Compte Rendu TP5

CSR

Roberto Henry Chacon Suarez
Manh-Huan Nguyen

Groupe M1-Miage 2D

Problématiques rencontrées

Problématiques Partie 1:

- L’une des problématiques a été le fait que les voyageurs ne montaient que sur les trains de la même voie, c’est-à-dire, si nous avions que 100 voyageurs qui arrivaient à la gare, et deux voies dispo, ils monteraient seulement sur la première voie, même s’il avait des trains dans les deux. Ce problème a été résolu en choisissant un nombre aléatoire entre 1 et le nombre de voies ayant un train, et assignant le ticket à cette voie. Comme ça chaque client aurait des tickets différents.

- Deuxième problématique, nous avons eu un interblocage quand à un certain moment, un voyageur essayait de monter sur une voie, mais que ce train venait juste de partir, donc il pouvait pas monter et nous avions un null pointer exception. Pour corriger cela, nous avons enlevé la possibilité d' assigner une voie à un voyageur, mais lui laisser plutôt monter à n’importe quel train tant qu’il avait de places.

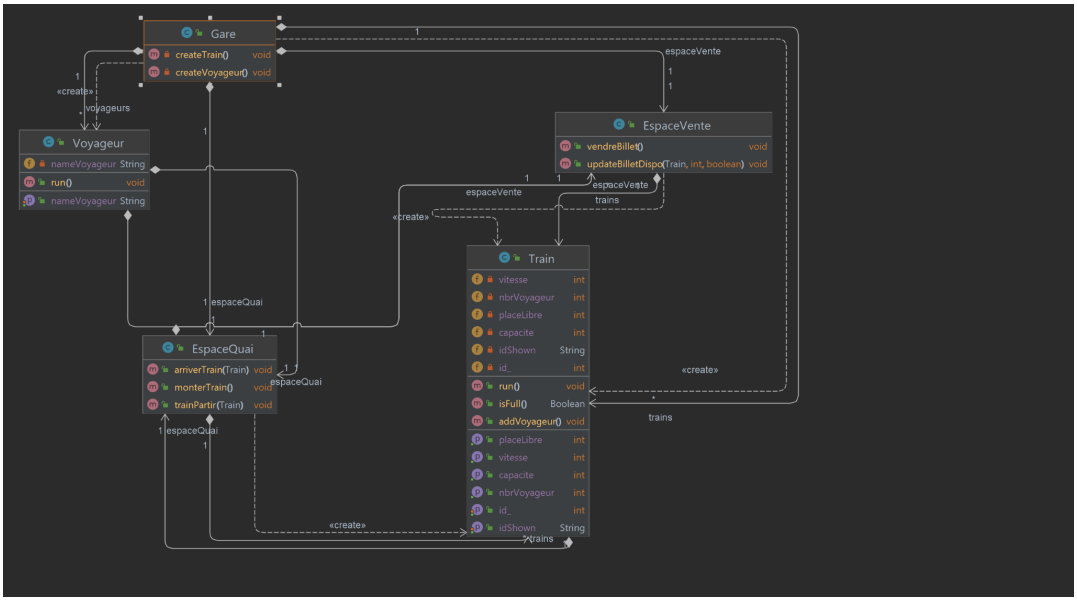
- Une autre problématique a été trouvée quand on augmentait le nombre de voyageurs, par exemple à 1000. Au moment donné, les clients arrêtent de monter dans le train en attendant des tickets disponibles, et il y avait juste les trains qui arrivaient à la gare. À l’aide des logs, nous nous sommes rendu compte qu’il y avait une erreur lors de la mise à jour des tickets disponibles du côté de l’espace vente, et que les voyageurs ne peuvent pas acheter un billet pour un train. Cela a été réglé en mettant en place une condition pour laisser les tickets disponibles de la gare à 0 au lieu de rester le nombre de places disponibles dans le train (ce qui parfois causait un nombre négatif et malgré l'arrivée des trains, le nombre restait toujours négatif).

- Enfin la dernière problématique rencontrée a été aussi liée au fait d’augmenter le nombre de voyageurs à 1000. Au moment donné, il y avait des voyageurs qui attendaient des tickets alors que tous les trains étaient déjà partis. Pour résoudre ce problème, nous avons simplement augmenté la valeur d’attente des trains ainsi que le nombre de trains arrivant en gare, de façon qu’il y aurait toujours un nombre suffisant de trains pour tous les voyageurs.

