::ng-deep

- Le sélecteur ::ng-deep permet d’appliquer un sélecteur pour toute l’application
- Combinée avec :host, le selecteur est appliquée uniquement pour le composant et ses fils
- Peut être utilisée pour modifier le style d’un composant externe
- Dépréciée depuis longtemps, mais sans alternative pour le moment

Host binding et Host listener

Host listener

- Retour sur le composant folder :

```
<div (click)="onClick($event)" class="folder">
    <input [(ngModel)]="tree.value" />
    @if(expanded) {
        @for (child of tree.children; track $index) {
            <app-tree [tree]="child"/>
        }
    }
</div>
```

- Angular génère un élement dans le HTML pour chaque composant, et ici un div qui encapsule tout le composant

```
<app-root ng-version="17.3.5">
  <div class="content exercice">flex
    <h1>Recursive Component</h1>
    <app-tree _nghost-ng-c2259708039 ng-reflect-tree="[object Object]">
      <div _ngcontent-ng-c2259708039 class="folder">
        <input _ngcontent-ng-c2259708039 ng-reflect-model="root" class="ng-pristine ng-valid ng-touched">
        <app-tree _ngcontent-ng-c2259708039 _nghost-ng-c2259708039 ng-reflect-tree="[object Object]">
          <div _ngcontent-ng-c2259708039 class="folder">…</div>
        </app-tree>
        <app-tree _ngcontent-ng-c2259708039 _nghost-ng-c2259708039 ng-reflect-tree="[object Object]">
          <div _ngcontent-ng-c2259708039 class="folder">…</div>
        </app-tree>
```

- On a déja vu qu'il est possible de spécifier du style directement sur les composants avec le sélécteur :host
- De manière similaire, on peut écouter des évènements directement sur l'élément du composant

- Pour cela on utilise la propriété host dans le décorateur @Component

```
@Component({
...
  host: {
    '(click)': 'onClick($event)'
  }
})
```

- Il est aussi possible d'utiliser le décorateur @HostListener directement sur la méthode dans le composant

```
@HostListener('click', ['$event'])
onClick(event: Event) {
```

- La préconisation d'Angular est d'utiliser host plutôt que le décorateur @HostListener

Host binding

- Il est également possible de faire du style/class binding avec l'élément hote du composant
- Pour cela on utilise également la propriété host

```
@Component({
...
  host: {
    'class': 'box',
    '[style.font-size.px]': 'fontSize',
  }
})
```

- Ou le décorateur @HostBinding

```
@HostBinding('style.font-size.px')
fontSize = 60

@HostBinding('style.color')
color = 'red'
```

Exercice

- Refaire le composant tree-composant en enlevant le div global

Organisation d’un projet Angular

Smart vs dumb components

- On distingue deux types de composants :
  - Des composants de présentation, ou “dumb components”
  - Des composants conteneurs, ou “smart components”
- Les composants de présentation s’occupent de l’affichage des données, et n’ont que pas connaissance du reste de l’application (ils communiquent avec des @Input ou @Output)
- Les composants de présentation peuvent être réutilisés facilement
- Les composants conteneurs ont connaissance de l’état de l’application, et donnent aux composants de présentation uniquement les données dont ils ont besoin

Bonnes pratiques SNCF

- https://dev.sncf/docs/frontend/angular/
- Les bonnes pratiques présentées sont celles de la SNCF, pas des vérités absolues !

Création d’un projet

```
ng new --skip-git --inline-style --inline-template --directory ./ --routing --skip-tests --style scss {nom-du-projet}-cl
```

inline-style et inline-template

- Lors de la création d’un nouveau composant avec ng generate , le style et le template sont dans le même fichier
- La préconisation de la SNCF est de faire des single file component

skip-tests

- ng generate ne génère pas les fichiers de tests

ng generate

- https://angular.io/cli/generate
- Permet de générer et modifier automatiquement les fichiers du projet Angular pour ajouter des nouveaux éléments (Module, Composant, ...)
- Nomme automatiquement les classes et les fichiers en suivant les bonnes pratiques (https://angular.dev/style-guide)
- Le comportement de ng generate dépends du fichier angular.json à la racine du projet