

Les services

Comme nous l'avons vu dans l'introduction, le framework Spring utilise massivement le principe d'injection de dépendances.

Injection de dépendance

L'injection de dépendances consiste à une classe instanciée par le framework (comme par exemple nos contrôleurs), de se faire fournir les classes dont elle dépend par une partie du framework qui s'appelle le conteneur d'injection de dépendance (aussi appelée conteneur d'inversion de contrôle). Cela permet donc de découpler intégralement une classe de ses dépendances grâce à la programmation par interface.

Component Scan

L'injection de dépendance dans Spring Boot utilise un mécanisme appelé le Component Scan, qui permet au conteneur d'injection de dépendance de détecter automatiquement les classes à injecter grâce à des annotations. La principale est l'annotation `@Component` mais elle possède des alias sémantiques (ils font la même chose mais permettent de donner plus de sens) comme par exemple `@Service`. Nous allons donc principalement utiliser `@Service`.

Pourquoi les services ?

Le but des services est de séparer la logique propre à l'application de la logique HTTP (qui réside dans les contrôleurs), les contrôleurs ne doivent avoir pour responsabilité que de gérer des requêtes et réponses HTTP et gérer les erreurs proprement. Pour tout traitement logique, nous allons utiliser un service. Le service va prendre en entrée les données traitées par le contrôleur, effectuer le traitement logique, et si besoin retourner une réponse au contrôleur.

Premier service

Pour créer un service il faut d'abord définir son contrat de service sous la forme d'une interface :

```
public interface TodoService {
```

```
...  
  
}
```

Ensuite, il faut fournir une implémentation de cet interface :

```
@Service  
public class TodoServiceImpl implements TodoService {  
  
    ...  
  
}
```

On utilise l'annotation `@Service` pour signaler au conteneur d'injection de dépendance qu'il s'agit d'une classe à scanner. Enfin, notre contrôleur pourra déclarer ce service comme dépendance en le prenant en paramètre de son constructeur :

```
public class TodoController {  
  
    private final TodoService todoService;  
  
    public TodoController(TodoService todoService) {  
        this.todoService = todoService;  
    }  
  
}
```

Ainsi, lors de la construction du contrôleur par le framework, notre implémentation de `TodoService` lui sera fournie par le conteneur d'injection de dépendance. De cette façon, le contrôleur pourra utiliser le service établi par le contrat de service défini par `TodoService` sans dépendre d'aucune façon de son implémentation.

Revision #5

Created 2021-01-22 09:46:19 UTC by Arsène Lapostolet

Updated 2021-01-28 15:56:00 UTC by Arsène Lapostolet