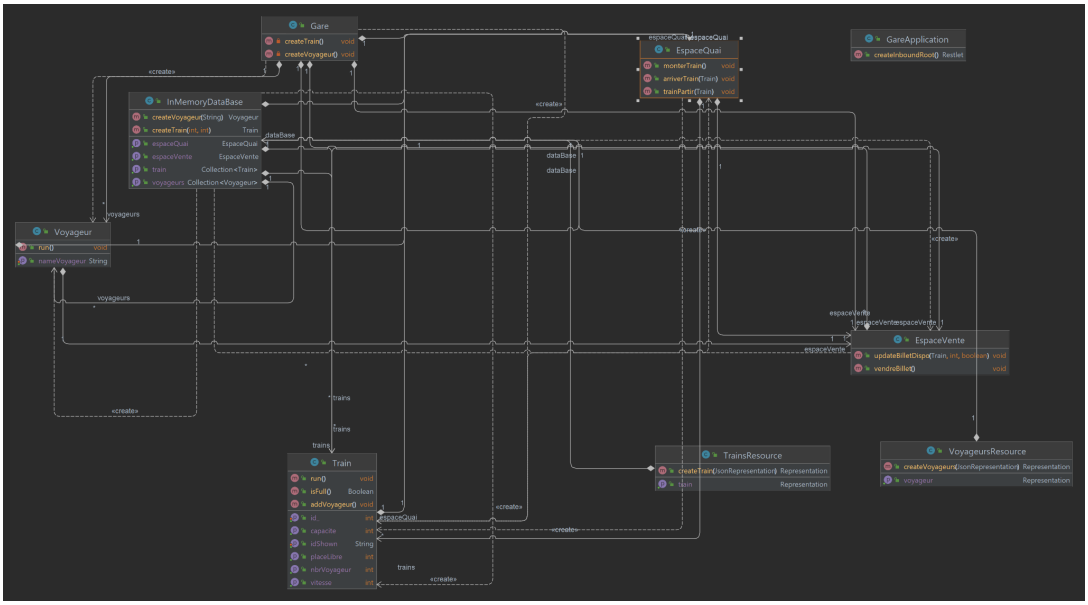
Contrairement à la partie 1, **la partie 2** en général n’a pas posé des problèmes significatifs qui méritent d’être soulignés.

Diagramme des classes

V1:



V2:



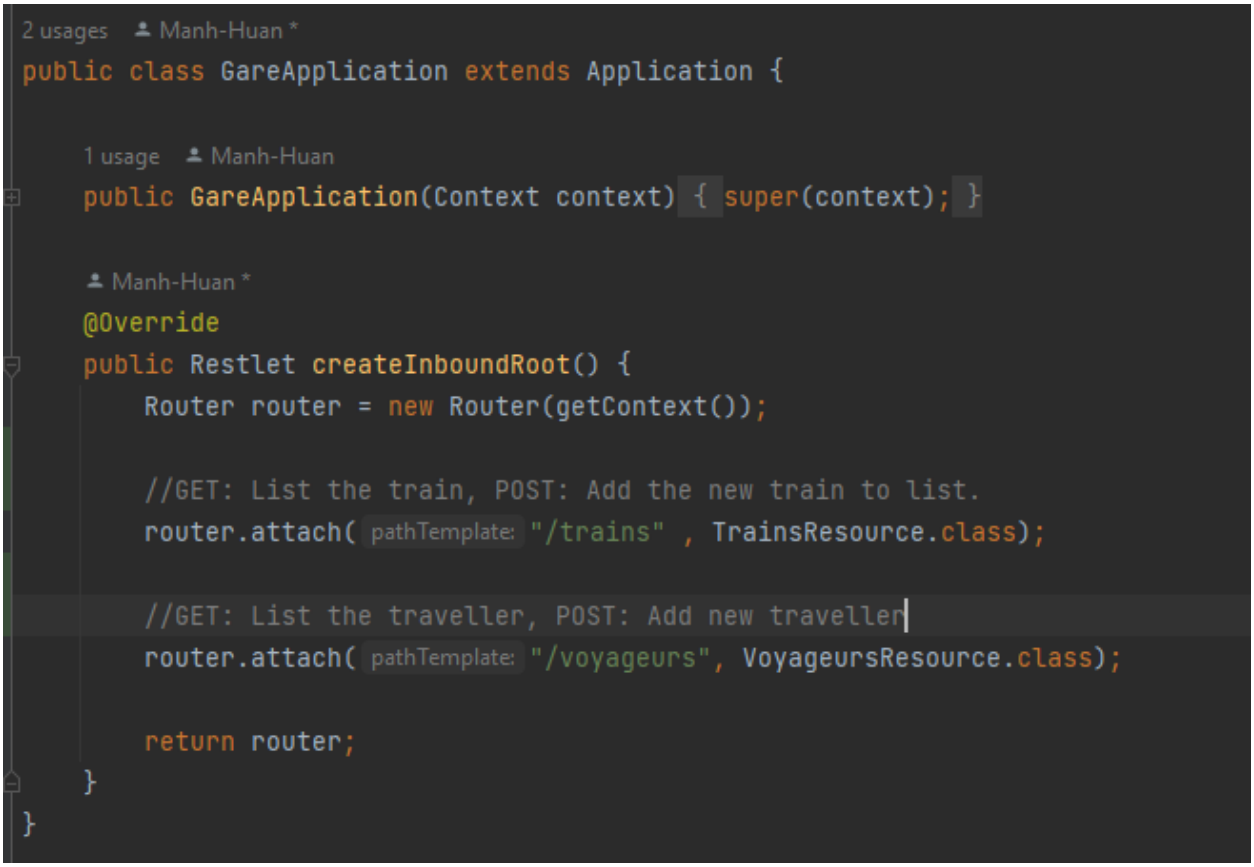
Threads:

