The God Game Documentation

Version 0.1

Ho33e5, Goldenf0x, Superboloss

18 May 2014

Table des matières

The God Game (ou *LE jeu*) est une implémentation du jeu de la vie de John Conway. Il s'agit d'un automate cellulaire créé en 1970.

Le principe est que le monde est divisé en une grille de cellules vivantes ou mortes. Les règles pour passer d'une génération à celle d'après sont les suivantes :
– Si une cellule morte a 3 voisines vivantes, elle naît à la génération d'après.
– Si une cellule vivante a 2 ou 3 voisines vivantes, elle reste en vie (sinon il y a sur-population ou sous-population).
On peut représenter les règles de Life par B3/S23 ('Born' et 'Stay alive'). D'autres variantes existent, comme HighLife (B36/S23) ou Seeds (B2/S).

CHAPITRE 1

Le projet

Nous sommes Peio *Ho33e5* BORTHELLE, Jean-Baptiste *Goldenf0x* GIRARD et Léo *Superboloss* TUNON-EMBARECK (lycée Edgar Quinet, Bourg-en-Bresse). Nous avons lancé ce projet dans le cadre de la spécialité ISN (Informatique et Science du Numérique) en terminale S.

Ce projet est écrit en python et utilise mercurial pour le versioning (herbergé sur Bitbucket)

1.1 Plan de route

- Écrire la classe `Dieu` qui gère l'évolution des cellules.
- Écrire l'interface graphique de base (juste l'évolution d'un état initial).
- Créer un bouton qui modifie la vitesse. Un bouton start/stop et avancer d'une génération.
- Créer un bouton zoom.
- Ajouter un éditeur d'état initial (gérer la souris, éventuellement un copier/coller).
- Gérer l'export et l'import des états initiaux (fichiers MCell, Life, RLE...).
- Déjà ici ça sera bien, mais des évolutions sont encore possible, comme l'export de videos...

1.2 Répartition des taches

Ho33e5 Interface graphique en général. Coordination (mise en place du repository mercurial, packaging de l'application...).

Goldenf0x Partie algorithmique : calcul d'une génération suivante. Il faut modifier *Dieu* pour qu'une cellule voulant s'échapper réapparaisse en bas (mode 'snake'). Créer un mode 'infini' où les cellules qui s'échappent sont toujours visibles si on dézoome.

Superboloss Gérer l'import et l'export des fichiers MCell, Life et RLE (voir le site de MCell pour les spécifications).

CHAPITRE 2

Contenu

2.1 Readme

2.1.1 LE jeu

The God Game (*LE jeu*) est un remake du jeu de la vie de Conway.

Le code source est hébergé sur bitbucket et la documentation sur readthedocs.

Installation

LE jeu a pour dépendances pyglet et cocos2D. Tout est installable de manière automatisée avec pip :

```
pip install hg+https://bitbucket.org/jambonbeurreteam/le-jeu/
```

Bon jeu !

2.2 Documentation de l'API

LE jeu est installable comme une bibliothèque et un script de démarrage. Voici donc une description de l'api (Application Programming Interface) que nous avons écrite. Tous modules sont regroupés dans le package `lejeu` qui contient donc :
– `lejeu.dieu` pour l'algorithme d'évolution du jeu de la vie.
– `lejeu.fenetre` pour l'interface graphique.
– `lejeu.sauvegarde` pour l'import et l'export des fichiers d'états initiaux.
Chaque module a été écrit principalement par une personne (voir *repartition*), mais tout le monde a participé au débuguage et aux details de tous les fichiers.

2.2.1 Package `lejeu`

Module `lejeu.dieu`

class `lejeu.dieu.`**`Dieu`**(*etat*)

Bases : `builtins.object`

Classe gérant la vie et la mort des cellules.
Initialise la classe Dieu.

param etat Liste de coordonnée des cellules vivantes.

type etat liste

`Dieu.`**`generation_suivante`**(*mode='snake'*)

Calcule la prochaine génération avec le bon mode assigné.

`Dieu.`**`generation_suivante_mur`**()

Applique l'algorithme d'évolution en mode `mur`.
–Fait une copie de `self.etat`.
–Prends tout les voisins de chaque cellules et applique les deux règles.
–Met à jour `self.etat`.

`Dieu.`**`generation_suivante_snake`**()

Applique l'algorithme d'évolution en mode `snake`.
–Fait une copie de `self.etat`.
–Prends tout les voisins de chaque cellules et applique les deux règles.
–Met à jour `self.etat`.

`Dieu.`**`print_etat`**()

Affiche l'etat de manière comprehensible sur la console (debug).

class `lejeu.dieu.`**`SuperDieu`**(*etat*, *size=None*, *mode='infini'*, *regle=((3,), (2, 3))*)

Bases : `builtins.object`

Classe optimisée permettant l'évolution des cellules.

Initialise la classe SuperDieu

param etat Set de coordonnée des cellules vivantes

type etat set

param size couple de nombre définissant la largeur et la hauteur du rectangle défini au préalable

type size tuple

param mode mode de jeu influant sur le calcule des voisins

type mode str

param regle defini les règles

type regle tuple

`SuperDieu.`**`coord_voisins`**(*c*)

Calcule les coordonnées des voisins d'une cellule.

