UNIVERSITÉ D'AVIGNON ET DES PAYS DE VAUCLUSE

L3 IL – 2018/19
UE Projet de programmation

Vincent Labatut & Nejat Arinik

Projet | Jeu Pingu Party

Le but du projet de cette année est de produire un logiciel en Java permettant de jouer à un jeu de carte appelé *Pingu Party*. Le logiciel devra permettre à un utilisateur humain d'affronter d'autres joueurs : soit humains via une connexion réseau, soit artificiels via un agent (relativement) intelligent. Le logiciel devra comporter une interface graphique et gérer des statistiques sur les parties jouées. Ce document donne une description générale du projet, qui sera approfondie au cours de l'année.

1 Pingu Party

Pingu Party est un jeu de Reiner Knizia datant de 2008 [1]. Attention, il en existe de nombreuses variantes : dans le cadre du projet, on se concentre sur les règles décrites ici.

Ce jeu se base sur un ensemble de cartes caractérisées uniquement par leur couleur. Il y a 5 couleurs différentes. L'ensemble des cartes comprend 7 cartes rouges, violettes, bleues et jaunes, et 8 cartes vertes, pour un total de 36 cartes.

Distribution. Les joueurs sont ordonnés de façon aléatoire, et cet ordre *relatif* sera ensuite utilisé pour déterminer les tours de jeu jusqu'à la fin de la partie. Supposons par exemple qu'on ait 4 joueurs A, B, C, D, et qu'on tire au sort l'ordre A, C, D, B. Alors c jouera toujours après A et avant C. Par contre, le joueur qui débute une manche peut changer d'une manche à l'autre (voir plus loin).

Les cartes sont mélangées. Chaque joueur reçoit exactement le même nombre de cartes, selon les modalités suivantes :

- 2 joueurs : 14 cartes chacun, les cartes restantes sont inutilisées et cachée aux joueurs.
- 3 joueurs : 12 cartes chacun.
- 4 joueurs : 9 cartes chacun.
- 5 joueurs : 7 cartes chacun, et la carte restantes est montrée aux joueurs.
- 6 joueurs : 6 cartes chacun.

Manche. Une partie se compose de plusieurs manches. Chaque joueur pose une carte à son tour, selon les règles décrites ci-après. S'il ne peut pas jouer, il est éliminé de la manche. La manche s'arrête quand plus aucun participant ne peut jouer. Le score d'un joueur pour la manche correspond au nombre de cartes qui lui restent en main à la fin de cette manche.

La manche suivante commencera avec le joueur qui suit (dans l'ordre établi en début de partie) le dernier participant à avoir joué. Donc, pour reprendre l'exemple précédent : si c'est C qui a joué en dernière lors de la première manche, alors ce sera D qui commencera la deuxième.

Le but de la manche est de construire un triangle avec les cartes, à la manière de la Figure 1. Celui-ci peut comporter un 1er étage de 8 cartes, puis un 2ème de 7 cartes, et ainsi de suite jusqu'au 8ème qui ne contient qu'1 seule carte. Pour les parties à 2 joueurs, on part directement de l'étage à 7 cartes (il n'y a donc pas d'étage à 8 cartes).

Quand un participant doit jouer, trois possibilités s'offrent à lui :

- Poser une carte au 1er étage : aucune contrainte.
- Poser une carte à un étage supérieur au 1er : elle doit reposer sur deux cartes de l'étage directement inférieur, et telles qu'au moins une des deux est de la même couleur que la carte à poser.
- Ne rien faire : le joueur est éliminé de la manche.

Notez que le joueur ne peut poser une carte à un étage que s'il y reste au moins une place libre.

Figure 1. Exemple de triangle créé lors d'une partie de Pingu Party.

