Parasites, le parser d'annotations

L'objectif des annotations en Java

Les annotations en Java permettent aux développeurs de nombreuses choses, mais toujours dans un objectif de plus grande claret du code ou encore de factorisation d'actions.

Les annotations peuvent être manipulée au moment de la compilation ou lors du « Runtime » .

Les annotations dans le jeu Parasites

Pour notre jeu, nous avons décidé d'utiliser les annotations pour mutualiser du code dans l'héritage de nos pièces.

En effet, nous avons une classe abstraite Parasite dont héritent 5 autres types de pièces.

L'idée, dans l'utilisation des annotations, est de mutualisé du code, sans avoir besoin de faire X méthodes abstraites au sein des classes.

Nous avons donc décidé de créer les annotations suivantes :

```
@Representation(« image.png »)
@Characteristics(cost = 3, attack = 5, …)
@SoundEffect("sound.mp3")
```

Annotation @Representation :

Cette annotation nous permet de préciser facilement quelle sera l'icône représentant le Parasite.

Annotation @Characteristics :

Cette annotation nous permet de mettre plus clairement les caractéristiques d'un parasite que si nous devions les placer à la suite dans le constructeur du Super.

Annotation @SoundEffect :

Cette annotation nous permet de préciser un son qui sera joué lors de la création d'un parasite, au cas où nous voudrions que chaque parasite ait un son différent.

Les annotations pour l'application Java FX :

Dans le cadre du jeu, une application JavaFX est créée et utilisée comme plateforme multijoueur.

Afin de mutualiser notre code JavaFX, qui peut parfois être verbeux, nous avons décidé de mettre en place 2 types d'annotations :

@PressEnter

@ColumnTarget

- Lorsque l'utilisateur renseigne un champ dans une application Desktop, il souhaite souvent pouvoir simplement appuyer sur la touche « entrée » de son ordinateur pour valider son texte, et non cliquer avec la sourie sur le bouton prévu à cet effet.
  L'annotation @PressEnter permet au développeur de renseigner, pour un champ quelconque, le nom de la méthode à appeler lorsque l'utilisateur se trouve dans ce champ et qu'il tape sur « entrée ».
  Ainsi, cette annotation a été placée à ce jour pour les champs de connexion ainsi que pour l'envoie d'un message dans le chat.

- Pour notre application JavaFX, nous utilisons notamment un tableau afin de renseigner l'état du serveur et plus précisément des parties en cours, en attente etc…
  Les tableaux JavaFX sont constitués de colonnes à qui nous devons préciser l'objet métier concerné, ainsi que le champ de cet objet.
  Par exemple, dans notre cas, le tableau représentant les parties actuelles gèrera un objet Game et les différentes colonnes porteront soit le nom du créateur, soit le nombre de joueurs actuels, etc…
  Signifier le champ concerné par une colonne est très verbeux en JavaFX et c'est un travail qui peut se mutualiser très efficacement grâce aux annotations !
  En effet, notre annotation @ColumnTarget renseigne immédiatement le champ de l'objet métier concerné et économise de nombreuses lignes de codes.