Université Gustave Eiffel
Licence 3 Informatique
Programmation Orientée Objet avec Java
Groupe 2 et Groupe 3
Année universitaire 2020-2021

Manuel développeur - Baba Is You

Étudiants : **HAMADI** Andil-dine (Gp. 2), **JAAFRI** Amyr (Gp. 3)

Préambule

Ce fichier comporte la documentation des améliorations, des corrections et de l'architecture du projet.

Amélioration (depuis soutenance bêta) :

- Création de niveaux (*level2, level3, …, level5 ; cf.* **fr.umlv.file**) ;
- Passage des niveaux automatisés ;
- Gestion des victoires et des défaites ;
- Ajout des attributs : *Sink*, *Defeat*, *Nothing ;*
- Ajout du nom : *Skull* ;
- Sauvegarde/Chargement (Sauvegarde indisponible à ce jour).

Correction (depuis soutenance bêta) :

- Factorisation sur les *Drawer* (cf. **fr.umlv.Drawer**) ;

Documentation :

Avant-propos

Cette documentation exhaustive se veut un complément à la *Javadoc*. Le lecteur y trouvera une brève explicitation des visées de chaque champ et méthode.

Ainsi nous conseillons vivement au lecteur de ne l'utiliser que comme appendice.

fr.umlv.attribute

You.java

Type : *Class*

Champs :

- Cette classe ***implements*** l'interface **Attribute.java** (*cf.* **fr.umlv.attribute**)

Description : permet de représenter l'attribut *You* sur le plateau.

- ***boolean react (int x, int y, int distance, KeyboardKey touch, ArrayList<Items> listIcons)*** : l'item correspondant est le joueur;
- ***String toString ()***: obtenir le nom de l'attribut ;
- ***String toTextFormat ()*** : obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.).

Win.java

Type : *Class*

Champs :

- Cette classe ***implements*** l'interface **Attribute.java**

(*cf.* **fr.umlv.attribute**)

Description : permet de représenter l'attribut *Win* sur le plateau.

- ***boolean react (int x, int y, int distance, KeyboardKey touch, ArrayList<Items> listIcons)*** : l'item correspondant fait gagner le joueur ;

- ***String toString ()***: obtenir le nom de l'attribut ;

- ***String toTextFormat ()*** : obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.).

➢ Stop.java

Type : *Class*

Champs :

- Cette classe ***implements*** l'interface **Attribute.java** (*cf.* **fr.umlv.attribute**)

Description : permet de représenter l'attribut *Stop* sur le plateau.

- ***boolean react (int x, int y, int distance, KeyboardKey touch, ArrayList<Items> listIcons)*** : l'item correspondant n'est pas déplaçable par le joueur ;

- ***String toString ()***: obtenir le nom de l'attribut ;

- ***String toTextFormat ()*** : obtenir le format du texte à écrire sur un fichier (*cf.*

Sauvegarde/Chargement.).

- Sink.java

Type : *Class*

Champs :

- Cette classe ***implements*** l'interface **Attribute.java** (*cf.* **fr.umlv.attribute**)

Description : permet de représenter l'attribut *Sink* sur le plateau.

- ***boolean react (int x, int y, int distance, KeyboardKey touch, ArrayList<Items> listIcons)*** : l'item correspondant tue le joueur ;
- ***String toString ()***: obtenir le nom de l'attribut ;
- ***String toTextFormat ()*** : obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.).

- Push.java

Type : *Class*

Champs :

- Cette classe ***implements*** l'interface **Attribute.java** (*cf.* **fr.umlv.attribute**)

Description : permet de représenter l'attribut *Push*

sur le plateau.

- ***boolean react (int x, int y, int distance, KeyboardKey touch, ArrayList<Items> listIcons)*** : l'item correspondant est déplaçable par le joueur;

- ***String toString ()***: obtenir le nom de l'attribut ;

- ***String toTextFormat ()*** : obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.).

➢ Nothing.java

Type : *Class*

Champs :

- Cette classe ***implements*** l'interface **Attribute.java** (*cf.* **fr.umlv.attribute**)

Description : permet de représenter l'attribut *Nothing* sur le plateau.

- ***boolean react (int x, int y, int distance, KeyboardKey touch, ArrayList<Items> listIcons)*** : l'item correspondant n'interagit pas avec le joueur ;

- ***String toString ()***: obtenir le nom de l'attribut ;

- ***String toTextFormat ()*** : obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.).

➢ Defeat.java

Type : *Class*

Champs :

- Cette classe ***implements*** l'interface **Attribute.java** (*cf.* **fr.umlv.attribute**)

Description : permet de représenter l'attribut *Defeat* sur le plateau.

