Rapport

Spring Boot

Introduction

Pour le travail demandé, il fallait choisir un type de données à représenter pour notre application Spring Boot, j'ai donc choisi d'utiliser des corps célestes car l'espace est passionnant. J'ai nommé mon projet **FakeUniverse** car il est question de créer et stocker ses propres corps plutôt que celles existantes.

J'ai choisi l'appellation « corps céleste » plutôt que « planète » pour donner la capacité de créer des étoiles en plus des planètes, car une étoile n'est pas une planète et vice versa, tandis qu'objet céleste représente les deux.

Structure

Pour commencer, voici mon objet définissant un objet céleste :

```
private String name;
private double radius;
private double distanceFromOrigin;
private double gravity;
```

Un corps céleste est donc représenté de manière unique par son nom (car il n'y a pas deux corps célestes portant le même nom dans notre univers, j'ai choisi de maintenir cette règle ici) ainsi que son rayon, sa distance à l'origine et sa gravité. J'ai choisi de ne pas inclure plus d'informations car ça ne rendrait pas plus compliqué l'utilisation de l'application.

A noter que ce que j'appelle l'origine est l'interprétation de la coordonnée (0, 0) dans l'univers fictif de l'utilisateur (généralement représenté par le Soleil dans notre univers, même si cela n'a aucun sens étant donné que nous sommes constamment en mouvement).

Pour le contrôleur, que je comprends comme étant la liaison entre les messages reçus et l'application, il est sous la forme :

```java
@RestController
public class PlanetController {

    @Autowired
    private IPlanetDAO dao;

    @GetMapping(value="/Bodies")
    public List<Planet> getBodies() { return dao.findAll(); }

    @GetMapping(value="/Bodies/{name}")
    public Planet getBody(@PathVariable String name) { return dao.findByName(name); }

    @PostMapping(value = "Bodies")
    public void addBody(@RequestBody Planet body) { dao.save(body); }

    @DeleteMapping(value="/Bodies/{name}")
    public void removePlanet(@PathVariable String name) { dao.remove(name); }

}
```

Passons en revue ses fonctions de manière simple :

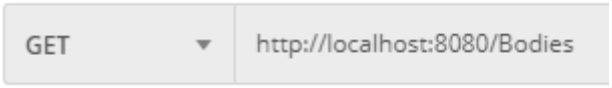
- *getBodies()* permet de récupérer, tous les corps de l'univers fictif.
- *getBody(String)* permet de récupérer un corps spécifique identifié par son nom.
- *addBody(Planet)* permet d'ajouter un corps au stockage local.
- *removePlanet(String)* permet de supprimer un corps du stockage, identifié par son nom.

Exemple d'exécution

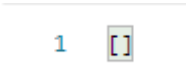
Pour finir, je vais utiliser ces méthodes grâce à Postman, un logiciel permettant l'envoi de requêtes.

Toutes les requêtes suivantes se feront sur la même instance de l'application.

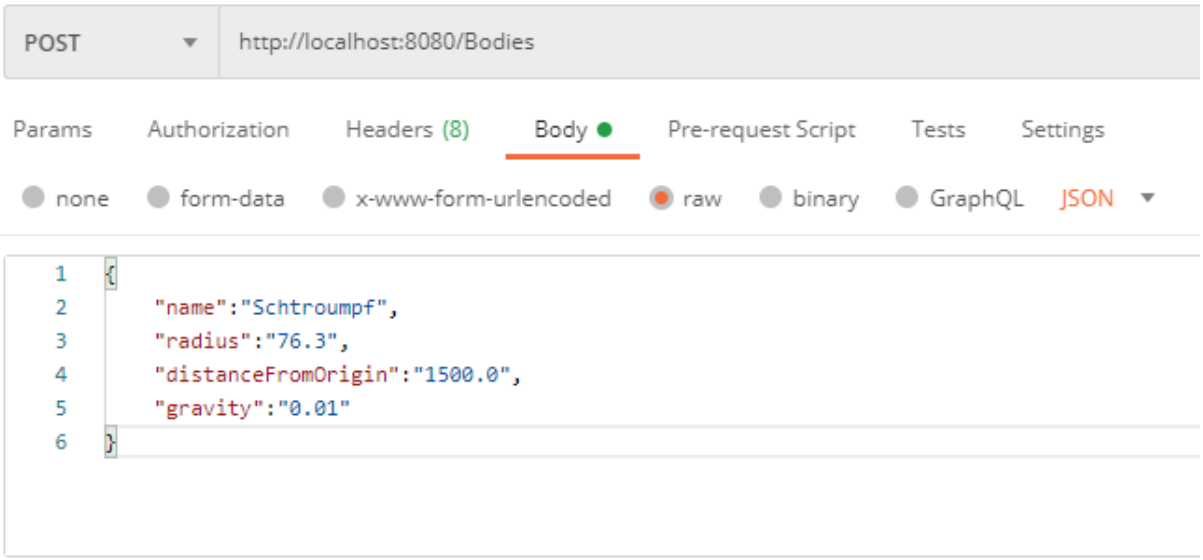
Je commence par lancer l'application et demander la liste des corps via Postman :



Et voici la réponse :