Paramètres **c** (*Tuple.*) – Couple de coordonnées.

Retourne Les voisins de c.

Type retourné Set.

`SuperDieu.`**`doit_mourir`**(*c*)

Défini les cellules qui mourront à la prochaine génération.

Paramètres **c** (*tuple*) – Couple de coordonnées.

Retourne Doit-elle mourir ?

Type retourné bool

`SuperDieu.`**`doit_naitre`**(*c*)

Défini les cellules qui naîtront à la prochaine génération.

Paramètres **c** (*tuple*) – Couple de coordonnées.

Retourne Doit-elle survivre ?

Type retourné bool

`SuperDieu.`**`generation_suivante`**()

Applique l'algorithme d'évolution en mode `infini`.
–Calcule les coordonnées des cellules nées et mortes suivants la règle du JEU.
–Met à jour `self.n_voisines`, `self.mortes_actives`, `self.vivantes`.

Retourne Les cellules nées et mortes.

Type retourné bool

Module `lejeu.fenetre`

Module gérant l'interface graphique

Ce module définie les classes utilisées pour toute l'interface graphique (menus, édition, import de fichiers, évolution des cellules). Il y a également une fonction `main()` qui démarre l'application.

class `lejeu.fenetre.`**`CoucheAide`**

Bases : `builtins.Layer`

Couche de texte de l'aide.

`CoucheAide.`**`is_event_handler`** = **True**

`CoucheAide.`**`on_mouse_press`**(*x*, *y*, *buttons*, *modifiers*)

Revient au menu principal si on clique.

class `lejeu.fenetre.`**`CoucheCellules`**(*etat*, *cell_dim=(10.0, 10.0)*, *theme=((255, 255, 255, 255), (0, 0, 0, 255))*)

Bases : `builtins.Layer`

Couche capable d'afficher des cellules.

Cette couche permet de se déplacer dans le monde en cliquant/glissant avec le clic droit et de zommer avec la molette.

param etat État à utiliser pour initialiser `lejeu.dieu.SuperDieu`.

type etat set

param cell_dim Hauteur et largeur d'une cellule (en pixels).

type cell_dim tuple

`CoucheCellules.`**`dessine_cellules`**(*nees*, *mortes*)

Met à jour la liste de cellules à dessiner.

`CoucheCellules.`**`draw`**()

Dessine les cellules vivantes sur la couche.

`CoucheCellules.`**`is_event_handler`** = **True**

`CoucheCellules` gère les évenement de la fenêtre.

`CoucheCellules.`**`on_mouse_drag`**(*x*, *y*, *dx*, *dy*, *buttons*, *modifiers*)

Déplacer les cellules avec la souris.

Paramètres

- **x** (*int*) – Abscisse de la souris.
- **y** (*int*) – Ordonnée de la souris.
- **dx** (*int*) – Déplacement en abscisse de la souris.
- **dy** (*int*) – Déplacement en ordonnée de la souris.
- **buttons** (*int*) – Combinaison des codes des boutons de la souris enfoncés.

`CoucheCellules.`**`on_mouse_scroll`**(*x*, *y*, *scroll_x*, *scroll_y*)

Zoom centré sur la souris.

Paramètres

- **x** (*int*) – Abscisse de la souris.
- **y** (*int*) – Ordonnée de la souris.
- **scroll_x** (*int*) – Scroll horizontal de la souris (rare).
- **scroll_y** (*int*) – Scroll vertical ('clics' de molette).

`CoucheCellules.`**`update`**(*dt*)

Met à jour la fenêtre si besoin.

Paramètres **dt** (*float*) – Durée depuis le dernier appel (environ 1/60sec).

`CoucheCellules.`**`vertexs_cellule`**(*c*)

Calcule les vertexs d'une cellule.

Paramètres **c** (*tuple*) – Coordonnées de la cellule.

Retourne attributs formatées des vertexes (voir vertex attributes)

Type retourné tuple

class lejeu.fenetre.**CoucheCredits**

Bases : builtins.Layer

Couche de texte des crédits.

CoucheCredits.**is_event_handler = True**

CoucheCredits.**on_mouse_press**(*x*, *y*, *buttons*, *modifiers*)

Revient au menu principal si on clique.

class lejeu.fenetre.**CoucheEdition**(***kwargs*)

Bases : lejeu.fenetre.CoucheCellules

Couche permettant l'édition d'un état initial.

CoucheEdition.**on_key_release**(*symbol*, *modifiers*)

Stoppe/redémarre l'évolution en appuyant sur <space>.

CoucheEdition.**on_mouse_drag**(*x*, *y*, *dx*, *dy*, *buttons*, *modifiers*)

Dessine des cellules (efface avec <ctrl>).

Paramètres

- **x** (*int*) – Abscisse de la souris.
- **y** (*int*) – Ordonnée de la souris.
- **dx** (*int*) – Déplacement en abscisse de la souris.
- **dy** (*int*) – Déplacement en ordonnée de la souris.
- **button** (*int*) – Code de la touche appuyée.
- **modifiers** (*int*) – Combinaison des codes des touches spéciales.

