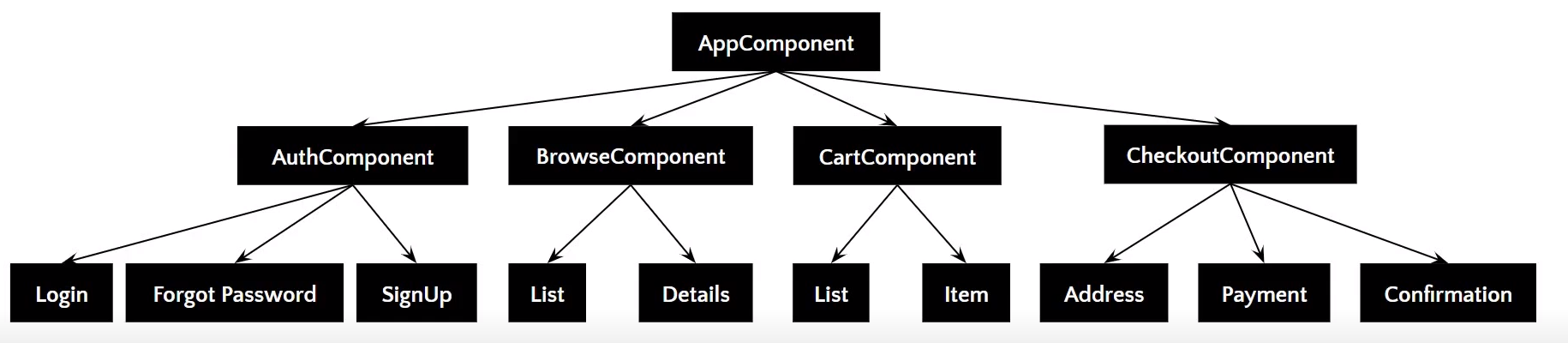
Angular Components

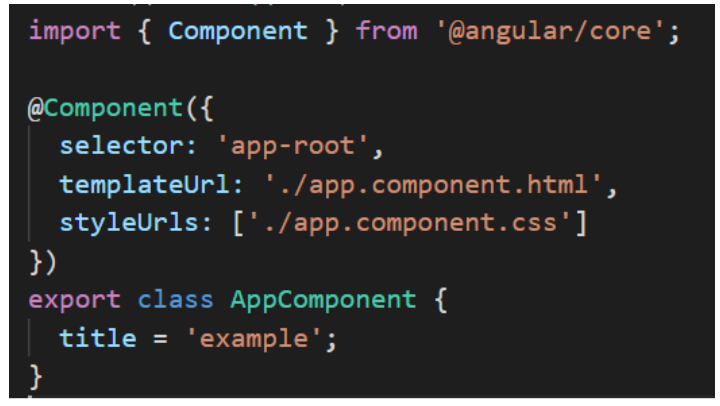
Components are the building blocks of a UI in an Angular application. These components are associated with a template and are a subset of directives.



The above image gives the tree structure of classification. There’s a root component, which is the AppComponent, that then branches out into other components creating a hierarchy.

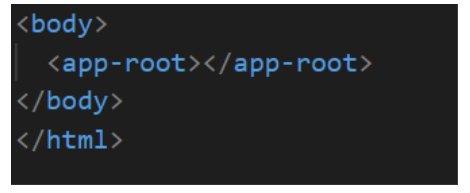
Here are some of the features of Angular Component:

* Components are typically custom [HTML](https://www.simplilearn.com/tutorials/html-tutorial/what-is-html) elements, and each of these elements can instantiate only one component.
* A TypeScript class is used to create a component. This class is then decorated with the “@Component” decorator.
* The decorator accepts a metadata object that gives information about the component.
* A component must belong to the NgModule in order for it to be usable by another component or application.
* Components control their runtime behavior by implementing Life-Cycle hooks.



The above image shows an App component, which is a pure TypeScript class decorated with the “@Component” decorator. The metadata object provides properties like selector, templateUrl, and so on—the templateUrL points to an HTML file that defines what you see on your application.

In the index.html file, <app-root> tag corresponds to component’s selector. By doing so, Angular will inject the corresponding template of the component.



Learn From The Best Mentors in the Industry!

Automation Testing Masters Program[EXPLORE PROGRAM](https://www.simplilearn.com/automation-testing-masters-program-certification-training-course?source=GhPreviewCTABanner)

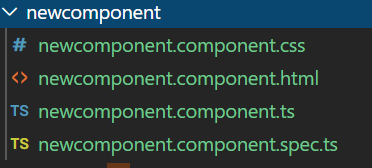


Creating Your First Angular Component

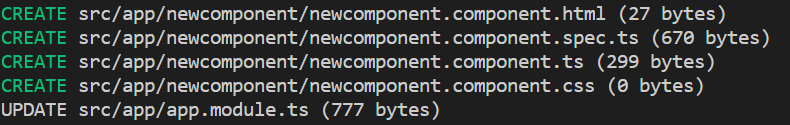
* To create an Angular Component, Angular CLI is used. In the terminal, type in the command,

  ng g c component-name

* This will create a folder named component-name with four files.



You’ll also receive a message:



The update message indicates that the component created is included in the declarations array of the main component.

It is crucial for Angular to know which component is to be run next and its features. For that, some metadata is created. The next section addresses the component metadata.

Component Decorator Metadata

As mentioned earlier, the @Component decorator accepts a metadata object that provides information about the component. Here’s a list of properties of the metadata object:

@Component({

  selector: 'app-root',

  template: `<h1>Hello! Welcome</h1>`,

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

  styles: [`

    h3{

      color: blue;

    }

  `],

  styleUrls: ['./app.component.css']

Selector

It is the CSS selector that identifies this component in a template. This corresponds to the HTML tag that is included in the parent component. You can create your own [HTML tag](https://www.simplilearn.com/tutorials/html-tutorial/html-tags). However, the same has to be included in the parent component.

Template

It is an inline-defined template for the view. The template can be used to define some markup. The markup could typically include some headings or paragraphs that are displayed on the UI.

TemplateUrl

It is the URL for the external file containing the template for the view.

Styles

These are inline-defined styles to be applied to the component’s view

styleUrls

List of URLs to stylesheets to be applied to the component’s view.

Providers

It is an array where certain services can be registered for the component

Animations

Animations can be listed for the components

Demo: Creating an Angular Component

Now that you have a good understanding of Angular components let me help you create an application using them.

Step 1: First, create a folder in your application to store all your components in.

ng g c components/new-component

Observe that the extension .component is appended to indicate that it is indeed a component.

Step2: Within the component, open the new-component.component.html file to type in whatever you’d like to see on the browser.

<h1>Hey! I'm the first component</h1>

Step3: In the new-component.component.ts file, copy the selector property to incorporate it in the app.component.html file.

@Component({

  selector: 'app-new-component',

  templateUrl: './new-component.component.html',

  styleUrls: ['./new-component.component.css']

})

In the app.component.html, which is the root component, go ahead and define the custom HTML tag. This indicates that the component created is being incorporated for the final render.

h1>Welcome to this tutorial on Angular Components</h1>

<app-new-component></app-new-component>

You can also define any styling conventions for the component in the CSS file.

h1 {

    text-align: center;

}

Once you execute the ng serve command, the output looks something like this.