- ***boolean react (int x, int y, int distance, KeyboardKey touch, ArrayList<Items> listIcons)*** : l'item correspondant tue le joueur ;
- ***String toString ()***: obtenir le nom de l'attribut ;
- ***String toTextFormat ()*** : obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.).

➢ Attribute.java

Type : *Interface*

Description : permet de représenter l'intersection des classes **Defeat.java**, Nothing**.java**, Push**.java, Sink.java , Stop.java, Win.java, You.java**. Ainsi :

- ***boolean react (int x, int y, int distance, KeyboardKey touch, ArrayList<Items> listIcons)*** : définir comment l'item doit interagir avec les autres items adjacents;

- ***String*** *toTextFormat ()* : obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.).

fr.umlv.ControlGame

Updates.java

Type : *Class*

Champs :

- Cette classe ***extends*** la classe **Plateau.java** (*cf.* **fr.umlv.ControlGame**)

Description : permet de gérer l'état du plateau.

- ***Items*** *find (**int** x, **int** y, ArrayList<**Items**> list)*: trouver un item sur le plateau ;

- ***void*** *ruleIsValid (**Items** name, **Items** prop, **Items** word, ArrayList<**Items**> listWords, ArrayList<**Rule**> listRules)* : ajouter une règle si celle-ci est valide ;

- *ArrayList<**Rule**> updateRules (ArrayList<Items> listWords)* : mettre à jour les règles ;

- ***void** applyRule (ArrayList<**Items**> basicList, ArrayList<**Rule**> listRules)* : appliquer les règles sur les items du plateau.

➢ StateGame.java

Type : *Class*

Description : permet de gérer les cas de victoire et de défaite.

- ***boolean** isWin (**Graphics2D** graphics, ArrayList<**Items**> list)* : gérer les victoires ;

- ***void** isDefeat (**Graphics2D** graphics, ArrayList<**Items**> list)* : gérer les défaites ;

➢ Plateau.java

Type : *Class*

Champs :

- *ArrayList<ArrayList<**Items**>> listItems* : liste à deux dimensions représentant le plateau ;

- ***final int** size* : nombre de case sur le plateau ;

- ***final int** getDistance*: pas de dé

Description : permet de gérer l'état du plateau.

- ***String*** *toString ()*: obtenir les items présents sur le plateau ;

- ***void*** *drawPlateau ()* : dessiner l'état du plateau ;

- ***int*** *getDistance ()* : getter de la distance ;

- ***void*** *move (**Items** it, **KeyboardKey** touch)* : déplacer un item sur le plateau.

➢ Items.java

Type : *Class*

Champs :

- ***int*** *x* : coordonnée en abscisse de l'item ;

- ***int*** *y* : coordonnée en ordonné de l'item ;

- ***final Drawer*** *drawer* : la représentation graphique de l'item (*cf.* **fr.umlv.drawer**);

- ***Attribute*** *attribute* : l'attribut de l'item (*cf.* **fr.umlv.attribute**).

Description : permet de gérer un item sur le plateau (initialisation, déplacement, etc …).

- ***void*** *setX (**int** x)* : setter de la coordonnée en abscisse ;

- ***void*** *setY (**int** y)* : setter de la coordonnée en ordonnée ;

- ***int*** *getX (**int** x)* : getter de la coordonnée en abscisse ;

- ***int*** *getY (**int** y)* : getter de la coordonnée en ordonnée ;

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**

- ***Drawer*** *getName ()* : getter de la représentation graphique de l'item ;

- ***Attribute*** *getAttribute ()* : getter de la représentation graphique de l'item ;

- ***void*** *setAttribute (**Attribute** prop)* : setter de l'attribut de l'item ;

- ***boolean*** *equals (**Object** obj)* : comparer la position de deux items ;

- ***String*** *toTextFormat ()* : obtenir le format texte de l'item à écrire sur un fichier (*cf.* Sauvegarde/ Chargement.).

➢ Initialisations.java

Type : *Class*

Champs :

- Cette classe ***extends*** la classe **Plateau.java** (*cf.* **fr.umlv.ControlGame**)

- Le constructeur fait seulement appel au super-constructeur.

Description : permet d’initialiser les items et les mots sur le plateau.

- *ArrayList<**Items**> initBasicList (ArrayList<**Items**> basicList)* : initialisation de la liste d’items ;

- *ArrayList<**Items**> initWordList (ArrayList<**Items**> basicList)* : initialisation de la liste de mots.

fr.umlv.drawer

Drawer.java

Type : *Interface*

Description : permet de représenter l’intersection des classes **DrawIcons.java** et **DrawWord.java**, c’est-à-dire l’affichage, respectivement, des **icônes** et des **mots** sur le plateau. Ainsi :

- ***void** draw (**Graphics2D** graphics, **Color** fond, **int** x, **int** y)*: dessiner la forme de l’icône.**;**

- ***String** toTextFormat ()* : obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.).

