Etablissement : ISET-Charguia	**Département :** Technologies de l'Informatique
Matière : Atelier Framework Côté Client	**Année :** 2ème année DSI
Année Universitaire : 2019 - 2020	

TP n° 5 : Les services

Objectifs du TP :

- Comprendre l'intérêt des services
- Créer et utiliser un service dans une application Angular

Application 1 :

L'objectif de cette application est de créer un service qui renvoie la date du jour et qui est appelé dans différentes composants de l'application

1. Accéder à votre répertoire TP qui comprend l'ensemble des TP du semestre et y créer un nouveau projet Angular nommé **TP5**
2. Ajouter au projet **3** composants nommés respectivement *principal*, *first* et *second* qui seront placés dans un dossier « application » et faire appel au composant *principal* dans *app.component.html*
3. Modifier les templates des différents composants de sorte que *principal* appelle *first* qui fait appel à son tour à *second* (*second* a pour parent *first* qui a pour parent *principal*).
4. Créer sous le répertoire application un service ***date*** en tapant dans le terminal :

ng g s application/date

5. Ajouter à la classe DateService une méthode ***dateJour*** qui renvoie un objet Date qui comporte la date du jour.
6. On souhaite faire appel à ce service dans le composant principal. Pour cela :
   - Ajouter à la classe PrinicpalComponent un attribut laDate de type Date
   - Injecter le service dans le constructeur comme suit

```
laDate: Date;
constructor(private service:DateService) {}

ngOnInit() {
  // Récupération de la date à partir du service
   this.laDate = this.service.dateJour();
```

   - Ajouter à la liste des importations

```
import {DateService} from '../date.service';
```

   - Afficher la date récupérée dans le template du composant *principal* : `{{laDate}}`
   - Observer le résultat obtenu dans *localhost :4200*

7. Faire appel au service DateService dans le composant *second* et l'afficher, puis commenter.

Exercice 1:

On souhaite développer une application Angular permettant de gérer les employés (figure 1)

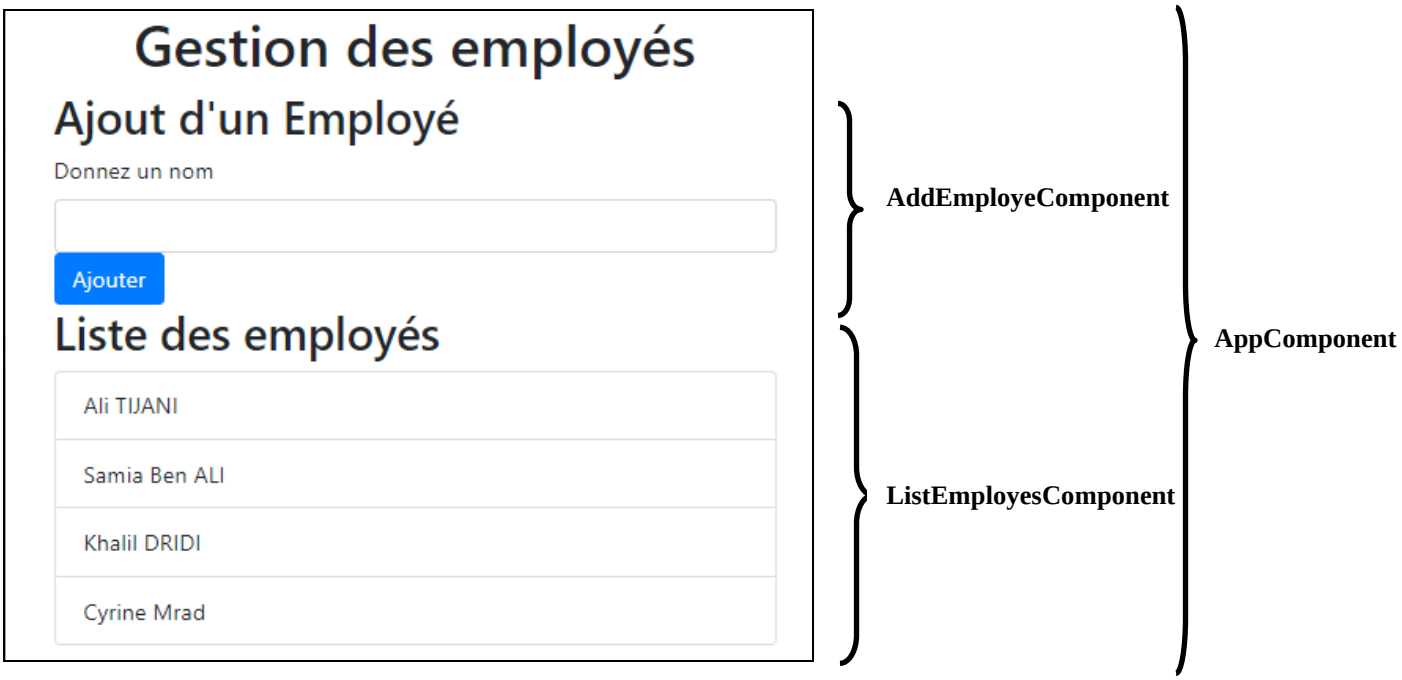

Figure 1: Gestion des employés

L'utilisateur introduit le nom de l'employé, en cliquant sur le bouton « Ajouter », le nom s'ajoute à la liste des employés.

