

# **Formation Angular 11**

---

## **TP N°3: Injection de dépendances (IOC) dans Angular**

---

# SOMMAIRE

1.	Pré-requis : .....	3
2.	Objectifs .....	3
3.	Développement .....	3
a.	Création de l'application dependency .....	3
b.	Création du composant Employee .....	4
c.	Création du service Records .....	5
d.	Implémentation des services fournis par la classe RecordsService .....	5
e.	Modification de la classe EmployeeComponent .....	6
f.	La page employee.component.html .....	7
g.	La page app.component.html.....	7

### 1. Pré-requis :

- Réaliser les TPs 1-2 en premier.

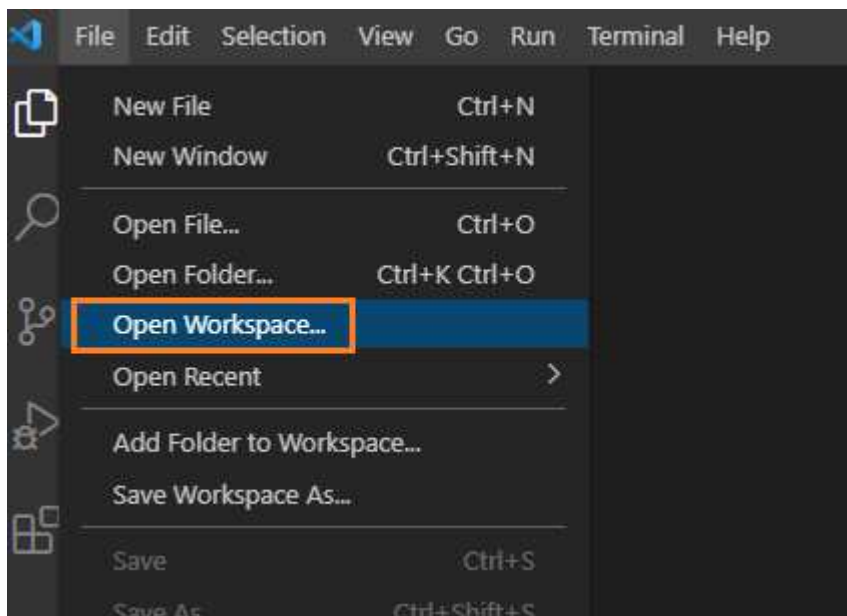
### 2. Objectifs

- ✓ Créer un service.
- ✓ Injecter un service dans un composant.
- ✓ Utiliser la directive \*ngFor.

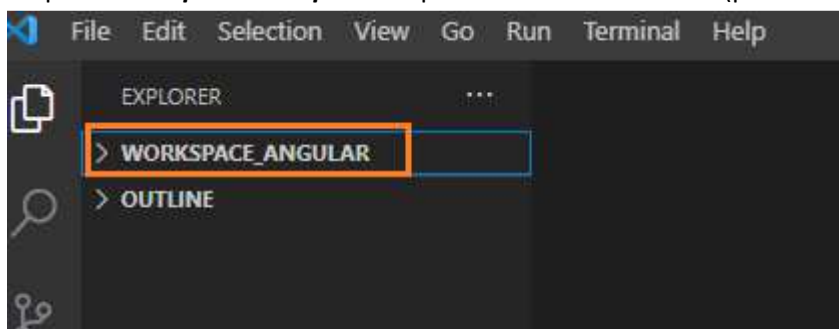
### 3. Développement

#### a. Création de l'application dependency

- Lancer VSC et ouvrir votre Workspace en suivant les étapes suivantes :



- Cliquer sur « **Open Workspace...** » puis choisir votre dossier (par exemple c:/workspace\_angular) :



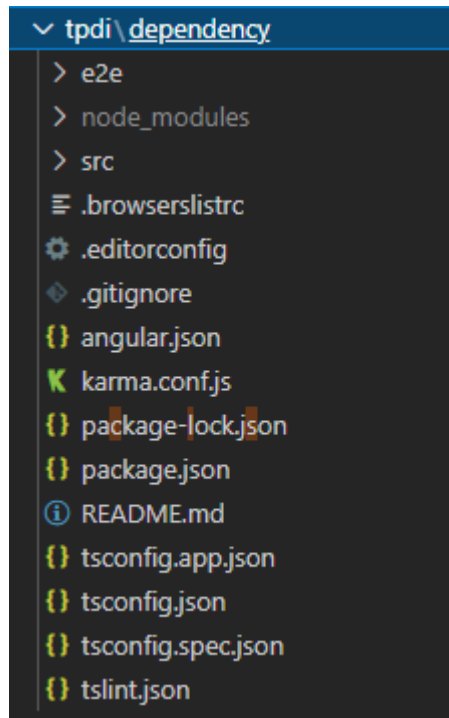
- Ouvrir ensuite un terminal et taper la commande suivante pour créer le projet **dependency** :

```

PS C:\workspace_Angular\tpdi> ng n dependency
? Do you want to enforce stricter type checking and stricter bundle budgets in the workspace?
  This setting helps improve maintainability and catch bugs ahead of time.
  For more information, see https://angular.io/strict Yes
? Would you like to add Angular routing? Yes
? Which stylesheet format would you like to use? CSS
CREATE dependency/angular.json (3655 bytes)
CREATE dependency/package.json (1200 bytes)

```

- L'application angular *dependency* ensuite générée par Angular CLI.



