Projet 1 : Appareils ménagers

I - Générer un premier composant

Générer un composant appareil avec la commande :

ng generate component appareil

II - Afficher les données par binding

1. Modifier la vue de l'appareilComponent

```
<li class="list-group-item">
  <h4>Ceci est dans AppareilComponent</h4>
</li>
```

et afficher trois composants appareil dans la vue du appComponent.

```
<div class="container">
  <div class="row">
    <div class="col-xs-12">
      <h2>Mes appareils</h2>
      <ul class="list-group">
        <app-appareil></app-appareil>
        <app-appareil></app-appareil>
        <app-appareil></app-appareil>
      </ul>
    </div>
  </div>
</div>
```

Résultat :

2. Dans le typescript du composant, créer une variable appareilName de type string représentant le nom de l'appareil. Exemple : "machine à laver".

```
export class AppareilComponent implements OnInit {

  appareilName: string = 'Machine à laver';

  constructor() { }
```

3. Afficher la variable appareilName dans la vue du composant appareil.

```
<li class="list-group-item">
  <h4>Appareil : {{ appareilName }}</h4>
</li>
```

4. Créer une variable booléenne isAuth représentant l'authentification de l'utilisateur.

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

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.scss']
})
export class AppComponent {
  isAuth = false;
}
```

5. Dans la vue du appComponent, créer un bouton pour allumer tous les composants. Il sera désactivé si la variable isAuth est à false, c'est-à-dire si l'utilisateur ne s'est pas authentifié. Il faudra donc mettre la variable isAuth à true pour tester.

```
<div class="container">
  <div class="row">
    <div class="col-xs-12">
      <h2>Mes appareils</h2>
      <ul class="list-group">
        <app-appareil></app-appareil>
        <app-appareil></app-appareil>
        <app-appareil></app-appareil>
      </ul>
      <button class="btn btn-success"
              [disabled]="!isAuth"
              (click)="onAllumer()">Tout allumer</button>
    </div>
  </div>
</div>
```

Dans le appComponent, créer la méthode onAllumer() qui affiche “on allume tout”.

```
onAllumer() {
    console.log('On allume tout !');
}
```

6. Ajouter une variable appareilStatus de type string au composant appareil, ayant pour valeur “allumé”.

7. Ajouter un champ input en two-way binding au composant appareil, qui sera attaché à la variable appareilName par un ngModel. Modifier la valeur du nom dans le champ input et tester la mise à jour de l’affichage d’appareilName.

```
<li class="list-group-item">
  <h4>Appareil : {{ appareilName }} -- Statut : {{ getStatus() }}</h4>
  <input type="text" class="form-control" [(ngModel)]="appareilName">
</li>
```

Ne pas oublier d’importer le module FormsModule pour que le two-way binding fonctionne.

```
import { AppComponent } from './app.component';
import { MonPremierComponent } from './mon-premier/mon-premier.component';
import { AppareilComponent } from './appareil/appareil.component';
import { FormsModule } from '@angular/forms';
```

III - Les propriétés personnalisées

1. Mettre le champ appareilName du composant appareil en @Input, et effacer sa valeur “machine à laver”.

```
export class AppareilComponent implements OnInit {

  @Input() appareilName: string;

  appareilStatus: string = 'éteint';

  constructor() { }

  ngOnInit() {
  }

  getStatus() {
    return this.appareilStatus;
  }

}
```

2. Dans le appComponent, ajouter les valeurs des noms des trois différents appareils qu’on souhaite créer.

```
export class AppComponent {
  isAuth = false;

  appareilOne = 'Machine à laver';
  appareilTwo = 'Frigo';
  appareilThree = 'Ordinateur';

  constructor() {
```

3. Passer trois valeurs différentes aux trois composants fils appareils.

```
<ul class="list-group">
    <app-appareil [appareilName]="appareilOne"></app-appareil>
    <app-appareil [appareilName]="appareilTwo"></app-appareil>
    <app-appareil [appareilName]="appareilThree"></app-appareil>
</ul>
```

Vérifier que les trois composants s'affichent avec des noms différents.

IV - Les directives

1. Avec un ngIf, ajouter une div de couleur rouge ne s'affichant que si le status de l'appareil est "eteint".

```
<li class="list-group-item">
  <div style="width:20px;height:20px;background-color:red;"
       *ngIf="appareilStatus === 'éteint'"></div>
  <h4>Appareil : {{ appareilName }} -- Statut : {{ getStatus() }}</h4>
  <input type="text" class="form-control" [(ngModel)]="appareilName">
</li>
```

2. Ajouter un tableau d'appareils dans le appComponent.

```
export class AppComponent {
  isAuth = false;

  appareils = [
    {
      name: 'Machine à laver',
      status: 'éteint'
    },
    {
      name: 'Frigo',
      status: 'allumé'
    },
    {
      name: 'Ordinateur',
      status: 'éteint'
    }
  ];

  constructor() {
```

3. Afficher ces composants avec un ngFor dans la vue du appComponent.

```
<div class="container">
  <div class="row">
    <div class="col-xs-12">
      <h2>Mes appareils</h2>
      <ul class="list-group">
        <app-appareil  *ngFor="let appareil of appareils"
                       [appareilName]="appareil.name"
                       [appareilStatus]="appareil.status"></app-appareil>
      </ul>
      <button class="btn btn-success"
              [disabled]="!isAuth"
              (click)="onAllumer()">Tout allumer</button>
    </div>
  </div>
</div>
```

4. Créer une méthode getColor() retournant red si le statut est éteint, ou green si le statut est allumé.

```
getColor() {
    if(this.appareilStatus === 'allumé') {
      return 'green';
    } else if(this.appareilStatus === 'éteint') {
      return 'red';
    }
}
```

5. Mettre le retour de la méthode getColor à l'attribut color de la directive ngStyle.

```
<h4 [ngStyle]="{color: getColor()}">Appareil : {{ appareilName }} -- Statut : {{ getStatus() }}</h4>
```

V - Les services

Nous allons créer un service appareilService, nous permettant de centraliser les fonctionnalités et données de notre application.

1. Dans le dossier app, créer un sous dossier services. Dans le dossier app/services, créer un service appareil avec la commande suivante :

   ng generate service appareil

2. Dans le fichier appModule.ts, importer le service de la même façon que sont importés les autres modules.

3. Injecter l'instance du service importé dans le tableau providers du appModule.

4. Dans le constructeur du appComponent, créer une instance privée du appareilService.

5. Déplacer dans le service le tableau d'appareils qui était dans le appComponent.

6. Dans le service, créer une méthode getAppareils() qui retourne le tableau d'appareils.

7. Implémenter la méthode ngOnInit() du appComponent qui initialise un tableau d'appareils récuperé du appareilService avec la méthode getAppareils().

8. Dans le service, créer des méthodes switchOnAll() qui passe tous les appareils à allumé, et switchOfAll() qui passe tous les appareils à éteint.

9. Dans la vue du appComponent, créer un bouton pour allumer tous les composants, et un autre pour les éteindre tous.

10. Dans les appComponent, créer des méthodes onAllumer() et onEteindre() qui allument ou éteignent tous les composants du service.