Projet NetWorddle

*chilowi at u-pem.fr*  
2018

Le but de ce projet est de réaliser un jeu de Worddle jouable par plusieurs utilisateurs en réseau. Il s'agit en fait d'une déclinaison du célèbre jeu Boggle (on évitera d'utiliser ce nom car il s'agit d'une marque déposée par Habsro).

Principe du Worddle

Le Worddle est un jeu sur les mots se pratiquant sur une grille de lettres rectangulaire (de N lignes sur M colonnes). Il s'agit de trouver en temps limité le maximum de mots composés de lettres voisines de la grille (avec une notion de voisinage utilisant la 8-connexité : nord, nord-ouest, ouest, sud-ouest, sud, sud-est, est, nord-est), une lettre ne pouvant être utilisée qu'une fois au plus. Pour plus d'informations, vous pouvez consulter [la page de Wikipedia consacrée à ce jeu](https://fr.wikipedia.org/wiki/Boggle).

Génération de grille

Tout d'abord, nous réalisons une classe capable de générer des grilles aléatoires de Worddle. Nommons là PuzzleGenerator et dotons là d'une méthode de signature char[][] generate(int n, int m) générant une grille de n lignes sur m colonnes en retournant un tableau à deux dimensions de *char* comportant les lettres tirées au sort.

Le jeu traditionnel fonctionne avec des dés placés dans une boîte compartimentée. Nous ajoutons une méthode void addDice(char[] dice) permettant d'ajouter un dé utilisable par le générateur. Un tel dé est défini par la valeur de ses six faces qui sont des lettres.

Une fois tous les dés ajoutés (nous pouvons ajouter au besoin plusieurs fois le même dé), nous lançons la génération de grille avec la méthode generate précédemment décrite. Tout d'abord, pour la génération, nous sélectionnons les dés à utiliser. Si nous disposons de plus de dés que la taille de la grille, nous choississons aléatoirement les dés nécessaires. Si nous n'en disposons de pas assez, nous dupliquons aléatoirement certains dés. Nous avons alors n\*m dés sur lesquels nous sélectionnons aléatoirement une face qui sera la lettre de la cellule de la grille.

Il peut être un peu rébarbatif d'initialiser à la main le générateur en ajoutant tous les dés, c'est pourquoi nous ajoutons une méthode void loadDices(String path) qui charge les dés indiqués dans le fichier texte (encodage UTF-8) de chemin spécifié. Chaque ligne de ce fichier contient les caractères des faces d'un dé. Un fichier de définition de dés vous est fourni en exemple pour l'anglais américain (adapté aux fréquences des lettres dans cette langue).

Vous pouvez rajouter une classe disposant d'une méthode main afin de tester votre générateur de grille.

Vérification de solution

Le serveur de jeu endossera le rôle d'arbitre. Il doit donc pouvoir vérifier si une solution proposée par un joueur est valide pour une grille donnée. On écrira pour cette tâche la classe AnswerChecker. La classe aura une méthode de signature boolean isValidAnswer(char[][] grid, String word) qui vérifiera si le mot en argument est une réponse correcte pour la grille spécifiée.

Vous avez de la chance, nous vous fournissons cette classe (mais si vous souhaitez la réimplanter à votre façon, il n'y a pas de problème).

Déroulement du jeu

Un serveur doit pouvoir organiser des parties de Worddle entre différents joueurs. Les joueurs s'affrontent en duel. On configure préalablement le serveur en spécifiant un numéro de port de contact ainsi que la langue de jeu supportée (en fait le chemin vers le dictionnaire). Vous devez définir un protocole de communication entre les clients (joueurs) et le serveur de jeu qui respecte le déroulement suivant :

* Tout d'abord un joueur se connecte sur le serveur de jeu. Il y déclare son identifiant avec son mot de passe. Si l'identifiant est utilisé pour la première fois, le serveur enregistre cet identifiant avec le mot de passe librement choisi. Si l'identifiant avait déjà été utilisé auparavant, le joueur doit proposer le même mot de passe que précédemment, faut de quoi le joueur est rejeté.
* Une fois authentifié, le joueur est placé dans un salon d'attente. Il doit patienter pour jouer.
* Une partie de jeu est organisée avec deux joueurs. Il faut donc patientier jusqu'à ce que deux joueurs soient disponibles dans le salon d'attente. Le jeu démarre alors.
* Le serveur communique aux deux joueurs la grille de jeu ; il faudra donc utiliser à cet effet votre générateur de grille PuzzleGenerator.
* La partie débute et dure un temps limité configuré (par exemple 2 minutes). Pendant cette période, le serveur reçoit les propositions des joueurs : la validité de chacun des mots reçu est vérifiée avec AnswerChecker.
* A la fin de la période de jeu, le serveur envoie à chacun des joueurs les mots validés par son adversaire ; il y associe un score. La somme de ces scores détermine le score de la partie pour un joueur. Ce score vient incrémenter le score global du joueurs correspondant à la somme des scores de toutes les parties qu'il a jouées.
* Les deux joueurs sont déconnectés du serveur du jeu. Ils peuvent néanmoins se reconnecter pour jouer de nouveau.

Protocole de communication

On mettra en place un protocole de communication basée sur le protocole de transport TCP. Il fonctionnera en mode texte en utilisant l'encodage UTF-8 selon un modèle requête-réponse : le client et/ou le serveur peuvent émettre une requête et le correspondant y répond. Chaque requête et chaque réponse tiennent sur une ligne de texte terminée par un caractère \n.

