

# @Directive v/s @Component in Angular

▲ What is the difference between `@Component` and `@Directive` in Angular? Both of them seem to do the same task and have the same attributes.

366



What are the use cases and when to prefer one over another?

javascript

angular

typescript



83

edited May 31 '17 at 13:28



shammelburg

2,792 4 19 32

asked Sep 20 '15 at 14:03



Prasanjit Dey

2,124 2 8 15

- 12 A component is a *directive with a template* and the `@Component` decorator is actually a `@Directive` decorator extended with template-oriented features - [source](#). – [Cosmin Ababei](#) Apr 13 '16 at 12:20
- 2 Directive vs Component is the new Service vs Factory. The confusion is also increased because when actually requiring other components from a component definition you specify them in the `directives` array... maybe Lida Weng comment below helps a bit clarifying that the component "it's actually an extended 'Directive' " – [Nobita](#) Jul 10 '16 at 0:25
- 1 components actually extend directive, they just require you to have a template (HTML) as opposed to directives.. So you'd use directive to modify existing html element, and component makes html elements – [Marko Niciforovic](#) Nov 3 '16 at 14:17

## 7 Answers



**A `@Component` requires a view whereas a `@Directive` does not.**

## 467 Directives

Unlike a `@Directive` to an Angular 1.0 directive with the option `replace: true` (Directives aren't limited to attribute usage) Directives add

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google



Facebook

```
import {Directive} from '@angular/core';

@Directive({
  selector: "[logOnClick]",
  hostListeners: {
    'click': 'onClick()',
  },
})
class LogOnClick {
  constructor() {}
  onClick() { console.log('Element clicked!'); }
}
```

Which would be used like so:

```
<button logOnClick>I log when clicked!</button>
```

## Components

A component, rather than adding/modifying behaviour, actually creates its own view (hierarchy of DOM elements) with attached behaviour. An example use case for this might be a contact card component:

```
import {Component, View} from '@angular/core';

@Component({
  selector: 'contact-card',
  template: `
    <div>
      <h1>{{name}}</h1>
      <p>{{city}}</p>
    </div>
  `
})
class ContactCard {
  @Input() name: string
  @Input() city: string
  constructor() {}
}
```

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH

 Google

Facebook 

`ContactCard` is a reusable UI component that we could use anywhere in our application, even within other components. These basically make up the UI building blocks of our applications.

## In summary

Write a component when you want to create a reusable set of DOM elements of UI with custom behaviour. Write a directive when you want to write reusable behaviour to supplement existing DOM elements.

Sources:

- [@Directive documentation](#)
- [@Component documentation](#)
- [Helpful blog post](#)

edited Aug 27 '17 at 19:37



Lazar Ljubenović

10.8k 4 29 53

answered Sep 20 '15 at 16:39



jaker

5,514 1 15 18

2 does @directive annotation has template/templateUrl property ? – Pardeep jain Oct 28 '15 at 7:59

6 Is this answer still true? The angular2 tutorial itself creates a component without a view – Tamas Hegedus Jan 3 '16 at 1:10

it's without a view, but templateUrl or template are mandatory in the component – Luca Trazzi Jan 7 '16 at 16:54

3 I like this kind of answers, but I would really appreciate an update when crucial changes happen to the framework. – Memet Olsen May 9 '16 at 8:02

First two points links are dead now. – Shashank Vivek Sep 14 '16 at 10:06

## Components

1. To register a component we use `@Component` meta-data annotation.

2. Component is a directive which uses shadow DOM to create encapsulated visual behavior called components. Components are

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH

Google

Facebook

## Directive

1. To register directives we use `@Directive` meta-data annotation.
2. Directive is used to add behavior to an existing DOM element.
3. Directive is use to design re-usable components.
4. Many directives can be used per DOM element.
5. Directive doesn't use View.

Sources:

<http://www.codeandyou.com/2016/01/difference-between-component-and-directive-in-Angular2.html>

edited Sep 28 '17 at 20:20



Aniruddha Das

6,738 9 53 65

answered Jan 8 '16 at 10:20



virender

2,517 3 23 24

You can use as many component instances as you want – [Mihai Răducanu](#) May 26 '16 at 20:52

4 Components - point 4. I thinks it's wrong - it can be used multiple times. it's actually an extended 'Directive' – [Lida Weng](#) Jul 3 '16 at 10:31

Could have expanded this with examples. – [Mukus](#) Oct 10 '16 at 0:35

A component is a directive-with-a-template and the `@Component` decorator is actually a `@Directive` decorator extended with template-oriented features.

