1 Descriptif du projet

On souhaite réaliser une adaptation du jeu de lettres « Boggle » (Hasbro), dans lequel les joueurs gagnent des points en essayant de construire le plus possible de mots (et les plus longs mots possibles) à partir d'une grille de lettres aléatoire. Ce jeu se joue en principe à plusieurs joueurs (tous les joueurs étant confrontés à la même grille pendant un temps limité), mais pour simplifier le problème nous l'adapterons en un jeu tour par tour en changeant la grille entre chaque joueur. Pour former les mots, on peut partir de n'importe quelle case, mais on doit obligatoirement prendre la lettre suivante sur une case adjacente (l'une des 8 cases voisines) et ne jamais utiliser deux fois la même case. Un mot ne compte que s'il comporte au moins 3 lettres (pour une grille 4×4) ou 4 lettres (pour une grille 5×5).
Ce projet sera réalisé en Java avec interface graphique en Swing. Les caractéristiques du jeu (taille du plateau, nombre de joueurs, dictionnaire à utiliser, valeur des dés pour le tirage des lettres de la grille, etc.) doivent être personnalisables et permettre des extensions.

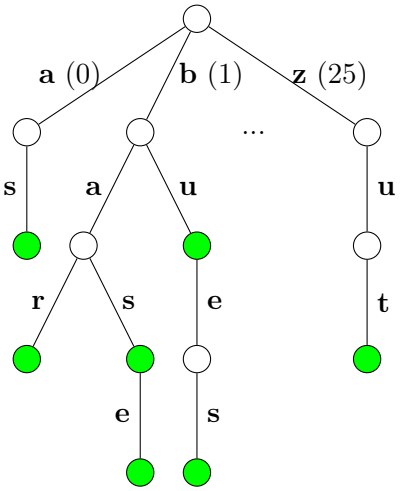
Pour pouvoir jouer de façon intéressante il faut au minimum implémenter les règles suivantes :

- On joue avec un nombre quelconque de joueurs (entre 1 et 5), soit pour un nombre de manches fixé au départ, soit jusqu'à ce qu'un joueur atteigne un score cible.
- Le tour de jeu (une manche) : chaque joueur joue à tour de rôle. Le programme tire une grille aléatoirement (voir ci-dessous), le joueur désigne alors ses mots en cliquant sur les lettres correspondantes puis sur un bouton `AJOUTER`. Lorsqu'il pense avoir fini, il clique sur un bouton `TERMINÉ` : on compte alors son score et on passe au joueur suivant. Le score et calculé en fonction de la taille des mots trouvés qui sont présents dans le dictionnaire et ont été construits en respectant la règle (1 pt pour 3-4 lettres, 2 pour 5, 3 pour 6, 5 pour 7, et 11 au-delà).
- Les grilles sont des carrés de côté N (en général $N = 4$) et sont construites par le lancer de N^2 dés. Les lettres affichées sur les faces visibles des $N \times N$ dés donnent le contenu de la grille. Évidemment pour assurer la jouabilité toutes les lettres ne sont pas équiprobables, il faut donc permettre la configuration des dés via un fichier.
- Un mot est valide si : **1)** il compte au moins $N - 1$ lettres ; **2)** il est construit de proche en proche, c'est-à-dire que ses lettres successives doivent être lues sur des cases adjacentes deux à deux (on ne peut pas « sauter » sur une case distante) ; **3)** il n'utilise pas deux fois le même dé ; **4)** il est présent dans le dictionnaire utilisé pour le jeu (en général liste de mots avec les formes fléchies).
- Il faut pouvoir charger les données de configuration du jeu en format texte (CSV/`Properties`).

2 Mise en œuvre

Gestion du dictionnaire

Pour accélérer la vérification des mots, les stocker efficacement et permettre au programme d'aider les joueurs, le dictionnaire utilisera un **arbre lexical** : chaque nœud de l'arbre peut avoir jusqu'à 26 sous-arbres, correspondant chacun à une lettre de l'alphabet (on ne tiendra pas compte des accents), et possède un attribut booléen qui indique si le nœud correspond à la fin d'un mot (cf. figure ci-contre). Ce sont donc non pas des arbres binaires mais des arbres « 26-aires ».
Représentation d'un arbre lexical : chaque branche correspond à une lettre possible ; les nœuds contiennent un booléen qui indique si la succession des branches parcourues jusqu'à ce nœud correspond à un mot (ici : vert = vrai). L'arbre présenté ici contient donc les mots *as*, *bar*, *bas*, *base*, *bu*, *bues* et *zut*, mais pas le mot *bue* puisque le nœud auquel on arrive par les branches `b`–`u`–`e` est positionné à faux.



