Enhance Web Development

Angular 2

Directives

- Qu'est-ce qu'une directive ?
- Les directives d'attribut
- Les directives structurelles
- Les directives personnalisées

Ressources

- angular.io - Attribute directives
  https://angular.io/docs/ts/latest/guide/attribute-directives.html
- Fake server
  http://jsonplaceholder.typicode.com/

Qu'est-ce qu'une Directive ?

Qu'est-ce qu'une directive ?

- Une directive est responsable de la modification d'un template
- Un composant est un type de directive avec un template
- Au final ... Composant = Directive + template

Les directives structurelles et d'attribut

Les directives d'attribut

- ngClass
  https://angular.io/docs/ts/latest/guide/template-syntax.html#!#ngClass
- ngStyle
  https://angular.io/docs/ts/latest/guide/template-syntax.html#!#ngStyle

Les directives structurelles

- **ngIf**
  https://angular.io/docs/ts/latest/guide/template-syntax.html#!#ngIf
- **ngFor**
  https://angular.io/docs/ts/latest/guide/template-syntax.html#!#ngFor
- **ngSwitch**
  https://angular.io/docs/ts/latest/guide/template-syntax.html#!#ngSwitch

Les directives personnalisés

Les directives personnalisés

- Importer le décorateur **Directive** depuis le module **@angular/core**
- Créer une classe décorée avec **@Directive()**

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

@Directive({
  selector: '[highlight]'
})
export class HighlightDirective {
  // Code here ;)
}
```

Accéder au DOM

- Importer **ElementRef** depuis le module **@angular/core**
- Injecter **ElementRef** dans le constructeur de la directive

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

@Directive({
  selector: '[highlight]'
})
export class HighlightDirective {
  constructor(private _elementRef: ElementRef) {
    this._elementRef.nativeElement.style.backgroundColor = 'red'
  }
}
```

Accéder au DOM de la vue

- Importer le décorateur **ViewChild** depuis le module **@angular/core**
- Décorer un attribut de classe avec **@ViewChild()**
  - **@viewChild(string) = #string** dans le template
  - **@viewChild(Type) = Type** dans le template

```
@Component({
  selector: 'child-cmp',
  template: '<p>child</p>'
})
class ChildCmp {
  doSomething() {}
}

@Component({
  selector: 'some-cmp',
  template: '<child-cmp></child-cmp>',
  directives: [ChildCmp]
})
class SomeCmp {
  @ViewChild(ChildCmp) child:ChildCmp;
  ngAfterViewInit() {
    // child is set
    this.child.doSomething();
  }
}
```

```
@Component({
  selector: 'child-cmp',
  template: '<p>child</p>'
})
class ChildCmp {
  doSomething() {}
}

@Component({
  selector: 'some-cmp',
  template: '<child-cmp #child></child-cmp>',
  directives: [ChildCmp]
})
class SomeCmp {
  @ViewChild('child') child:ChildCmp;
  ngAfterViewInit() {
    // child is set
    this.child.doSomething();
  }
}
```

Répondre aux évènements

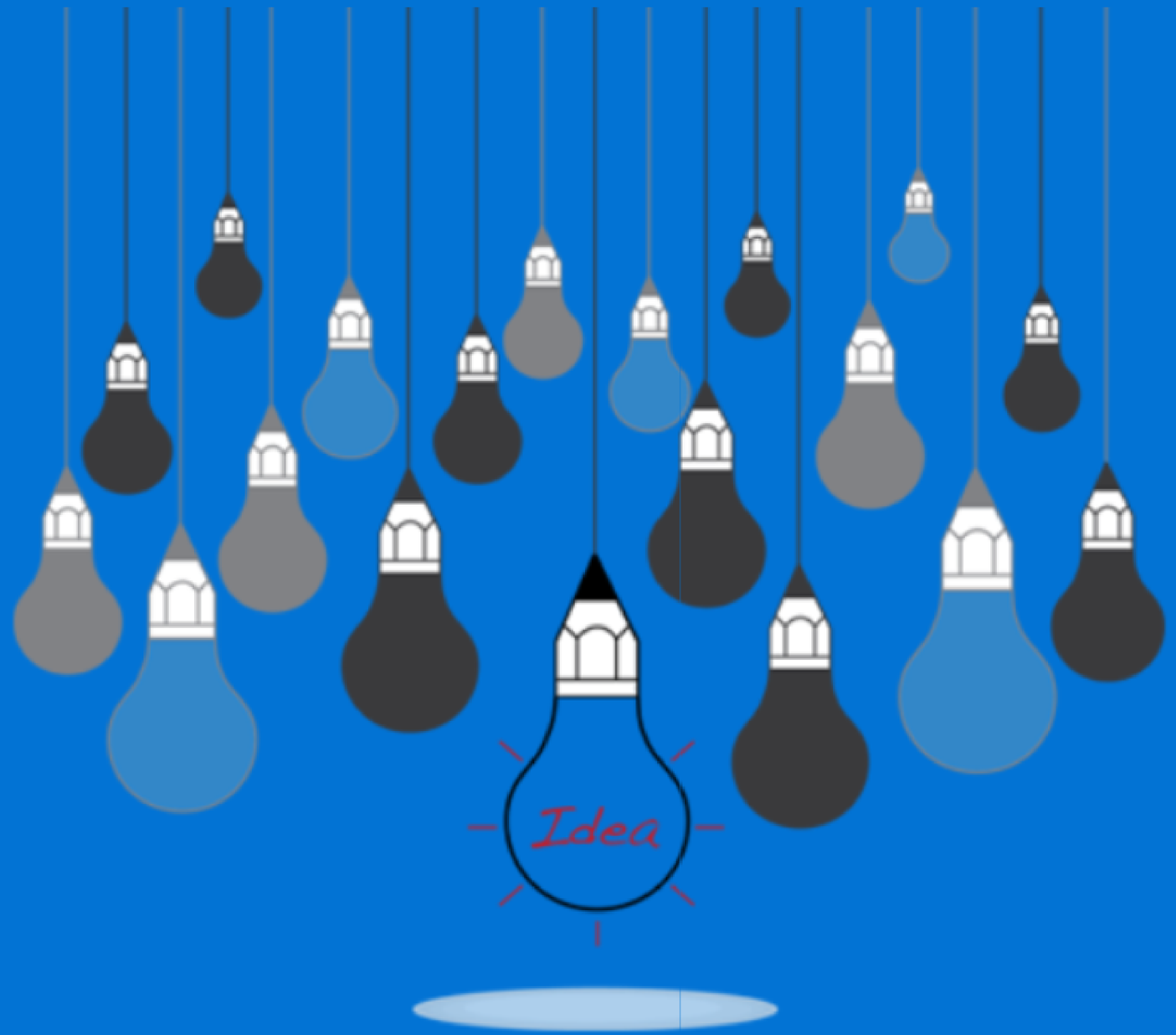
- Importer le décorateur **HostListener** depuis le module **@angular/core**
- Décorer une méthode dans la directive pour écouter les évènements du Host

```
@HostListener('mouseenter') onMouseEnter() {
  this.highlight('yellow');
}

@HostListener('mouseleave') onMouseLeave() {
  this.highlight(null);
}

private highlight(color: string) {
  this._elementRef.nativeElement.style.backgroundColor = color;
}
```

Demo time !



Défi

- Créer une directive **Blink** qui fait changer la couleur du texte toutes les 10 secondes
- Créer une directive qui met en gras tous les span de l'élément HTML
- Créer une directive qui wrap un plugin jQuery!