Directive

- Directives are components *without* a view. They are components without a template. Or to put it another way, components are directives *with* a view.
- Everything you can do with a directive you can also do with a component. But not everything you can do with a component you can do with a directive.
- We typically *associate* directives to existing elements by use *attribute* selectors, like so:
  - `<element aDirective></element>`
- At the core, a **directive** is a function that executes whenever the **Angular** compiler finds it in the DOM. **Angular directives** are used to extend the power of the HTML by giving it new syntax.
- Directives are instructions in the DOM. They specify how to place your components and business logic in the Angular.
- Directives are js class and declared as @directive. There are 3 directives in Angular.
  - Component Directives
  - Structural Directives
  - Attribute Directives
- **Component Directives:** Component directives are used in main class. They contain the detail of how the component should be processed, instantiated and used at runtime.
- **Structural Directives:** Structural directives start with a * sign. These directives are used to manipulate and change the structure of the DOM elements. For example, *ngIf and *ngFor.
- **Attribute Directives:** Attribute directives are used to change the look and behavior of the DOM elements. For example: ngClass, ngStyle etc.

Difference between Attribute Directive and Structural Directive

Attribute Directives	Structural Directives
Attribute directives look like a normal HTML Attribute and mainly used in databinding and event binding.	Structural Directives start with a * symbol and look different.
Attribute Directives affect only the element they are added to.	Structural Directives affect the whole area in the DOM.

How to Create Custom Directives?

- In this section, we will discuss about Custom Directives to be used in components. Custom directives are created by us and are not standard.
- Let us see how to create the custom directive. We will create the directive using the command line. The command to create the directive using the command line is –
  - ng g directive nameofthedirective
  - e.g
  - ng g directive changeText

This is how it appears in the command line

```
C:\projectA6\Angular6App>ng g directive changeText
CREATE src/app/change-text.directive.spec.ts (241 bytes)
CREATE src/app/change-text.directive.ts (149 bytes)
UPDATE src/app/app.module.ts (486 bytes)
```

The above files, i.e., **change-text.directive.spec.ts** and **change-text.directive.ts** get created and the **app.module.ts** file is updated.

app.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppComponent } from './app.component';
import { NewCmpComponent } from './new-cmp/new-cmp.component';
import { ChangeTextDirective } from './change-text.directive';
@NgModule({
  declarations: [
    AppComponent,
    NewCmpComponent,
    ChangeTextDirective
  ],
  imports: [
    BrowserModule
  ],
  providers: [],
```

```
  bootstrap: [AppComponent]
})
export class AppModule { }
```

The **ChangeTextDirective** class is included in the declarations in the above file. The class is also imported from the file given below.

change-text. directive

```
import { Directive } from '@angular/core';
@Directive({
  selector: '[appChangeText]'
})
export class ChangeTextDirective {
  constructor() { }
}
```

The above file has a directive and it also has a selector property. Whatever we define in the selector, the same has to match in the view, where we assign the custom directive.

In the **app.component.html** view, let us add the directive as follows −

```
<div style = "text-align:center">
  <span appChangeText >Welcome to {{title}}.</span>
</div>
```

We will write the changes in **change-text.directive.ts** file as follows −

change-text.directive.ts

```
import { Directive, ElementRef} from '@angular/core';
@Directive({
  selector: '[appChangeText]'
})
export class ChangeTextDirective {
  constructor(Element: ElementRef) {
    console.log(Element);
    Element.nativeElement.innerText = "Text is changed by changeText Directive. ";
  }
}
```

In the above file, there is a class called **ChangeTextDirective** and a constructor, which takes the element of type **ElementRef**, which is mandatory. The element has all the details to which the **Change Text** directive is applied.

We have added the **console.log** element. The output of the same can be seen in the browser console. The text of the element is also changed as shown above.

Now, the browser will show the following.