Consignes pour le projet court (PR02)

19 février 2023

0.0.1 Sujet

Le projet court [1] doit être fait à partir des éléments fournis dans votre projet Git (dossier 'projet-court') et rendu via ce même projet Git sur Gitlab.

Il ne faut pas déplacer les fichiers fournis !

0.0.2 Jalons

Voir le sujet...

0.0.3 Tests en boîte noire

Pour que vous sachiez comment votre programme sera testé, nous vous fournissons quelques tests ainsi qu'un script qui permet de les lancer (testeur.sh). Ils sont sur votre projet Git dans le dossier tests/.

Chaque test est défini par deux fichiers. Le premier a l'extension .run et est le test à lancer (par exemple en faisant : sh testeur.sh leTest.run). Le second a l'extension .expected. Il contient le résultat attendu, ce que le programme doit afficher dans le terminal.

Le fichier testeur.sh est le testeur. Il prend en argument les tests à exécuter (fichier avec l'extension .run).

```
# Lancer le seul test exempleTricheurSujet.run
sh testeur.sh tests/exempleTricheurSujet.run

# Lancer tous les tests du dossier courant
sh testeur.sh tests/*.run
```

Il lance chaque test (fichier .run) et enregistre son résultat avec l'extension .computed. La commande diff permet de calculer la différence entre les deux fichiers (.expected et .computed). Les options -Bbw de diff sont utilisées pour ignorer les lignes vides et les blancs. Le verdict est enfin affiché, 'ok' ou 'ERREUR'. Dans le cas d'une erreur, le contenu du fichier diff est affiché.

Les tests peuvent être lancés depuis le dossier 'projet-court' ou le dossier 'projet-court/src'.

1. to-1sn-2021-pr-02-sujet.pdf

Attention à ne pas ajouter les fichiers .diff ou .computed sur Git car ce sont des fichiers engendrés. On peut ajourter les lignes suivantes dans le fichier .gitignore (à la racine de votre projet) :

```
*.diff
*.computed
```

0.0.4 Réponses à vos questions

Recommencer une partie Doit on pouvoir recommencer une partie une fois l'actuelle terminée ? (avec alternance du départ entre joueur 1 et joueur 2 ?)

Non. Une seule partie.

NoSuchElementException. j'ai aujourd'hui effectué les tests boites noires que vous nous avez fournis.

Je n'ai qu'une seule erreur que je n'arrive pas à comprendre : il semblerait que dans exemplePresqueSujet, vous testiez avec les entrées suivantes : 2 - X - 5 - 3 - 1.

Sous Eclipse, comme sur un terminal, je peux tout à fait lancer une partie avec Toto@humain et Titi@rapide puis saisir ces 5 arguments au fur et à mesure sans avoir d'erreur et ainsi finir la partie. Cependant, avec la commande sh testeur.sh tests/*.run, l'exemplePresqueSujet me lève une erreur "NoSuchElementException" dans le Scanner de ma classe Humain. Il semblerait que ce soit à la saisie du deuxème caractère (X).

Il semblerait également pour en avoir discuté avec mes camarades que je ne sois pas le seul à avoir cette exception. Ne peut-elle pas venir du test ?

NoSuchElementException se produit quand il n'y a plus de données en entrée. Vous pouvez le reproduire en tapant un CTRL-D au clavier (qui ferme l'entrée standard).

Le problème vient de la manière dont vous utilisez Scanner. Vous créez un nouveau Scanner à chaque getPrise(). Le Scanner précédent a consommé les caractères (lecture en avant). Ils ne sont donc plus disponibles pour les lectures suivantes.

Vous pouvez le vérifier en ajoutant, juste avant la création du Scanner et juste après une saisie :

```
try {
    System.out.println(" available ␣:␣" + System.in.available());
} catch (java.io.IOException e) {
    System.out.println(e);
}
```

qui affiche le nombre de caractères disponibles sur l'entrée standard.

Vous verrez que s'il y a 10 caractères disponibles avant la création du Scanner, dès qu'une lecture est faite, il n'y a plus de caractères disponibles sur l'entrée standard. D'où le NoSuchElementException.

Ce qu'il faut faire, c'est ne créer qu'un seul Scanner pour toute l'application.

Tests fournis. Que fait l'utilisateur ? Vous nous avez fourni un fichier contenant des exemples de tests.

Lors de l'exemple "exemplePresqueSujet.expected", ligne 12, je ne suis pas sûre de ce que fait l'utilisateur. Est-ce bien un " " ?

Pour savoir ce que fait l'utilisateur il faut regarder le fichier en .run. Ce que fournit l'utilisateur n'apparaît pas sur la sortie (et donc pas non plus dans .expected) car ce n'est pas saisi au clavier (donc pas d'écho dans le terminal) mais récupéré grâce à la redirection de l'entrée standard avec '«EOF'. Ce qui est fourni au programme est donc entre les deux EOF dans le .run. L'utilisateur rentre donc successivement :

```
2
X
5
3
1
```

Problème de compilation. J'ai voulu compiler (juste compiler) la classe Jouer sans rien modifier et elle ne compile pas car la classe ConfigurationException est introuvable alors qu'elle est dans le même paquetage.

Il faut être dans le dossier qui contient le dossier "allumettes" (le plus simple)... Ou définir le CLASSPATH ou utiliser l'option '-cp' de javac et java.

```
cd ...../projet-court/src
javac allumettes/Jouer.java
java allumettes.Jouer
```

Utiliser Eclipse est certainement une bonne idée ! Il est important de savoir utiliser ce type d'outil qui peut, par bien des aspects, vous faciliter la gestion du code Java (ou d'autres langages).

Mon programme boucle avec le message "vous devez donner un entier" avec exemplePresqueSujet J'ai un problème dans le projet court, c'est que j'ai exécuté les tests fournis et ils marchent très bien mais quand j´exécute le test avec exemplePresqueSujet je trouve dans le fichier.computed une boucle infinie avec le message "vous devez donner un entier" après la saisie de 'X' alors que dans eclipse ça marche très bien avec les mêmes entrées.

Lors d'une saisie incorrecte, il faut consommer les caractères qui posent problèmes, ici "X". On peut utiliser la méthode nextLine() de Scanner pour le faire.