Une requête est une ligne de texte du format suivant :

Q:n:<contenu de la requête>

Q indique qu'il s'agit d'une requête (*query*), n est un entier indiquant le numéro de la requête (incrémenté à chaque requête) ; le contenu de la requête est ensuite spécifié.

Une réponse est une ligne de texte du format suivant :

R:n:<contenu de la réponse>

R indique qu'il s'agit d'une réponse (*reply*), n reprend le numéro de la requête auquel cette réponse répond ; le contenu de la réponse est indiqué ensuite.

Nous répertorions maintenant les différentes requêtes possibles et indiquons entre-crochets si celles-ci peuvent être initiées par le client ([C]), le serveur ([S]) ou les deux correspondants ([CS]). Le nom d'une requête peut être suivi de plusieurs arguments séparés par des virgules.

* connect,username,password [C] Pour permettre au client de s'authentifier sur le serveur en indiquant son nom d'utilisateur et son mot de passe. Le serveur peut répondre NEW s'il s'agit d'un nouvel utilisateur, OK s'il s'agit d'un utilisateur déjà connu avec un mot de passe correct ou KO si l'utilisateur est connu mais le mot de passe incorrect. En cas de réponse KO, le serveur clôture la connexion TCP.
* disconnect [CS] Pour permettre une déconnexion en cours de jeu ou d'attente dans le salon. Peut être initié aussi bien par le client que par le serveur (par exemple si son administrateur décide de l'arrêter : les sessions de jeu doivent alors être stoppées). Dans tous les cas, le correspondant doit répondre OK et clôturer la connexion TCP.
* start,duration,row0,row1,...,rowN [S] Indique le démarrage d'une nouvelle partie avec la grille spécifiée ligne par ligne (chaque ligne est une séquence de lettres) ; le serveur indique aussi la durée de la partie en secondes (entier *duration*). Cette requête est envoyée simultanément aux deux joueurs d'une partie par le serveur. Les deux joueurs doivent répondre OK.
* proposition,word [C] Permet au client d'envoyer une proposition de mot au serveur. Le serveur y répond par score,n, nétant le score de la proposition (0 si le mot n'est pas validé).
* stop,word0/a,word1/b,word2/c...,wordi/z [S] Permet au serveur d'indiquer la fin d'une partie ; il en profite pour indiquer les mots trouvés par l'adversaire en y associant le score.

Il vous sera demandé d'ajouter de nouvelles requêtes permettant à un joueur de connaître son propre score global (score cumulé de toutes les parties), le score global de n'importe quel autre utilisateur ainsi que les scores des meilleurs joueurs.

Serveur

Un serveur respectant le protocole précédemment décrit devra être implanté. Il doit être démarrable avec la commande suivante (si l'on souhaite qu'il soit contactable sur le port 2017, qu'il utilise le dictionnaire de mots french et qu'il se joue sur des grilles de taille 5x4 pour des parties de 120 secondes) :

java fr.upem.networddle.Server 2017 french 5 4 120

Client

Dans un premier temps, il n'est pas obligatoire d'implanter un client : en effet, il est possible de tester votre serveur avec un simple client TCP générique en mode texte tel que netcat. Par exemple pour contacter le serveur lancé avec la commande précédente sur la machine locale, on pourra utiliser nc localhost 2017. Néanmoins l'ergonomie de jeu n'est alors pas optimale. Il faudra donc implanter un client plus convivial qui fonctionne soit en console ou (encore mieux) à l'aide d'une interface graphique.

Le client sera lancé avec la commande suivante pour contacter le serveur sur le port 2017 :

java fr.upem.networddle.Client 2017

Personnalisation du jeu

De nombreux aspects du jeu peuvent être personnalisés à votre convenance. Par exemple, vous pouvez adopter la politique de score de votre choix (le plus simple étant d'attribuer des points égaux à la longueur du mot) ; on pourra par exemple attribuer un bonus pour un mot qui ne serait trouvé que par un seul adversaire.

Une amélioration possible serait de généraliser les parties à un nombre de joueurs arbitraire (3,4, ... n joueurs).

[Rendu du projet](https://elearning.u-pem.fr/mod/assign/view.php?id=71307)

Le projet fera l'objet d'une soutenance d'une quinzaine de minutes où vous ferez la démonstration de votre jeu à l'aide de trois machines différentes (serveur et deux joueurs). Quelques jours auparavant, vous devrez rendre votre code sur Moodle sous la forme d'une archive zip contenant les sources (fichiers .java) organisées en paquetages (et si possible bien commentées). Une documentation devra également être jointe ; elle se compose des parties suivantes :

* un fichier README à la racine du répertoire du projet indiquant comment compiler votre projet ainsi que comment configurer et exécuter le serveur ainsi que le client
* un fichier rapport.pdf à la racine contenant un rapport de développement du projet au format PDF où vous décrirez l'architecture du projet, les difficultés rencontrées et les solutions techniques adoptées pour les surmonter ; ce rapport devra également indiquer la répartition des tâches de chacun dans le projet. Les dates exactes de rendu feront l'objet d'une communication sur le forum Moodle du cours.

Vous avez le choix d'écrire votre documentation en français ou en anglais. Vous devrez également rendre le code source de votre projet disponible sous [une licence libre de votre choix](https://aful.org/ressources/licences-libres) : ajoutez à la racine un fichier LICENSE avec le contenu de la licence du projet.