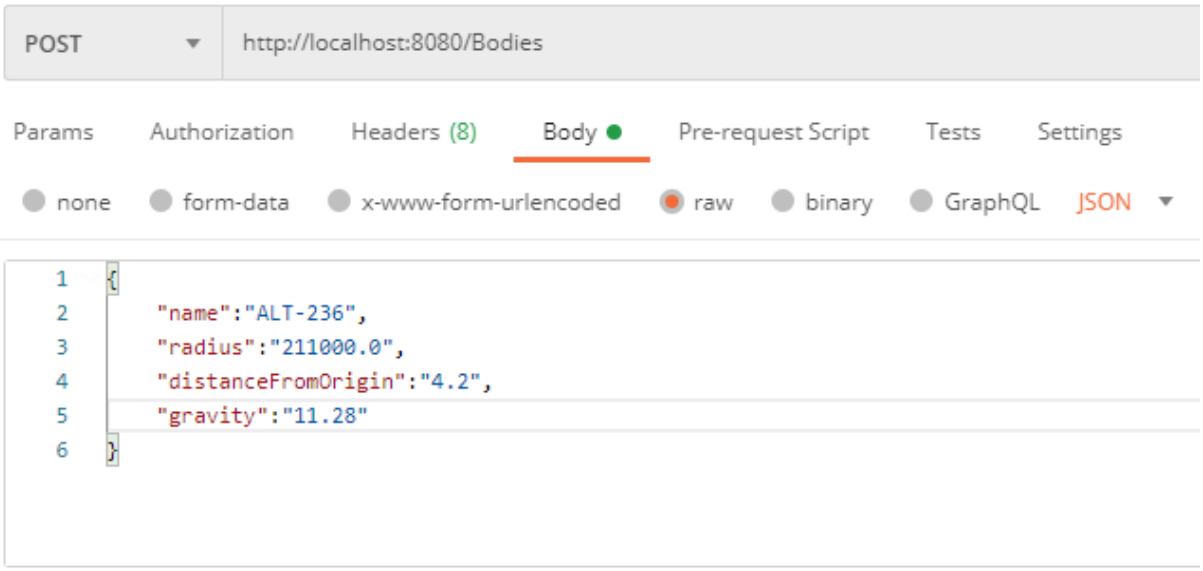


La réponse contient une collection vide étant donné que nous n'avons pas encore ajouté de corps céleste, je procède donc à l'ajout de la planète Schtroumpf :

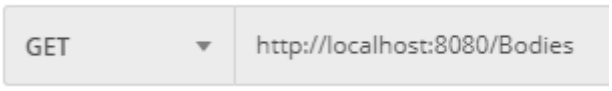


J'envoi donc une requête d'ajout de corps avec la méthode POST, en précisant dans le Body de la requête la description, au format JSON, de la planète.

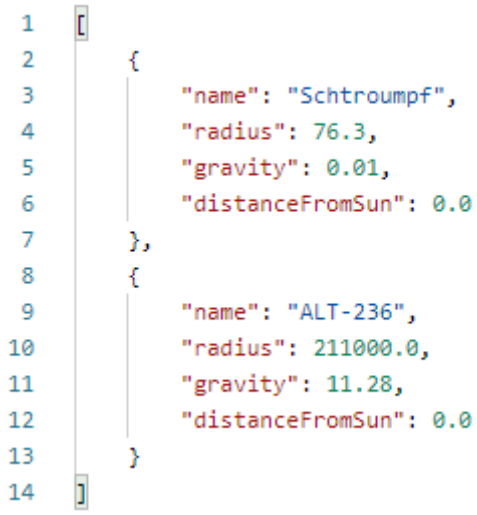
Je procède à l'ajout d'un second corps céleste nommé cette fois ALT-236 (les astrologues raffolent des noms compliqués et mathématiques) :



Je vais désormais, pour vérifier que l'ajout a bien eu lieu, demandé la liste de tous les corps célestes en passant par la même requête que précédemment :

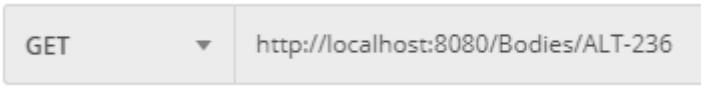


Cette fois-ci, la réponse n'est plus vide :



Elle contient bien les deux corps ajoutées.

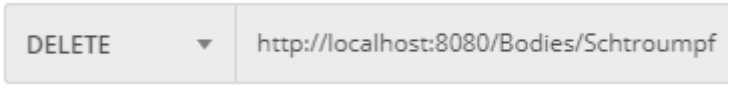
Pour l'affichage d'une seule planète, il suffit d'ajouter le nom du corps voulu à la requête de cette façon :



Et on obtient cette réponse :

```
{
    "name": "ALT-236",
    "radius": 211000.0,
    "gravity": 11.28,
    "distanceFromSun": 0.0
}
```

Pour finir, je vais supprimer la planète Schtroumpf (Gargamel aurait-il enfin réussi à créer la pierre philosophale ?) avec la requête suivante :



Pour vérifier la suppression, j'effectue à nouveau une requête pour afficher la liste des corps céleste, et j'obtiens la réponse suivante :

```
[
    {
        "name": "ALT-236",
        "radius": 211000.0,
        "gravity": 11.28,
        "distanceFromSun": 0.0
    }
]
```

J'en déduis que la suppression a bien eu lieu.

Notes de fin

Voilà pour la présentation du fonctionnement de l'application, pour finir je souhaite indiquer que j'ai suivi le cours en détail pour la fondation de mon application et tout (ainsi que la suppression) s'est déroulé sans problèmes majeurs. Merci donc pour ce cours et j'espère que votre lecture fut agréable.