

събота, 22.07.2017

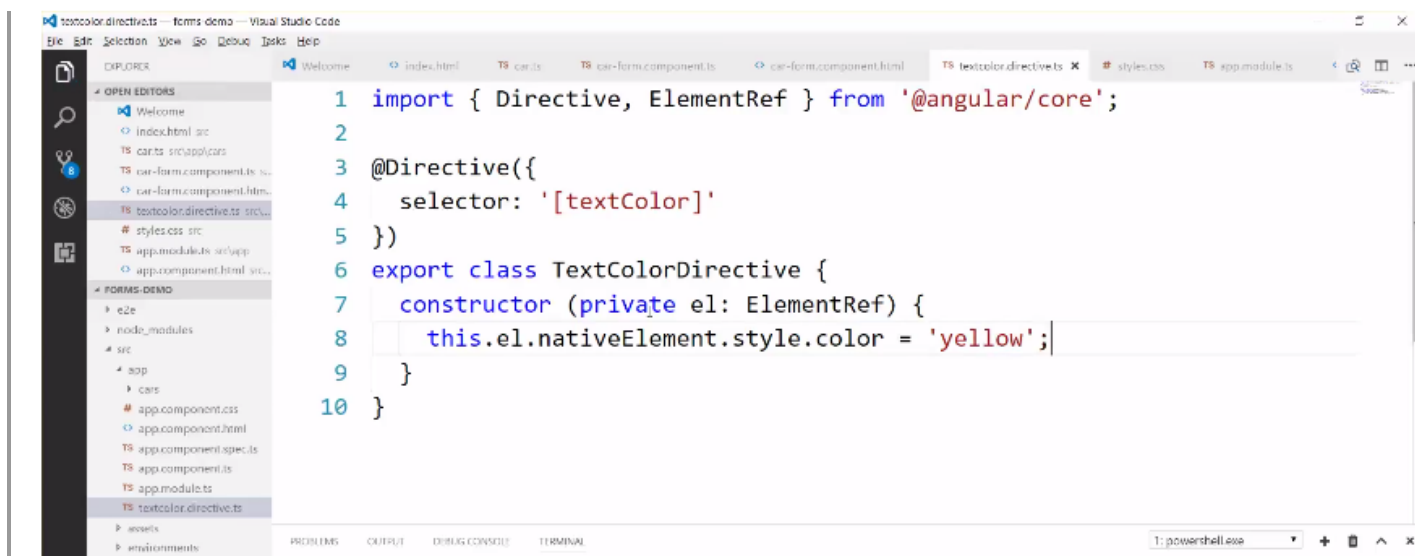
Directives

Накратко:

- 1. Първо трябва да си импортируем **Directive** и **ElementRef**
- 2. В конструктора трябва да вземем този **ElementRef** и вече можем да си правим каквото си искаме с него.
- 3. Директивата има **селектор**, но не и **template** и се използва **@Directive** като декоратор.
- 4. **ОСНОВНАТА РАЗЛИКА Е, ЧЕ ТЯ НЕ РЕНДЕРИРА НИЩО, А ПО-СКОРО ПРОМЕНЯ ДАДЕН ЕЛЕМЕНТ.**

Пример:

- 1. Да създадем файл **textcolor.directive.ts**



```
1 import { Directive, ElementRef } from '@angular/core';
2
3 @Directive({
4   selector: '[textColor]'
5 })
6 export class TextColorDirective {
7   constructor(private el: ElementRef) {
8     this.el.nativeElement.style.color = 'yellow';
9   }
10 }
```

*Тук **nativeElement** представлява връзката с елемента от дом-дървото, върху което ще се приложи директивата.*