- *Voyageur*
- *Train*

Objets partagés:

- *Espace Vente*
- *Espace Quai*

L’application



On y remarque 2 url différentes :

- http://localhost:8124/voyageurs :
    - GET permet de renvoyer tous les voyageurs.
    - POST permet d’en ajouter un.

- http://localhost:8124/trains :
    - GET permet de renvoyer tous les trains avec leurs informations.
    - POST permet d’en ajouter un.

Simulation

Nous avons construit des requêtes afin de tester les différentes fonctionnalités, elles se trouvent dans test Commands.sh, ce sont celles ci-dessous :

```
printf "\n*** INIT ***\n"

curl http://localhost:8124/trains
curl http://localhost:8124/voyageurs

printf "\n*** ADD TRAINS ***\n"

curl -X POST -H 'Content-type:application/json' -H 'Accept:application/json'
-d '{"capacite":100,"vitesse":250}' http://localhost:8124/trains
curl -X POST -H 'Content-type:application/json' -H 'Accept:application/json'
-d '{"capacite":80,"vitesse":200}' http://localhost:8124/trains


printf "\n*** GET TRAINS ***\n"

curl http://localhost:8124/trains

printf "\n*** ADD VOYAGEURS ***\n"

curl -X POST -H 'Content-type:application/json' -H 'Accept:application/json'
-d '{"name":VoyageurA}' http://localhost:8124/voyageurs
curl -X POST -H 'Content-type:application/json' -H 'Accept:application/json'
-d '{"name":VoyageurB}' http://localhost:8124/voyageurs

printf "\n*** GET VOYAGEURS ***\n"

curl http://localhost:8124/voyageurs
```

Afin de tester les commandes, il suffit de lancer le main du package V2, et soit les lancer entre la simulation des trains prédéfinis ou à la fin (le programme reste en cours afin de pouvoir tester les commandes).

1. **Exemple d'une exécution à la fin de la simulation des trains/voyageurs prédéfinis :**

- Ajout d'un **voyageur** de nom *VoyageurA*:

```
rchacons@roberto-ThinkPad:~$ curl -X POST -H 'Content-type:application/json' -H
'Accept:application/json' -d '{"name":VoyageurA}' http://localhost:8124/voyageur
s
{"name":"VoyageurA"}
```

Son état dans la simulation est A car il n'est pas encore muni d'un ticket (Il n'y a plus de trains):

```
L'état du voyageur : VoyageurA/Thread-117 est : A - en route vers la gare
2022-12-09  23:11:11    127.0.0.1    -    -    8124    POST    /voyageurs  -    200 -    18  3    http://localhost:8124    curl/7.74.0 -
```

- Ajout d'un **train** (avec un id=4 assigné automatiquement) :

```
rchacons@roberto-ThinkPad:~$ curl -X POST -H 'Content-type:application/json' -H
'Accept:application/json' -d '{"capacite":100,"vitesse":250}' http://localhost:8
124/trains
{"placeDispo":100,"capacite":100,"id":4,"vitesse":250}
```

Nous constatons dans la simulation les différents états du train une fois il est ajouté, et comment le VoyageurA ajouté précédemment peut aussi continuer son processus:

```
L'état du train : 4/Thread-119 est :A - en route vers la gare
2022-12-09  23:16:10    127.0.0.1    -    -    8124    POST    /trains -    200 -    30  3    http://localhost:8124    curl/7.74.0 -
L'état du train : 4/Thread-119 est :C - en gare
L'état du voyageur : VoyageurA/Thread-116 est : B - muni d'un ticket
L'état du voyageur : VoyageurA/Thread-116 est : C - monté dans un train
L'état du train : 4/Thread-119 est : D - parti
```