#### b. Création du composant Employee

- Dans le terminal, lancer les commandes suivantes :

```

PS C:\workspace_Angular\tpdi> cd .\dependency\
PS C:\workspace_Angular\tpdi\dependency> ng g c employee
CREATE src/app/employee/employee.component.html (23 bytes)
CREATE src/app/employee/employee.component.spec.ts (640 bytes)
CREATE src/app/employee/employee.component.ts (283 bytes)
CREATE src/app/employee/employee.component.css (0 bytes)
UPDATE src/app/app.module.ts (483 bytes)
PS C:\workspace_Angular\tpdi\dependency>

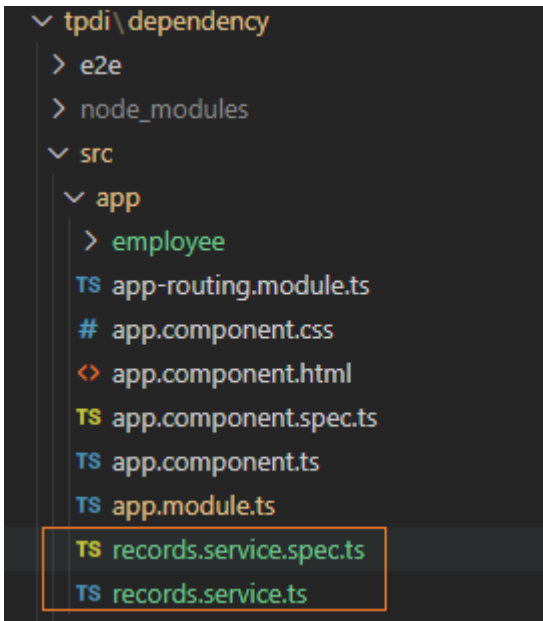
```

### c. Création du service Records

- Lancer la commande suivante :

```
PS C:\workspace_Angular\tpdi\dependency> ng g s records
CREATE src/app/records.service.spec.ts (362 bytes)
CREATE src/app/records.service.ts (136 bytes)
```

- Vérifier que les deux fichiers suivants ont été créés :



```
▼ tpdi\dependency
  > e2e
  > node_modules
  ▼ src
    ▼ app
      > employee
      TS app-routing.module.ts
      # app.component.css
      <> app.component.html
      TS app.component.spec.ts
      TS app.component.ts
      TS app.module.ts
      TS records.service.spec.ts
      TS records.service.ts
```

### d. Implémentation des services fournis par la classe RecordsService

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

@Injectable({
  providedIn: 'root'
})
export class RecordsService {

  info1 : string[]=["ALAOUI", "Mohammed","22"]
  info2 : string[]=["Amrani", "Ahmed","40"]
  info3 : string[]=["Abbou", "Samir","34"]

  getInfo1():string[] {
    return this.info1
  }

  getInfo2():string[] {
    return this.info2
  }
}
```

```

    getInfo3():string[] {
        return this.info3
    }
    constructor() { }
}

```

#### e. Modification de la classe EmployeeComponent

```

import { Component, OnInit } from '@angular/core';
import {RecordsService} from "../records.service";

@Component({
  selector: 'employee',
  templateUrl: './employee.component.html',
  styleUrls: ['./employee.component.css'],
  providers : [RecordsService]
})
export class EmployeeComponent implements OnInit {
  employee1 : string[]=[]
  employee2 : string[]=[]
  employee3 : string[]=[]
  constructor(private service:RecordsService) { }

  public getEmployee1() {
    this.employee1=this.service.getInfo1();
  }

  public getEmployee2() {
    this.employee2=this.service.getInfo2();
  }

  public getEmployee3() {
    this.employee3=this.service.getInfo3();
  }

  ngOnInit(): void {
  }
}

```

f. La page employee.component.html

```
<button type="button" name="button" (click)='getEmployee1()'>Employee1</button>
<ul class="list-group">
<li *ngFor="let info of employee1" class="list-group-item">{{info}}</li>
</ul>

<pre></pre>

<button type="button" name="button" (click)='getEmployee2()'>Employee2</button>
<ul class="list-group">
<li *ngFor="let info of employee2" class="list-group-item">{{info}}</li>
</ul>

<pre></pre>

<button type="button" name="button" (click)='getEmployee3()'>Employee3</button>
<ul class="list-group">
<li *ngFor="let info of employee3" class="list-group-item">{{info}}</li>
</ul>
```

g. La page app.component.html

```
<h1>Liste des employés :</h1>
<employee></employee>
```

- Démarrer ensuite le serveur et lancer le lien suivant :



- Cliquer sur les boutons ci-dessous et vérifier les données affichées :



C'est le principe d'injection de dépendances (IOC). Dans cet exemple, le service RecordsService a été injecté au niveau du composants Employee.

Remarquez que nous pouvons créer un autre composant et injecter le même service RecordsService.

Vous venez d'injecter votre premier service sur un composant !

Vous pouvez passer au TP n°4.

**Fin du TP 3.**