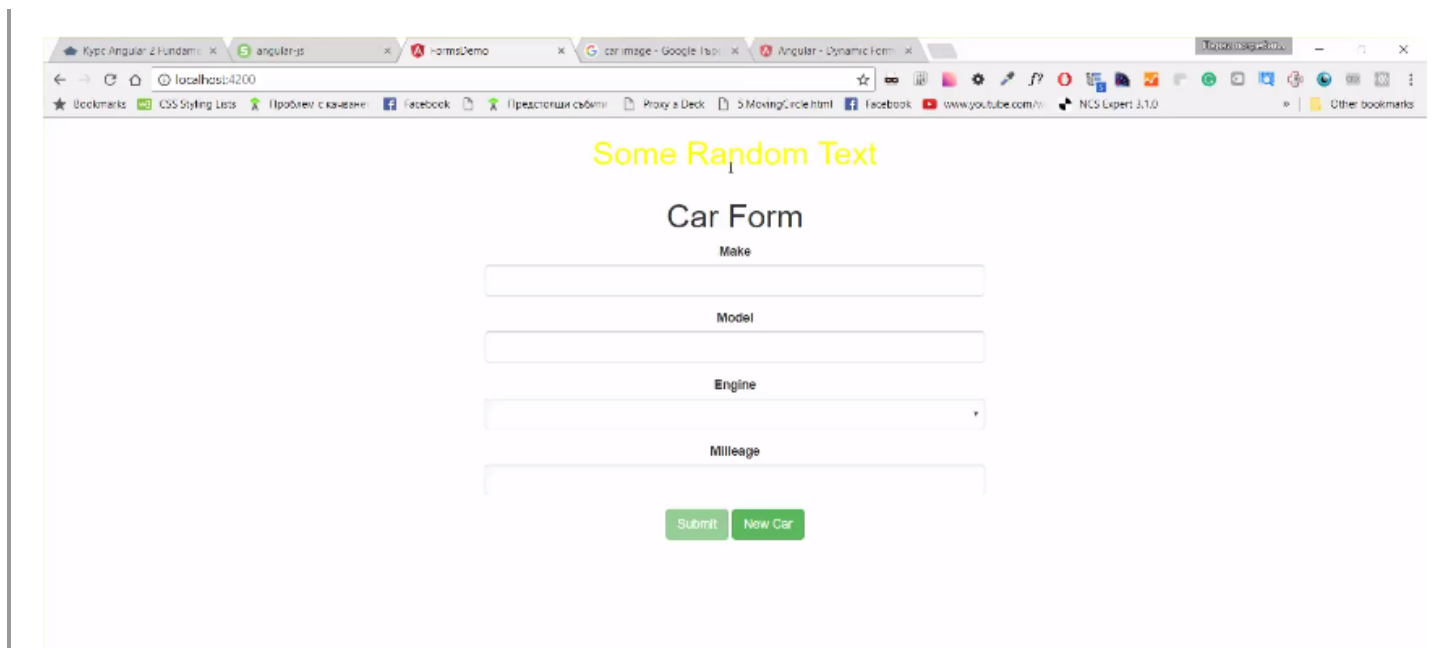
- 2 За да работи това нещо, първо трябва да се добави в декларациите в **app.module.ts**



```
4
5 import { AppComponent } from './app.component';
6
7 import { CarFormComponent } from './cars/car-form.component';
8
9 import { TextColorDirective } from './textcolor.directive';
10
11 @NgModule({
12   declarations: [
13     AppComponent,
14     CarFormComponent,
15     TextColorDirective
16 ],
```

- 3 И вече го използваме върху някакъв **html**

```
1 <div style="text-align:center">
2   <h1 textcolor></h1>
3
4   <car-form></car-form>
5 </div>
```



И ако няма никакви грешки , трябва html елемента да се промени. В случая да стане в жълт цвят.

Какво още ни се позволява да правим ?

- Да кажем , че искаме не винаги цветът ни да е жълто, а да го подадем отвън, тоест да можем да напишем следното:

```
1 <div style="text-align:center">
2   <h1 [textColor]="yellow">Some Random Text</h1>
3
4   <car-form></car-form>
5 </div>
```

3 а да се получи това трябва да направим следните неща:

1. Трябва да отидем във файла на директивата - в конкретния случай се казва **textcolor.directive.ts** и да импортируем **Input**

```
1 import { Directive, ElementRef, Input } from '@angular/core';
2
```

```
1 import { Directive, ElementRef, Input } from '@angular/core';
2
3 @Directive({
4   selector: '[textColor]'
5 })
6 export class TextColorDirective {
7   @Input("textColor") color: string;
8
9   constructor (private el: ElementRef) {
10     this.el.nativeElement.style.color = this.color;
11   }
12 }
```

- **ТУК ОБАЧЕ ИМАМ ПРОБЛЕМ И ТОЙ Е , ЧЕ ПОНАЧАЛО ЦВЕТЪТ НЕ Е ВКАРАН В КОНСТРУКТОРЪТ ВСЕ ОЩЕ. ЗАТОВА ЩЕ ГО НАПРАВИМ С Т. НАР. **HOSTLISTENER**.**

- **Идеята на HostListener-ът е да имаме евенти, защото те директно могат да се прикачат към елемента. Освен това чрез него Ангулар ще се грижи за това да ги слага и да ги маха, докато ако ръчно ги слагам примерно като `this.el.nativeElement.onclick = ...` в този случай ще трябва после и ръчно да го махам.**

- **С HostLister можем да напишем примерно:**

```
6 export class TextColorDirective {
7   @Input("textColor") color: string;
8
9   @HostListener('mouseenter') onMouseEnter () {
10     this.changeColor(this.color);
11   }
12
13   constructor (private el: ElementRef) { }
14
15   private changeColor(color) {
16     this.el.nativeElement.style.color = color;
17   }
18 }
```

```
6 export class TextColorDirective {
7   @Input("textColor") color: string;
8
9   @HostListener('mouseenter') onMouseEnter () {
10     this.changeColor(this.color);
11   }
12 }
```

```
12
13 @HostListener('mouseleave') onMouseLeave () {
14     this.changeColor(null);
15 }
16
17 constructor (private el: ElementRef) { }
18
```

Attribute Directives



```
import { Directive, ElementRef, HostListener, Input } from '@angular/core';
@Directive({
  selector: '[myHighlight]'
})
export class HighlightDirective {
  @Input('myHighlight') color: string;

  constructor (private el: ElementRef) { }
  @HostListener('mouseenter') onMouseEnter () {
    this.highlight(this.color);
  }
  @HostListener('mouseleave') onMouseLeave () {
    this.highlight(null);
  }
  private highlight(color: string) {
    this.el.nativeElement.style.color = color;
  }
}
```