DrawIcons.java

Type : *Class*

Champs :

- Cette classe ***implements*** l'interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

  - ***Ellipse2D.Float*** *ellipse* : un cercle représentant une icône;

Description : permet de représenter une **icône** sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l'icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ DrawWord.java

Type : *Class*

Champs :

- Cette classe ***implements*** l'interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

  - ***Rectangle2D.Float*** *rectangle* : un rectangle représentant un mot;

Description : permet de représenter un **mot** sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur le rectangle**;**

- ***Color*** *getColor ()*: obtenir la couleur du rectangle ;

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.).

fr.umlv.drawer.image

Water.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawIcons.java** (*cf*. **fr.umlv.drawer**).

- Le constructeur fait appel au super-constructeur.

Description : permet de représenter l'icône *Water* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l'icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ Wall.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawIcons.java** (*cf*. **fr.umlv.drawer**).

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter l'icône *Wall* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l'icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.).

➢ Empty.java

Type : *class*

Champs :

- Cette classe ***implements*** l'interface **Drawer.java** (*cf*. **fr.umlv.drawer**).

Description : permet de représenter les cases du plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.*

➢ Skull.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawIcons.java** (*cf*. **fr.umlv.drawer**).

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter l'icône *Skull* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l'icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ Rock.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawIcons.java** (*cf*. **fr.umlv.drawer**).

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter l'icône *Baba* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**
- ***Color*** *getColor ()*: obtenir la couleur de l'icône.
- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ Flag.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawIcons.java** (*cf*. **fr.umlv.drawer**).

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter l'icône *Flag* sur

le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l'icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ Baba.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawIcons.java** (*cf*. **fr.umlv.drawer**).

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter l'icône *Baba* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l'icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

fr.umlv.drawer.word.name

- WaterWord.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawWord.java** (*cf.* **fr.umlv.drawer**).

- Cette classe ***implements*** l'interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter le mot *Water* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur

l’icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l’icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ WallWord.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawWord.java** (*cf.* **fr.umlv.drawer**).

- Cette classe ***implements*** l’interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter le mot *Wall* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l’icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l’icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ SkullWord.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawWord.java** (*cf.* **fr.umlv.drawer**).

- Cette classe ***implements*** l'interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter le mot *Skull* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l'icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ RockWord.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawWord.java** (*cf.* **fr.umlv.drawer**).

- Cette classe ***implements*** l'interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter le mot *Rock* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**
- ***Color*** *getColor ()*: obtenir la couleur de l'icône.
- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ FlagWord.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawWord.java** (*cf.* **fr.umlv.drawer**).

- Cette classe ***implements*** l'interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter le mot *Flag* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l'icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ BabaWord.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawWord.java** (*cf.* **fr.umlv.drawer**).

- Cette classe ***implements*** l'interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter le mot *Baba* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l'icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

fr.umlv.drawer.word.operation

- IsWord.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawWord.java** (*cf.* **fr.umlv.drawer**).

- Cette classe ***implements*** l'interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter le mot *Is* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur

l’icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l’icône.
- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

fr.umlv.drawer.word.propriety

➢ DefeatWord.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawWord.java** (*cf.* **fr.umlv.drawer**).

- Cette classe ***implements*** l’interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter le mot *Defeat* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l’icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l'icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ PushWord.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawWord.java** (*cf.* **fr.umlv.drawer**).

- Cette classe ***implements*** l'interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter le mot *Push* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l'icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ SinkWord.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawWord.java** (*cf*. **fr.umlv.drawer**).

- Cette classe ***implements*** l'interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter le mot *Sink* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l'icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ StopWord.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawWord.java** (*cf*. **fr.umlv.drawer**).

- Cette classe ***implements*** l'interface **Drawer.java**

(*cf.* **fr.umlv.drawer**)

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter le mot *Stop* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l'icône **;**
- ***Color*** *getColor ()*: obtenir la couleur de l'icône.
- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ WinWord.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawWord.java** (*cf.* **fr.umlv.drawer**).

- Cette classe ***implements*** l'interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter le mot *Win* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur

l’icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l’icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

➢ YouWord.java

Type : *class*

Champs :

- Cette classe ***extends*** la classe **DrawWord.java** (*cf*. **fr.umlv.drawer**).

