Using a directive to add class to host element

Asked 3 years ago Active 6 days ago Viewed 40k times

25

I am currently learning Angular 2. I understood how to use the Angular `Renderer` to set an `ElementStyle` , but now I would like to use the `Renderer` method:

```
setElementClass(renderElement: any, className: string, isAdd: boolean) : void
```

My question is how can I import a CSS class to my attribute directive? Do I have to convert my CSS class to JSON?

8

angular angular2-directives

edited Oct 16 '18 at 19:32

Alexander Abakumov
5,809 5 50 81

asked Sep 22 '16 at 12:29

Daniel Hoppe Alvarez
614 1 9 15

What do you mean by importing a css class? className is string so why you have to load or convert or import css class? – jaguwalapratik Sep 22 '16 at 12:52

3 Answers

58

Original OP was asking how to use Renderer. I've included the @HostBinding for completeness.

Using @HostBinding

To add a class to an element you can use @HostBinding

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

```
export class MyDirective {

  @HostBinding('class')
  elementClass = 'custom-theme';

  constructor() {
  }
}
```

Using @HostBinding with multiple classes

To make multiple classes more comfortable to use, you can use the ES6 getter and join the classes together before returning them:

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

@Directive({
  selector: '[myDirective]',
})
export class MyDirective {
  protected _elementClass: string[] = [];

  @Input('class')
  @HostBinding('class')
  get elementClass(): string {
      return this._elementClass.join(' ');
  }
  set(val: string) {
      this._elementClass = val.split(' ');
  }

  constructor() {
      this._elementClass.push('custom-theme');
      this._elementClass.push('another-class');
  }
}
```

Using Renderer

The more low level API is Renderer2. Renderer2 is useful when you have a dynamic set of classes you would like to apply to an element.

```
import { Directive, ElementRef, Renderer2 } from '@angular/core';

@Directive({
  selector: '[myDirective]',
})
export class MyDirective {

  constructor(private renderer: Renderer2, hostElement: ElementRef) {
    renderer.addClass(hostElement.nativeElement, 'custom-theme');
  }
}
```

edited Oct 9 at 16:46

developer033
16.1k 5 46 73

answered May 6 '17 at 12:43

cgatian
14.4k 4 42 58

I'm using for loop of classes array, is that practical? – Bear0x3f Sep 21 '17 at 3:21

For multiple classes you should perform a loop. If you take a look at the source you'll see Angular calls the classList.add method. While this native method supports multiple arguments, the Angular implementation only allows passing one class name at a time. – cgatian Nov 6 '17 at 14:11

1 pushing classes from the constructor doesn't seem to work. But it works from ngAfterViewInit – Adrien H May 12 at 13:37

1 The setter never seems to get called, causing any initial classes of the host to be lost :-(– Benjamin Kindle Jul 22 at 16:38

You don't need to DI the renderer or manipulate the existing classes instead use `@HostBinding('class.your-class')` - stackoverflow.com/a/58071653/1148107 – mtpultz Sep 24 at 0:41

7

Why would you want to use the Renderer or Renderer2 class? The preferred way to do this in a directive is to use the @HostBinding decorator.

Example:

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

@Directive({
    selector: '[myDirective]'
})
```

```
    className = 'my-directive-css-class';
}
```

edited May 24 at 12:55

Liam
17.4k 16 80 135

answered Sep 13 '17 at 2:46

csnate
2,321 1 10 3

1 This is useful if you want to add a class based on some other property, i.e. dynamically: – dannrob Nov 6 '17 at 11:46

5 I think this is only useful to completely replace the classes that may already be defined in the HTML. How would I *add* a class with this method? Renderer seems more flexible. – Simon_Weaver Jun 10 '18 at 2:35

3 This syntax is better : `@HostBinding('class.isPlaying') class_isPlaying = true;` (see this stackoverflow.com/a/35183074/16940). It will toggle classes as opposed to just wiping out the whole attribute. – Simon_Weaver Jun 10 '18 at 2:38

5

Example of how to use Renderer and ElementRef to add css class to element.

```
@Directive({
   selector: '[whatever]'
})
class WhateverDirective {
   constructor(renderer: Renderer, el: ElementRef) {
       renderer.setElementClass(el.nativeElement, 'whatever-css-class', true);
   }
}
```

The whatever-css-class is defined in a css file, which is referenced in html

edited May 24 at 12:55

Liam
17.4k 16 80 135

answered Sep 22 '16 at 12:56

jaguwalapratik
321 1 9

Yes absolutely, so far I understood the setElementClass function, but where lays the whatever-css-class, an attribute directive doesn't have an css file or is there a possibility to add one ? – Daniel Hoppe Alvarez Sep 22 '16 at 13:00

Right, so what you can do is define class in css file which you are referencing in html, so that way it will be applied to the element. – jaguwalapratik Sep

You don't need to DI the renderer instead use `@HostBinding('class.your-class')` - stackoverflow.com/a/58071653/1148107 – mtpultz Sep 24 at 0:36