Fin. On joue autant de manches qu'il y a de joueurs. Le score total est obtenu en sommant les scores des manches. Le classement général est obtenu en ordonnant les joueurs par score total décroissant, le gagnant étant celui qui a le plus petit score à la fin de la partie. En cas d'égalité, on considère le nombre de manches gagnées. Le gagnant d'une manche est le joueur qui a posé une carte lors du dernier tour de jeu de cette manche : il peut donc y en avoir plusieurs. S'il y a toujours égalité dans le classement général, on considère celui qui a posé le moins de cartes *vertes* au cours de la partie. S'il y a toujours égalité, alors on laisse le classement tel quel et plusieurs joueurs partagent la même place.

2 Organisation du projet

Constitution des groupes. Le travail sera réalisé en groupes de 4 personnes. Vous devez utiliser le lien prévu à cet effet sur e-uapv pour constituer les groupes. Les groupes contenant moins de 4 personnes courent le risque de se voir imposer des membres. Une fois fixés, ces groupes ne pourront plus changer en cours de semestre (du moins de votre fait). Par contre, ils pourront évoluer entre les deux semestres, si certains étudiants désirent échanger leurs places.

Premier semestre. L'objectif du S1 est la mise en place des outils de production, et l'élaboration d'une version incomplète mais fonctionnelle du jeu. Plusieurs tâches relatives à LaTeX et Git seront demandées, dont certaines seront personnelles et d'autres à faire en groupe.

Pour ce qui est du jeu proprement dit, il y a 4 tâches à effectuer au cours du S1, qui seront détaillées en séance :

- Structure de données, moteur du jeu, et agent minimal, le tout testable en mode console.
- Interface graphique minimale, affichage des statistiques basiques sous forme de tableaux.
- Mode réseau client/serveur.
- Gestion de statistiques basiques, profils basiques pour les joueurs

Second semestre. Le but du S2 est de terminer l'implémentation du jeu, de réaliser un agent capable de jouer convenablement, et d'uniformiser le protocole de communication afin que les agents puissent s'affronter lors d'un tournoi à la fin du semestre. Là encore, cela représente 4

paquets de tâches :

- Développement d'un agent avancé.
- Compléter l'interface graphique avec des graphiques affichant les statistiques des joueurs.
- Uniformiser le protocole réseau de manière à ce que tous les logiciels puissent communiquer.
- Statistiques temporelles, authentification des joueurs.

Communication. La présence est obligatoire à chaque séance, donc il faut privilégier la communication orale avec les enseignants. En dehors des séances, il faut obligatoirement passer par le forum d'e-uapv (pas d'emails). La communication entre étudiants peut également passer par ce forum, mais il faut aussi considérer le système d'issues et de commentaires de la plate-forme d'hébergement du code source (cf. Section 5).

Respect des consignes. La plupart des problèmes rencontrés par les étudiants les années précédentes était due au fait qu'ils ne respectaient pas scrupuleusement les consignes. Nous vous recommandons donc d'être extrêmement attentifs au contenu des documents qui vous sont fournis, et des instructions qui vous sont communiquées oralement. Si vous ne comprenez pas certains des points mentionnés, ou si vous détectez des imprécisions ou des contradictions, vous devez solliciter les enseignants.

3 Rédaction des rapports

Tous les rapports produits au cours du projet doivent respecter les contraintes décrites ci-dessous.

Système LaTeX. Tous les documents à rendre pour évaluation doivent être écrits en utilisant le système LaTeX [3] : aucun autre format ne sera accepté. La remise se fait en déposant sur la plateforme e-uapv le PDF produit par LaTeX. Attention à ne pas déposer le code source LaTeX, mais bien le PDF résultant de sa compilation.

OverLeaf. Vous devez rédiger vos documents LaTeX en utilisant le site OverLeaf. Il s'agit d'un service Web permettant l'écriture collaborative de documents en LaTeX. Il offre l'avantage de ne pas nécessiter d'installer quoi que ce soit sur votre machine.

La création d'un compte est gratuite. Vous devez utiliser un nom d'utilisateur permettant de vous identifier clairement : `prenom.nom`.