- Cette classe ***implements*** l’interface **Drawer.java** (*cf.* **fr.umlv.drawer**)

- Le constructeur fait **seulement** appel au super-constructeur.

Description : permet de représenter le mot *You* sur le plateau.

- ***String*** *toString ()*: obtenir le texte à écrire sur l’icône **;**

- ***Color*** *getColor ()*: obtenir la couleur de l’icône.

- ***String*** *toTextFormat ()*: obtenir le format du texte à écrire sur un fichier (*cf.* Sauvegarde/Chargement.)

fr.umlv.file

LoadLevel.java

Type : *final class*

Description : permet de charger un niveau à partir de la lecture d'un fichier. Ce dernier respectera, pour chaque ligne, le format suivant :

[COORDONNÉE X] [COORDONNÉE Y] [DRAWER] [ATTRIBUTE]

Cette classe est un conteneur de méthode.

- ***private static final String** FIELD SEPARATOR :* le symbole de séparation des éléments fichier texte.
- ***static Attribute** instantAttribute (**String** attribute) :* instancier le bon attribut à partir d'une chaîne de caractères.
- ***static Items** parseLine (**String** line) :* analyser une ligne du fichier courant.

- ***static*** *List<**Items**> readFromFile(**BufferedReader** reader)* **:** obtenir et trier toutes les lignes du fichier courant.

➢ SaveLevel.java

Type : *final class*

Description : permet de sauvegarder l'état de la partie courante dans un fichier. L'écriture respectera pour chaque ligne le format suivant :

[COORDONNÉE X] [COORDONNÉE Y] [DRAWER] [ATTRIBUTE]

Cette classe est un conteneur de méthodes.

- ***static void*** *saveLevel (List<**Items**> items,* ***BufferedWriter*** *writer)* **:** écrire l'état du plateau dans le fichier courant.

fr.umlv.word

Word.java

Type : *Interface*

Description : permet de représenter l'intersection des classes **Name.java**, **Operation.java**, **Propriety.java**. En effet, un **mot** ne possède pas les mêmes caractéristiques en fonction de sa **catégorie**. Ainsi :

- ***static final int*** *distance* : le nombre de pixels pour un déplacement sur le plateau **;**
- ***boolean*** *isAdjacentTo (**Items** item)* : le placement des **mots** sur le plateau qui ont le droit d'être adjacent au mot courant.
- ***Category*** *getCategory ()* : obtenir la **catégorie** du **mot** courant.

Category.java

Type : *Enumeration*

Description : permet de représenter les différentes catégories de **mots** qui composent une **règle**.

1. ***NAME*** : les noms ;
2. ***OPERATION*** : les opérations ;
3. ***ATTRIBUTE*** : les propriétés.

➢ Name.java

Type : *Class*

Description : permet de représenter un **nom**.

Champs :

- Cette classe **extends** la classe **Items.java** (*cf.* **fr.umlv.demo**).

- Cette classe **implements** l'interface **Word.java** (*cf.* **fr.umlv.word**)

  - ***Category*** *category* : représente la catégorie du **mot ;**

  - ***private static*** ArrayList<***String***> *names* : représente tout les noms existants et autorisés.

➢ Operation.java

Type : *Class*

Description : permet de représenter une **operation**.

Champs :

- Cette classe ***extends*** la classe **Items.java** (*cf.* **fr.umlv.demo**).

- Cette classe ***implements*** l'interface **Word.java** (*cf.* **fr.umlv.word**)

  - ***Category*** *category* : représente la catégorie du **mot ;**

  - ***private static*** ArrayList<***String***> *operations* : représente toutes les opérations existantes et autorisées.

➢ Propriety.java

Type : *Class*

Description : permet de représenter une **propriété**.

Champs :

- Cette classe ***extends*** la classe **Items.java** (*cf.* **fr.umlv.demo**).

- Cette classe ***implements*** l'interface **Word.java** (*cf.* **fr.umlv.word**)

  - ***Category*** *category* : représente la catégorie du **mot ;**

  - ***private static*** ArrayList<***String***> *proprieties* : représente toutes les propriétés existantes et autorisées.

- Rule.java

Type : *Class*

Description : permet de représenter une **règle**.

Champs :

Une **règle** est composée respectivement : d'un **nom**, d'un **opérateur** et d'une **propriété**. Ainsi :

- ***Items** name* : la représentation d'un **nom** sur le plateau;
- ***Items** operation* : la représentation d'une **operation** sur le plateau;
- ***Items** propriety* : la représentation d'une **propriété** sur le plateau.