52

edited Apr 21 '16 at 22:02



madhead

15.7k 13 95 135

answered Apr 21 '16 at 12:06



yusuf tezel

641 7 15

3 Not sure why you've got downvoted too much. It seems that `@Component` is a Directive with a template (to generate view) for me. – [Harry Ninh](#) Jun 1 '16 at 2:48

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH

Google

Facebook

21 page, through both custom elements and attributes that add functionality to our existing components.

<http://learnangular2.com/components/>

But what directives do then in Angular2+ ?

Attribute directives attach behaviour to elements.

There are three kinds of directives in Angular:

1. Components—directives with a template.
2. Structural directives—change the DOM layout by adding and removing DOM elements.
3. Attribute directives—change the appearance or behaviour of an element, component, or another directive.

<https://angular.io/docs/ts/latest/guide/attribute-directives.html>

So what's happening in Angular2 and above is **Directives** are attributes which add functionalities to **elements** and **components**.

Look at the sample below from Angular.io:

```
import { Directive, ElementRef, Input } from '@angular/core';

@Directive({ selector: '[myHighlight]' })
export class HighlightDirective {
  constructor(el: ElementRef) {
    el.nativeElement.style.backgroundColor = 'yellow';
  }
}
```

So what it does, it will extends you components and HTML elements with adding yellow background and you can use it as below:

```
<p myHighlight>Highlight me!</p>
```

But components will create full elements with all functionalities like below:

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH

 Google

Facebook 

```
template: `
  <div>Hello my name is {{name}}.
  <button (click)="sayMyName()">Say my name</button>
</div>
`
})
export class MyComponent {
  name: string;
  constructor() {
    this.name = 'Alireza'
  }
  sayMyName() {
    console.log('My name is', this.name)
  }
}
```

and you can use it as below:

```
<my-component></my-component>
```

When we use the tag in the HTML, this component will be created and the constructor get called and rendered.

edited May 16 '18 at 0:04

answered Apr 16 '17 at 2:16



Alireza

58k

14

193

126

## Change detection

6

Only `@Component` can be a node in the change detection tree. This means that you cannot set `ChangeDetectionStrategy.OnPush` in a `@Directive`. Despite this fact, a Directive can have `@Input` and `@Output` properties and you can inject and manipulate host component's `ChangeDetectorRef` from it. So use Components when you need a granular control over your change detection tree.

answered Mar 30 '18 at 4:51



Evgeniy Malyutin

71k

0

11

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google

Facebook

## 4 behaviour.

“Directives allow you to attach behavior to elements in the DOM.”

directives are split into the 3 categories:

- Attribute
- Structural
- Component

Yes, in Angular 2, Components are a type of Directive. According to the Doc,

“Angular components are a subset of directives. Unlike directives, components always have a template and only one component can be instantiated per an element in a template.”

Angular 2 Components are an implementation of the **Web Component** concept. Web Components consists of several separate technologies. You can think of Web Components as reusable user interface widgets that are created using open Web technology.

- So in summary directives The mechanism by which we attach **behavior** to elements in the DOM, consisting of Structural, Attribute and Component types.
- Components are the specific type of directive that allows us to utilize **web component functionality** AKA reusability - encapsulated, reusable elements available throughout our application.

answered Jul 3 '18 at 12:49



[Sachila Ranawaka](#)

21.3k 4 31 55

If you refer the official angular docs

2 <https://angular.io/guide/attribute-directives>

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google

Facebook

3. Attribute directives—change the appearance or behavior of an element, component, or another directive. e.g [ngClass].

As the Application grows we find difficulty in maintaining all these codes. For reusability purpose, we separate our logic in smart components and dumb components and we use directives (structural or attribute) to make changes in the DOM.

answered Dec 19 '18 at 11:38



Akshay Rajput

856 2 8

**protected** by [Community](#) ♦ Apr 6 '17 at 12:57

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 [reputation](#) on this site (the [association bonus](#) does not count).

Would you like to answer one of these [unanswered questions](#) instead?

Join **Stack Overflow** to learn, share knowledge, and build your career.

Email Sign Up

OR SIGN IN WITH



Google

Facebook 