CoucheEdition.**on_mouse_press**(*x*, *y*, *button*, *modifiers*)

Dessine des cellules (efface avec <ctrl>).

Paramètres

- **x** (*int*) – Abscisse de la souris.
- **y** (*int*) – Ordonnée de la souris.
- **button** (*int*) – Code de la touche appuyée.
- **modifiers** (*int*) – Combinaison des codes des touches spéciales.

class lejeu.fenetre.**CoucheEvolution**(*etat*, *vitesse_evo=0.25*, ***kwargs*)

Bases : lejeu.fenetre.CoucheCellules

Couche permettant l'évolution des cellules (scène principale).

param etat État à utiliser pour initialiser lejeu.dieu.SuperDieu.

type etat set de coordonnées des cellules vivantes

param cell_dim Hauteur et largeur d'une cellule (en pixels).

type cell_dim tuple d'entier

param vitesse_evo Nombre de secondes pour qu'une nouvelle génération apparaisse.

type vitesse_evo int ou float

CoucheEvolution.**on_key_press**(*symbol*, *modifiers*)

Stoppe l'évolution avec la touche espace.

CoucheEvolution.**update**(*dt*)

Met à jour une nouvelle génération s'il le faut.

Paramètres **dt** (*float*) – Durée en seconde depuis le dernier appel.

class lejeu.fenetre.**MenuImport**

Bases : builtins.Menu

Menu d'import de fichier.

`MenuImport.`**`on_import`**`()`
Essaye d'importer le fichier demandé.

`MenuImport.`**`on_retour`**`()`
Revient au menu principal.

`MenuImport.`**`on_value`**`(`*value*`)`
Met a jour la valeur entrée.

class `lejeu.fenetre.`**`MenuPrincipal`**

Bases : `builtins.Menu`

Couche du menu principal.

`MenuPrincipal.`**`on_aide`**`()`
Lance la couche `CoucheAide`.

`MenuPrincipal.`**`on_create`**`()`
Lance la couche `CoucheEdition`.

`MenuPrincipal.`**`on_credits`**`()`
Lance la couche `CoucheCredits`.

`MenuPrincipal.`**`on_import`**`()`
Lance le menu `MenuImport`.

`MenuPrincipal.`**`on_quit`**`()`

`lejeu.fenetre.`**`main`**`(`*etat*`)`
Démarre l'interface graphique

Module `lejeu.sauvegarde`

exception `lejeu.sauvegarde.`**`LifeError`**

Bases : `builtins.ValueError`

class de base des erreurs life

`lejeu.sauvegarde.`**`exporter_life_106`**`(`*fichier*, *etat*`)`
Export des fichiers ".lif" en version 1.06.

L'export ce fera en life 1.06 car c'est le plus facile a exporter puisque que pour coder les coordonnées des cellules on utilise les coordonnées de cellules vivantes. Transformation des données utilisées dans le code en caracters écrit en format life 1.06.

Paramètres **fichier** (*str*) – Chemin du fichier à importer.

`lejeu.sauvegarde.`**`importer_MCell`**`(`*fichier*`)`
Import Mcell en stand by.

`lejeu.sauvegarde.`**`importer_RLE`**`(`*fichier*`)`
Import des fichiers RLE.

Le fichier est comprimé en RLE sur le format life de versions .05 et/ou .06. Utilisation de l'expression regulière pour trouver et sortir les caractères voulus. Transformation des données de caractères en données de position des cellules.

Paramètres **fichier** (*str*) – Chemin du fichier à importer.

`lejeu.sauvegarde.`**`importer_fichier`**`(`*fichier*`)`
Import de tout les fichiers reconnus.

Lecture du format de fichier. Transfère dans une fonction spécifique. En cas de format inconnu affichage d'un message d'erreur.

Paramètres **fichier** (*str*) – Chemin du fichier à importer.

lejeu.sauvegarde.**importer_life**(*fichier*)
Import des fichiers ".lif".

La première ligne du fichier life donne la version. Trie et transfère les versions connues à une fonction spécifique. En cas de version inconnue affichage d'un message d'erreur.

Paramètres **fichier** (*str*) – Chemin du fichier à importer.

lejeu.sauvegarde.**importer_life_105**(*fichier*)
Import des fichiers ".lif" version 1.05.

Le life 1.05 est une représentation de la grille des cellules en caractère. Recherche des caractères correspondant à une cellule vivante et récuperation des coordonnées des cellules vivantes.

Paramètres **fichier** (*str*) – Chemin du fichier à importer.

lejeu.sauvegarde.**importer_life_106**(*fichier*)
Import des fichiers ".lif" version 1.06.

Le life 1.06 stoque uniquement les coordonnées de cellules vivantes. Transformation des caractères de coordonnées en coordonnées.

Paramètres **fichier** (*str*) – Chemin du fichier à importer.

2.3 License

CHAPITRE 3

Index & tables

– *genindex*
– *modindex*
– *search*

l