Décomposition

L'application sera découpée en 3 packages au minimum, assurant les fonctionnalités suivantes :

- `boggle.gui` sera en charge de toute la partie graphique (interface utilisateur, dessin du plateau de jeu, affichage des mots et des scores, événements du clavier et de la souris, etc.) ;
- `boggle.jeu` contiendra tout ce qui concerne la gestion du cycle de jeu proprement dit, à savoir le traitement des joueurs, le calcul des scores, la gestion de la fin de tour et de la fin de partie ;
- `boggle.mots` contiendra tout ce qui a un rapport avec les mots : notamment, des classes `GrilleLettres`, `De` et `ArbreLexical`. En particulier, la classe `De` doit gérer le fait qu'il y a dans le jeu exactement $N \times N$ dés dont les valeurs sont spécifiées dans un fichier de configuration. Une grille de lettres est générée à partir du placement aléatoire de ces dés (position dans la grille et face visible).

Documents fournis

Vous disposez sur Moodle d'une archive `projet-long.tgz` contenant un répertoire `sources` (avec des éléments de code destinés à spécifier un certain nombre de fonctionnalités attendues), un répertoire `config` contenant les fichiers textes permettant la configuration du jeu (dés, dictionnaires), et des répertoires pour le bytecode, la javadoc et le manuel utilisateur.

3 Travail demandé

Ce projet est à réaliser en binôme. Vous ne serez pas jugé exclusivement sur le rendu graphique du jeu. L'accent doit porter avant tout sur une implémentation efficace des classes et de leurs relations, ainsi que sur la documentation.

Objectifs minimaux à atteindre

Le point auquel vous devrez arriver au minimum est le suivant :

- un programme portable (au moins Windows/Linux) ;
- une interface graphique en Swing, parfaitement fonctionnelle, avec affichage d'une grille de lettres cliquable, des mots proposés par chaque joueur et des points qu'ils rapportent ;
- la possibilité de jouer à N joueurs humains (N paramétrable) selon toutes les règles obligatoires décrites page 1, et via des fichiers texte pour la configuration du jeu ;
- une gestion des fins de partie (arrêt et désignation du gagnant).

Améliorations souhaitées

Vous devez également choisir quelques améliorations à votre gré, parmi les idées suivantes :

- possibilité de choisir (via un menu ou un fichier `Properties`) la taille de la grille, le dictionnaire et la configfuration des dés (p.ex. un dé `"Qu"`) ;
- mémorisation des 10 meilleurs scores ;
- changement d'état des dés déjà utilisés (p.ex. fond rouge) ;
- possibilité de proposer les mots non pas en cliquant mais en les tapant dans une zone de texte (plus rapide pour le joueur, mais vérification plus compliquée !) ;
- mise en place d'un sablier qui limite le temps de chaque joueur et donne automatiquement la main au joueur suivant (le bouton `TERMINÉ` reste utilisable avant le temps imparti) ;
- aide contextuelle : si le joueur a commencé un mot d'au moins 3 lettres, le programme signale (par exemple par une surbrillance) les cases adjacentes à la dernière case sélectionnée, susceptibles de produire un mot plus grand (et ainsi de suite au fur et à mesure que le mot grandit) ;
- conception d'un joueur piloté par l'ordinateur (de force réglable) :
  - version basique : le joueur artificiel effectue une recherche en profondeur d'abord à partir d'une position aléatoire ;
  - version améliorée : le joueur artificiel s'appuie sur des informations statistiques contenues dans l'arbre lexical (p.ex. nombre de mots par sous-arbre) pour guider son exploration.

Documents à rendre

L'ensemble du projet doit être rendu sur la plateforme Moodle dans une archive au format `tar/gz` contenant :

- tout le répertoire de travail (sources, doc, classes, manuel et config) ;
- un exécutable sous forme d'archive JAR (voir documentation de Java) ;
- un fichier texte `README` décrivant la procédure à suivre pour compiler les sources et démarrer le jeu ;
- un court manuel utilisateur sous la forme d'une page HTML, indiquant comment jouer (buts du jeu, règles ayant été implémentées, ajouts introduits, description de l'interface graphique, etc.).

Quel que soit le degré d'avancement de votre projet, tout doit compiler et se conformer aux squelettes de code et fichiers de configuration fournis.

Une **démonstration** de votre projet aura lieu le **mardi 15 décembre 2015** (après le DS). Cette démonstration (de 10 à 15 minutes maximum, pendant le créneau 16h–18h) doit permettre d'illustrer les points réalisés. L'archive (code documenté !) sera rendue au plus tard le **vendredi 18 décembre 2015** sur Moodle.