Théo Descoustey
Vincent Samson
groupe 5

Rapport du projet Programmation Culturelle de la ville de JolieCité

info : java 11

Nous avons créé 3 couches : domain , infra et application:
La couche domain contient les concepts métiers.
La couche infra est une infrastructure qui permet de sauvegarder les programmations.
La couche Application qui permet d'appliquer notre projet.

Dans la couche domain , nous avons 6 classes dont une abstraite, et une interface.

- Event : un événement. (Value Object)
  * name : nom du groupe/piece de theatre
  * capacity : capacité de la salle
  * dates : liste de dates (minimum 1 date)
  * duration : durée de l'événement (heure)

Cette classe est abstraite et est utilisée aux classes Concert et Theatre.
Nous utilisons une liste de dates pour que cela soit simple , on peut utiliser une liste de dates correspondant au créneau pour la classe Théâtre et une liste de 1 date pour la classe Concert. Cela permet de faciliter le parcours des dates dans city.

donc

- Concert : un concert (Value Object)
  * dates : liste d'une seule date
  * group_name : Nom du groupe
  * capacity : capacité pour cet événement
  * duration : durée du concert

et

- Théâtre (Value Object)
  * dates : liste de dates pour cette pièce de théâtre
  * name : nom de la pièce
  * capacity : capacité pour cet événement
  * duration : durée de la pièce

CityRepository est le repository pour City, c'est une interface .Il est dans la couche Domain pour pouvoir accéder à domain quand on créera une RepositoryInMemory

la classe Hall est la salle qui est défini

- Hall : salle (Entity)
  * capacity : la capacité de la salle
  * date: Hashmap qui lie une date avec la durée d'ouverture de la salle lors de cette date.

Il y a aussi une ID de type UUID pour identifier la salle et une liste d'événements qui se déroule dans la salle et qui sera remplie dans l'aggregate .

La classe HallDisponibilty permet de savoir la disponibilité des salles :

- HallDisponibilty (Entity)

il y a aussi une map hallsDisponibility. Cette map lie le numéro de la semaine (1-52) avec une autre map qui lie elle-même de numéro du jour (1-7) avec la liste de salles disponibles ce jour-là.
On génère toutes les hashmaps et les listes dans le constructeur.

La classe City (aggregate)
est composé de l'ID de l'aggregate , que l'on sauvegardera et qu'on pourra la retrouver en faisant load dans repositoryInMemory.

Nous avons créé des dates pour 3 salles pour le mois de janvier 2021. ou la salle 1 a une capacité max de 300 personnes , la salle 2, 500 personnes et 1000 personnes pour la salle 3

attributeEvent vérifie si on peut attribuer un événement à une date proposée en se servant l'objet hallDisponibility.

RepositoryInMemory est dans la couche infra qui sauvegardera l'id et les éléments dans la classe City

et on peut retrouver les éléments d'une salle grâce à l'id dans City.

La classe MyAppli est dans la couche application.
On démarre l'application par un choix , 1 si on veut utiliser le jeu d'essai d'événement que nous avons attribué ou 2 si on veut attribuer nous même un

événement.S’il y a une erreur de date ou autre motif (déjà un événement dans une date, dépasse la capacité …) , on met que cela n’est pas possible.