Modèle de document / Tutoriel LaTeX. Pour chaque rapport, vous devez utiliser le modèle qui vous est fourni. Le but est d'avoir une présentation et une structure uniformes, afin que vous puissiez plus facilement comprendre ce que les autres groupes font, et aussi pour faciliter l'évaluation de vos rapports.

Ce modèle est disponible sur OverLeaf à l'adresse suivante :
`https://www.overleaf.com/latex/templates/modele-rapport-uapv/pdbgdpzsgwrt`
Attention à toujours utiliser la dernière version du modèle disponible pour chaque nouveau rapport.

Notez que ce modèle est également un tutoriel vous expliquant les fonctionnalités de LaTeX dont vous avez besoin pour écrire vos rapports. Une partie de ces instructions apparaît dans la version compilée du document (le PDF), mais aussi dans son code source, sous la forme de commentaires. Il est très important que vous teniez compte de toutes ces informations.

4 Contraintes d'implémentation

Langage de programmation. Le logiciel doit obligatoirement être implémenté en Java.

Licence du logiciel. Le logiciel produit sera publié sous licence libre GPL [4] et hébergé sur une plateforme GitLab (cf. Section 5).

Environnement de programmation. Le logiciel doit prendre la forme d'un projet Eclipse. Il est donc recommandé d'utiliser Eclipse pour le développement. Vous pouvez très bien utiliser

un autre IDE tant que celui-ci n'interfère pas avec la forme du projet (i.e. un projet Eclipse) et que celui-ci peut être exécuté depuis Eclipse.

Réutilisation de ressources. Il faut donner la priorité aux bibliothèques standard Java. Il est possible d'utiliser des bibliothèques externe, à trois conditions : 1) la bibliothèque ne doit pas se substituer aux fonctionnalités à implémenter dans le cadre de ce projet ; 2) elle doit être elle-même sous licence libre ;et 3) elle doit être validée par les encadrants.

Conventions Java. Le code source doit respecter au maximum les standards existants, et les conventions Java, notamment en ce qui concerne les identificateurs (noms de variables, fonctions, classes, etc.). Voir les ressources pour plus de détails (Section 6).

Commentaires. Le code source doit impérativement être commenté selon le standard Javadoc [2]. Vous devez :

- Pour chaque *classe* : insérer un commentaire qui la décrit et indique son (ou ses) auteur(s) (tag `@author`).
- Pour chaque *champ* : écrire une ligne de commentaire expliquant ce qu'il représente.
- Pour chaque *méthode* : écrire un commentaire qui la décrit, puis indiquer ses paramètres (tag `@param`), valeurs de retour (tag `@return`), exceptions (tag `@throws`) et auteur(s) (tag `@author`), si ceux-ci diffèrent de ceux de la classe.

Internationalisation. Le logiciel et son code source sont destinés à être accessibles à un public international. Les identificateurs (noms de classes, méthodes, champs, etc.) doivent donc être en anglais, de même que les commentaires et la documentation.

5 Hébergement du code source

Hébergement. Le projet sera hébergé sur GitLab. L'utilisation de ce service d'hébergement est gratuite.

Notez que c'était la plateforme GitHub qui était utilisée les années précédentes. Cependant, cette année chaque groupe est chargé de réaliser la même tâche, et il est donc nécessaire de pouvoir créer des dépôts privés, afin d'éviter tout plagiat entre groupes. GitHub ne fournit pas cette fonctionnalité, alors que GitLab oui, donc ce dernier lui a été préféré.

Un dépôt de référence a déjà été créé sur le compte du CERI sur GitLab : pour les détails, reportez-vous au document consacré à Git qui vous est fourni par ailleurs.

Fonctionnalités. Chaque groupe doit tirer parti des fonctionnalités offertes par GitLab pour s'organiser. Il faut en particulier utiliser les outils suivants :

- *Issues* pour faire du suivi sur les points à traiter ;
- *Wiki* pour écrire la documentation du projet.