2. Exemple d'une exécution pendant la simulation des trains/voyageurs prédéfinis :

- Pendant la simulation, nous lançons le testCommands.sh:

```
rchacons@roberto-ThinkPad:~/Documents/Cours/S7/CSR/TP/CSR/Projet$ /bin/bash /home/rchacons/Documents/Cours/S7/CSR/TP/CSR/Projet/testCommands.sh

*** INIT ***
[{"capacite":99,"placeLibre":99,"NbrDeVoyageur":0,"id":0,"vitesse":200}][]
*** ADD TRAINS ***
{"placeDispo":100,"capacite":100,"id":1,"vitesse":250}{"placeDispo":80,"capacite":80,"id":2,"vitesse":200}
*** GET TRAINS ***
[{"capacite":99,"placeLibre":99,"NbrDeVoyageur":0,"id":0,"vitesse":200},{"capacite":100,"placeLibre":100,"NbrDeVoyageur":0,"id":1,"vitesse":250},{"capacite":80,"pla
ceLibre":80,"NbrDeVoyageur":0,"id":2,"vitesse":200}]
*** ADD VOYAGEURS ***
{"name":"VoyageurA"}{"name":"VoyageurB"}
*** GET VOYAGEURS ***
[{"name: ":"VoyageurA"},{"name: ":"VoyageurB"}]
```

Dans la simulation nous remarquons nos voyageurs ajoutés et en cours d'exécution avec d'autres processus (les voyageurs prédéfinis n'ont pas de nom) :

```
 L'état du voyageur : VoyageurA/Thread-113 est : A - en route vers la gare
 L'état du voyageur : name/Thread-61 est : B - muni d'un ticket
 L'état du voyageur : name/Thread-61 est : C - monté dans un train
 L'état du voyageur : VoyageurB/Thread-114 est : A - en route vers la gare
 2022-12-09  23:23:29    127.0.0.1    -    -    8124    POST    /voyageurs  -    200 -    18  2    http://localhost:8124    curl/7.74.0 -
 2022-12-09  23:23:29    127.0.0.1    -    -    8124    POST    /voyageurs  -    200 -    18  1    http://localhost:8124    curl/7.74.0 -
```

```
L'état du voyageur : name/Thread-63 est : A - en route vers la gare
L'état du voyageur : VoyageurA/Thread-113 est : B - muni d'un ticket
L'état du voyageur : VoyageurA/Thread-113 est : C - monté dans un train
```

```
L'état du voyageur : name/Thread-67 est : A - en route vers la gare
L'état du voyageur : VoyageurB/Thread-114 est : B - muni d'un ticket
L'état du voyageur : VoyageurB/Thread-114 est : C - monté dans un train
L'état du voyageur : name/Thread-68 est : A - en route vers la gare
```

Idem pour les trains (ils ont des id alors que les prédéfinis n'en ont pas):

```
L'état du train : 1/Thread-110 est :A - en route vers la gare
L'état du train : 2/Thread-112 est :A - en route vers la gare
2022-12-09  23:23:29    127.0.0.1   -   -   8124    POST    /trains -   200 -   30  7   http://localhost:8124   curl/7.74.0 -
2022-12-09  23:23:29    127.0.0.1   -   -   8124    POST    /trains -   200 -   29  2   http://localhost:8124   curl/7.74.0 -
```

```
L'état du voyageur : name/Thread-77 est : C - monté dans un train
L'état du train : 2/Thread-112 est :C - en gare
L'état du train : 1/Thread-110 est : B - en attente d'une voie libre
L'état du voyageur : name/Thread-83 est : A - en route vers la gare
```

```
L'état du voyageur : name/Thread-93 est : B - muni d'un ticket
L'état du train : 2/Thread-112 est : D - parti
L'état du voyageur : name/Thread-93 est : C - monté dans un train
L'état du voyageur : name/Thread-97 est : A - en route vers la gare
L'état du voyageur : name/Thread-92 est : B - muni d'un ticket
L'état du voyageur : name/Thread-92 est : C - monté dans un train
L'état du voyageur : name/Thread-98 est : A - en route vers la gare
L'état du voyageur : name/Thread-97 est : B - muni d'un ticket
L'état du train : 1/Thread-110 est :C - en gare
```

```
L'état du voyageur : name/Thread-105 est : B - muni d'un ticket
L'état du voyageur : name/Thread-105 est : C - monté dans un train
L'état du voyageur : name/Thread-104 est : B - muni d'un ticket
L'état du train : 1/Thread-110 est : D - parti
```