Travail demandé

1. Ajouter au projet les composants *addEmploye* et *listEmployes* à placer dans un répertoire nommé *ex1*.
2. Définir dans le répertoire ex1 un service *employe* décrit par :
   - Un attribut *employes* qui définit un tableau ayant quelques noms d'employés
   - Une méthode *addEmploye(nom :string)* qui ajoute au tableau le nom d'un employé passé en paramètre.
3. Implémenter les composants en injectant le service employe pour répondre au fonctionnement demandé (utiliser les classes bootstrap nécessaires vues dans le TP4 et vous pouvez utiliser la classe form-control pour la zone de texte).

I

Exercice 2:

On souhaite développer une application Angular permettant de gérer les promotions sur des livres (figure 2)

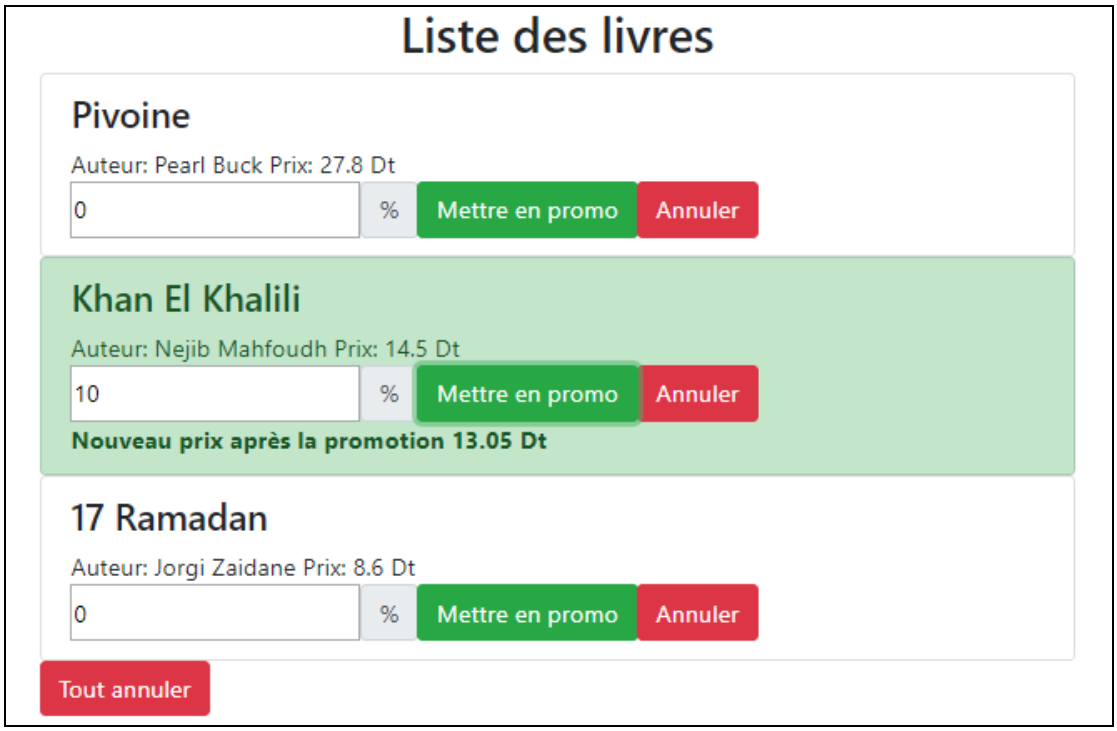

Figure 2: Gestion des promotions des livres

L'interface affiche pour chaque livre son titre, son auteur, son prix et son % de réduction (0 par défaut).

L'utilisateur peut introduire, pour un livre donné, un pourcentage de réduction qui ne sera appliqué au prix que s'il clique sur le bouton « Mettre en promo ». Ainsi, les livres en promotion sont colorés en vert et le nouveau prix après réduction s'affiche.

Le bouton « Annuler » permet de modifier le pourcentage de réduction à 0.

Le bouton « Tout annuler » remet toutes les réductions à 0 pour tous les livres.

Travail demandé

1. Ajouter au projet deux composants *livre* et *listLivres* qui seront placés dans un répertoire nommé *ex2*.
2. Ajouter au répertoire *ex2* une classe **Livre**(.ts) décrite par :
   - Les attributs *titre*, *auteur*, *prix*, et *reduction*
   - Un constructeur paramétré sachant que par défaut un nouveau livre ne bénéficie d'aucune réduction

3. Ajouter au répertoire *ex2* un service **livre**(.service.ts) décrit par :

   - Un attribut *Livres* qui renferme un tableau de livres
   - Une méthode *mettreEnPromo (reduction:number, index:number)* qui prend en paramètre le pourcentage de réduction et l'indice dans le tableau. La méthode modifie le taux de réduction du livre associé et renvoie le prix après réduction
   - Une méthode *estEnPromo(index:number)* qui renvoie si un livre dont l'index est passé en paramètre est en promotion ou non
   - Une méthode *addLivre (l :livre)* qui ajoute dans le tableau des livre un livre passé en paramètre
   - Une méthode *annulerReductionLivre(index :number)* qui remet à 0 la réduction d'un livre dont l'indice est passé en paramètre
   - Une méthode *annulerReduction()* qui remet à 0 toutes les réductions de tous les livres

4. Compléter l'implémentation des composants **livre** et **listLivres** pour répondre au fonctionnement demandé

Indications pour la mise en forme : vous pouvez utiliser les classes bootstrap **input-group-append** et **input-group-text**
https://www.w3schools.com/bootstrap4/bootstrap_forms_input_group.asp