Les enseignants se baseront sur l'activité enregistrée sur GitLab pour évaluer votre travail, mais aussi pour déterminer qui a copié sur qui en cas de plagiat. Pensez donc à bien utiliser la plate-forme pour rendre votre travail visible.

Conventions. On attend de chaque groupe qu'il utilise la branche `master` du projet seulement pour les versions stables du logiciel. Le développement de chaque version doit faire l'objet d'une sous-branche spécifique de `master`, qui est fusionnée avec elle seulement quand le développement de la version est terminé. Le développement de chaque fonctionnalité du logiciel doit faire l'objet d'une sous-branche spécifique de la version concernée, qui est fusionnée à celle-ci quand le développement de la fonctionnalité est terminé.

Documentation. En plus de la Javadoc (cf. Section 4), le projet devra être accompagné d'un fichier `README.md` (rédigé en langage markdown) indiquant :

- Une présentation du logiciel, avec la liste de ses auteurs ;
- La description de l'organisation du code source ;
- Des explications sur l'installation du logiciel ou des ressources nécessaires ;
- La procédure à suivre pour l'exécuter et l'utiliser ;

- La liste de ses dépendances (i.e. bibliothèques utilisées dans le code source) ;
- Éventuellement les références bibliographiques utilisées lors du développement.

En plus de cette documentation technique, le projet devra contenir des pages Wiki constituant le mode d'emploi du jeu : explication des règles, de l'utilisation de l'interface graphique, comment débuter une partie, faire une partie réseau, etc. Autrement dit, le `README.md` et la Javadoc s'adressent aux programmeurs, alors que le Wiki vise l'utilisateur final.

6 Ressources

Cette section regroupe quelques ressources qui peuvent vous être utiles (voire nécessaires) dans le cadre de ce projet. Si vous trouvez d'autres ressources que vous jugez intéressantes, n'hésitez pas à les poster sur le forum d'e-uapv.

Java. Le langage imposé pour la phase d'implémentation :

- Java Platform, Standard Edition 10 API Specification : la documentation officielle de Java.
- Java naming conventions : un document décrivant les conventions de nommage en Java.
- Naming convention (programming) : article Wikipedia sur le même sujet.

Git. Le système imposé pour la gestion de versions :

- Le tutoriel Git/GitHub disponible sur e-uapv.
- Git Tutorial : un tutoriel sur le site officiel.
- Gérez vos codes sources avec Git : un tutoriel en français sur OpenClassrooms.
- EGit : le plugin Git pour Eclipse.
- Visualizing Git Concepts with D3 : un simulateur permettant de mieux comprendre le fonctionnement de Git.

GitLab. Le service Web imposé pour l'hébergement en ligne du projet :

- User documentation : documentation pour les utilisateurs du service.
- How to fork a project : comment créer une copie d'un projet.

LaTeX. Le système de composition imposé pour les rapports :

- OverLeaf : service collaboratif gratuit permettant de rédiger et compiler du LaTeX en ligne.
- Modèle de rapport et tutoriel LaTeX : disponible sur le site OverLeaf (voir ci-dessous), à utiliser obligatoirement pour tous vos rapports. Il s'agit aussi d'un tutoriel à lire attentivement.
- Lexique : document décrivant la terminologie et les notations mathématiques à utiliser obligatoirement dans vos rapports.
- Tutoriel LaTeX : une introduction à LaTeX sur le site UKO.
- LaTeX : Wikibook expliquant comment utiliser LaTeX (en anglais).

Références

[1] Jeux de Nim. *Pingu party*. 2008. URL : https://www.jeuxdenim.be/jeu-PinguParty.

[2] Wikipedia. *Javadoc*. 2018. URL : http://en.wikipedia.org/wiki/Javadoc.

[3] Wikipedia. *LaTeX*. 2018. URL : https://fr.wikipedia.org/wiki/LaTeX.

[4] Wikipedia. *Licence publique générale GNU*. 2018. URL : https://fr.wikipedia.org/wiki/Licence_publique_g%C3%A9n%C3%A9rale_GNU.