Spring HATEOAS

Introduction

Qu'est-ce que Spring HATEOAS ?

- Spring HATEOAS est un projet qui permet de faciliter la création de services RESTful qui suivent le principe HATEOAS.
- HATEOAS (Hypermedia As The Engine Of Application State) est un principe de conception d'API RESTful qui permet de rendre les services plus découvrables et plus facilement utilisables.
- Spring HATEOAS fournit des classes utilitaires pour faciliter la création de liens hypertextes dans les réponses des services RESTful.

Qu'est-ce que Spring HATEOAS ?

- Il vise à simplifier la création de services web RESTful hypermedia-driven.
- L'idée est que les réponses d'un service REST doivent contenir non seulement les données demandées mais aussi des hyperliens vers d'autres ressources pertinentes.
- Cela permet aux clients de naviguer dans l'application en utilisant uniquement les liens fournis dans les réponses, rendant l'application plus découvrable et plus facile à intégrer.

Dépendance

- Pour utiliser Spring HATEOAS dans un projet Spring Boot, il suffit d'ajouter la dépendance suivante dans le fichier `pom.xml` :

```xml
<dependency>
    <groupId>org.springframework.hateoas</groupId>
    <artifactId>spring-hateoas</artifactId>
</dependency>
```

Implémentation Sans HateOAS

- Sans HateOAS, un service RESTful peut retourner simplement les données demandées par le client.
- Par exemple, avec un model `Users` qui contient les informations d'un utilisateur, le service peut retourner une liste d'utilisateurs :

Implémentation Sans HateOAS

````
## Exemple

```java
public class Users {
    private String nom;
    private Long salaire;

    public Users(String nom, Long salaire) {
        this.nom = nom;
        this.salaire = salaire;
    }

    // constructeurs, getters et setters
}
````

Implémentation Sans HateOAS

```java
import org.springframework.hateoas.Resource;
import org.springframework.hateoas.mvc.ControllerLinkBuilder;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequestMapping("/api/users")
public class UsersController {

    @GetMapping("/all")
    public List<Users> getAll() {

        Users users = new Users('John Doe', 2300L);
        Users users2 = new Users('Jane Doe', 2500L);

        return Arrays.asList(users , users2);
    }
}
```

Implémentation sans HateOAS

- Dans cet exemple, la méthode `getAll` retourne une liste d'utilisateurs sans aucun lien hypertexte.
- Cela signifie que le client doit connaître à l'avance les URLs des services pour pouvoir naviguer dans l'application.
- Cela rend l'application moins découvrable et plus difficile à intégrer.
```

Implémentation avec HateOAS

- Avec Spring HATEOAS, il est possible de retourner des liens hypertextes dans les réponses des services RESTful.
- Cela permet de rendre les services plus découvrables et plus facilement utilisables.
- Par exemple, on peut ajouter des liens hypertextes aux ressources retournées par le service pour permettre au client de naviguer dans l'application.

Exemple

```
public class Users extends RepresentationModel{ // ou ResourceSupport dans les
ancienne versions
    private String nom;
    private Long salaire;

    public Users(String nom, Long salaire) {
        this.nom = nom;
        this.salaire = salaire;
    }

    // constructeurs, getters et setters
}
```

```
@RestController
@RequestMapping("/api/users")
public class UsersController {

 GetMapping(value = "/hateoas/all", produces = MediaTypes.HAL_JSON_VALUE)
    public List<Users> getHateOASAll() {

        Users users = new Users('John Doe', 2300L);
        Link namelink =
ControllerLinkBuilder.linkTo(UsersController.class).slash(users.getNom()).withSelf
Rel();
        Link salairelink =
ControllerLinkBuilder.linkTo(UsersController.class).slash(users.getSalaire()).with
SelfRel();
        users.add(namelink, salairelink);

        Users users2 = new Users('Jane Doe', 2500L);
        Link namelink2 =
ControllerLinkBuilder.linkTo(UsersController.class).slash(users2.getNom()).withSel
fRel();
```

```
        Link salairelink2 =
ControllerLinkBuilder.linkTo(UsersController.class).slash(users2.getSalaire()).wit
hSelfRel();
        users2.add(namelink2, salairelink2);

        return Arrays.asList(users , users2);
    }
}
```

Exemple

- Le lien hypertexte est créé avec la méthode `linkTo` qui prend en paramètre la classe du contrôleur et la méthode à laquelle le lien doit pointer.
- La méthode `withSelfRel` permet de spécifier le type de lien, ici un lien vers la ressource elle-même.
- Le lien hypertexte est ajouté à la ressource avec la méthode `add`.

Conclusion

- Spring HATEOAS est un projet qui facilite la création de services RESTful qui suivent le principe HATEOAS.
- Il fournit des classes utilitaires pour créer des liens hypertextes dans les réponses des services RESTful.
- Cela permet de rendre les services plus découvrables et plus facilement utilisables.