

The creation of the Angular component requires you to follow these steps

1. Create the Component file
2. Import the required external Classes/Functions
3. Create the Component class and export it
4. Add @Component decorator
5. Add metadata to @Component decorator
6. Create the Template
7. Create the CSS Styles
8. Register the Component in Angular Module

### Creating the Component File

### 2. Import the Angular Component Library

Before we use any Angular (or external) functions or classes, we need to tell Angular how and where to find it. This is done using the Import statement.

To define the Component class, we need to use the @Component decorator. This function is part of the Angular Core library. So we import it in our class



### 3. Create the Component Class and export it

The third step is to create the Component class using the export keyword. The Export keyword allows other components to use this component importing it. The AppComponent class is shown below

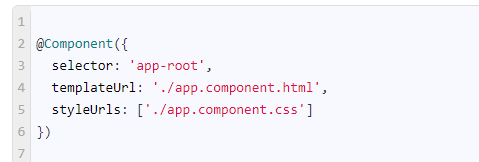
|  |  |
| --- | --- |
|  |  |

### 4. Add @Component decorator

The next step is to inform the Angular that this is a Component class. We do that by adding the @Component decorator.

### 5. Add meta data to @Component decorator

The next step is to add the metadata to the component using the @component decorator. Add the following to the component metadata



#### selector

The angular places the view (template) inside the selector app-root

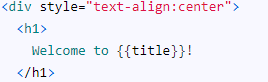
#### templateUrl

In the above example, we have used an external template using templateUrl metadata. The templateUrl points to the external HTML file **app.component.html**.

#### styleUrls

Defines the styles for our template. The metadata points to the external CSS file **app.component.css**. The Component specific CSS styles can be specified here

### 6. Create the



### 7. Add the Styles

The next step is to add the CSS Styles. The styleUrls metadata tells Angular, where to find the CSS File. This property points to external file **app.component.css**

### 8. Register the Angular Component in Angular Module

We have created the Angular Component. The next step is to register it with the [Angular Module](https://www.tektutorialshub.com/angular/angular-modules/)

The Angular Module organizes the components, [directives](https://www.tektutorialshub.com/angular/angular-directives/), [pipes](https://www.tektutorialshub.com/angular/angular-pipes/), and [services](https://www.tektutorialshub.com/angular/angular-services/)that are related and arrange them into cohesive blocks of functionality.

We use [@NgModule](https://www.tektutorialshub.com/angular/angular-modules/) class decorator to define a [Angular Module](https://www.tektutorialshub.com/angular/angular-modules/) and provide metadata about the Modules.



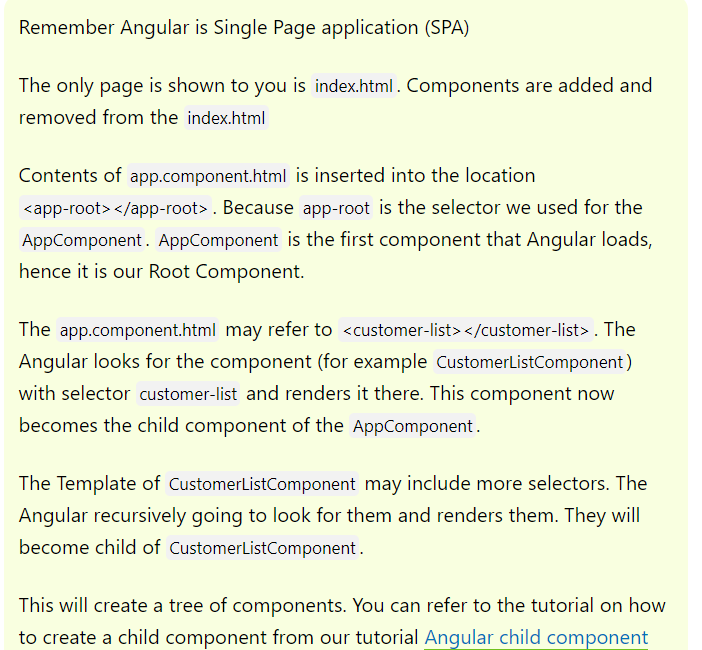
The **declaration arrays** is where we include the components, [pipes](https://www.tektutorialshub.com/angular/angular-pipes/) and [directives](https://www.tektutorialshub.com/angular/angular-directives/) that are part of this module.

We add all the other [Angular Modules](https://www.tektutorialshub.com/angular/angular-modules/) that this module uses in the **imports array**.

Include all the [Angular Services](https://www.tektutorialshub.com/angular/angular-services/) that are part of this module in the [**providers**](https://www.tektutorialshub.com/angular/angular-providers/)**‘ array**.

The Component that angular should load, when the app.module is loaded is assigned to the bootstrap property.





There are several ways you can specify the Component selector:



