HomePlans - Rapport
INFO-F-307

Titouan Christophe
Pierre Gérard
Florentin Hennecker
Walter Moulart
Bruno Rocha Pereira
Julian Schembri

31 octobre 2014

Table des matières

Chapitre 1

Itération 1

1.1 Introduction

Nous avons décidé de développer l'histoire 1 proposée par le client lors de cette phase. Ce choix a été fait dans le but de fournir le plus vite possible un socle pour les fonctionnalités à suivre, et se présentait logiquement comme le seul choix possible.

Vous pourrez lire dans la suite de ce chapitre la décomposition de cette histoire en top-level tasks, ainsi que la description des choix qui ont été faits quant aux librairies que l'équipe a décidé d'utiliser.

Les PV des réunions officielles qui ont eu lieu jusqu'ici peuvent être trouvés en annexe.

1.2 Histoires utilisateur

1.2.1 Enregistrement d'un projet

Durée estimée : 20h | Difficulté estimée : 2/3 | Assigné à : Titouan

Les projets d'architecture doivent être enregistrables dans des fichiers, et il doit être possible de charger un projet depuis un fichier. Différents éléments doivent être enregistrés, comme la configuration du projet (nom, auteur, ...), ainsi que des éléments affichables (polyèdres pour le moment, éventuellement les tags par la suite).

Les projets doivent être enregistrés séparément et sélectionnables dans un navigateur de fichiers pour ouverture ou enregistrement.

1.2.2 Interface graphique

Durée estimée : 30h | Difficulté estimée : 1/3 | Assigné à : Julian et Pierre

L'interface graphique sera composé d'un menu, d'une barre d'outils et d'un écran splitté en deux. Sur cet écran splitté il y aura à gauche les objet utilisés et a droite le

rendu 2D et 3D.

1.2.3 Affichage du monde 2D et 3D

Durée estimée : 10h | Difficulté estimée : 1/3 | Assigné à : Florentin, Bruno et Walter

L'utilisateur devra pouvoir choisir d'afficher le plan d'architecture autant en 3D qu'en 2D. Chaque pièce pourra avoir une forme différente et n'aura pas forcément la forme d'un rectangle.

1.2.4 Navigation dans le monde

Durée estimée : 5h | Difficulté estimée : 1/3 | Assigné à : Bruno et Julian

Dans la vue 3D, l'utilisateur devra avoir le contrôle de la caméra, que ce soit pour zoomer ou déplacer la caméra sur les cotés, et ce, aussi bien grâce à la souris que grâce aux flèches.

1.2.5 Modification de la géométrie d'une pièce

Durée estimée : 40h | Difficulté estimée : 3/3 | Assigné à : Walter, Titouan, Bruno, Florentin

L'utilisateur a accès à une vue d'édition lui permettant de rajouter ou éditer des murs et des étages.

Tous les points du "sol" sont à la même hauteur, et à chacun d'eux est associée une élévation.

La hauteur de l'étage est déterminée par son point le plus haut. Le sol de l'étage suivant est égal à la hauteur du point le plus haut de l'étage précédent

1.2.6 Création d'un projet de demo

Durée estimée : 5h | Difficulté estimée : 1/3 | Assigné à : Walter et Julian

Lorsque l'utilisateur lance pour la première fois le programme EA3D, un projet de démonstration complet s'ouvre, pour lui montrer tout ce qu'il est possible de faire.

1.2.7 Enregistrement automatique d'un projet

Durée estimée : 2h | Difficulté estimée : 1/3 | Assigné à : Titouan et Pierre

Le projet sur lequel travaille l'utilisateur est sauvé automatiquement sur le disque. L'utilisateur peut aussi enregistrer une copie du projet, auquel cas un nouveau fichier de projet est créé, et devient le projet courant.

Par défaut, lorsque l'utilisateur ouvre le programme, le dernier projet utilisé lui est présenté.

1.2.8 Intégration de la vue 3D dans la GUI

Durée estimée : 10h | Difficulté estimée : 2/3 | Assigné à : Pierre

Il faut intégrer jmonkey qui s'occupe de la 3d dans swing qui s'occupe de l'interface graphique

http ://hub.jmonkeyengine.org/wiki/doku.php/jme3 :advanced :swing_canvas

1.3 Librairies introduites

1.3.1 Enregistrement du projet

Utilisation de SQLite pour enregistrer les projets, à l'aide de ORMlite, et ses dépendances (ormlite-jdbc, ormlite-core, sqlite-jdbc)

- sqlite a les avantages d'une DB relationnelle, mais enregistre dans des fichiers : facilité de déplacement des projets, pas besoin de serveur
- sqlite permet d'avoir une DB en mémoire vive : pratique pour les tests
- sqlite est utilisable dans de nombreux autres langages : augmente la productivité en permettant d'éditer les fichiers de projets dans le langage préféré de chaque dev
- ORM facilite l'accès aux données (Design pattern DataMapper http ://martinfowler.com/eaaCatalog/dataMapper.html)
- ORMLite permet d'utiliser plusieurs types de bases de données différentes à travers JDBC. Possibilité donc d'utiliser autre chose qu'SQLite si nécessaire.

1.3.2 Librairie GUI

Les requirements de l'interface graphique sont les suivants

- Afficher une fenêtre esthétique
- Mettre des menus / boutons
- Incorporer une vue 2D/3D
- Compatible avec la lib 3D choisie

Les différentes possibilités de GUI sont :

— AWT (Abstract Window Toolkit)
— Swing
— SWT (Standard Widget Toolkit)
— JavaFX
— Apache Pivot
— Qt Jambi

Swing

Nous avons retenu **Swing**.
— Customisable : Oui
— Léger : Non
— Cross-Platform : Oui
— Documentation : Complet
— Faiblement couplé
— Suit le design pattern MVC
— Compatible avec jMonkeyEngine

Toutes ces raisons font qu'on utilisera Swing comme librairie pour construire notre interface graphique. Voici un aperçu des autres libraires éligibles, et les raisons pour lesquelles elles ont été refusées.

AWT

— Customisable : Non
— Léger : Oui
— Cross-Platform : Oui
— Documentation : Complet
— AWT utilise les objets du système
— Swing hérite des objets de AWT

On n'utilisera pas AWT car il n'est pas assez customisable.

SWT

— Customisable : Non
— Léger : Oui
— Cross-Platform : Oui
— Documentation : Très complet
— Développé par l'équipe d'Eclipse
— Se place un peu comme premier concurrent de Swing

On n'utilisera pas SWT car il n'est pas assez customisable.

Java FX

On n'utilisera pas Java FX car ils se sont spécialisés dans les interfaces d'application web et ce n'est pas forcément utile ici.

Apache Pivot

On n'utilisera pas Apache Pivot pour les mêmes raisons que Java FX.

Qt Jambi

— Customisable : Oui
— Léger : Non
— Cross-Platform : Oui mais limité à QT4.6
— Documentation : Mauvaise
— Intégré à Eclipse
— Assez complet
— Accès a une database sql incluse

Inconvénients :

— Pas de doc sur le site QtJambi
— Compliqué pour l'intégration de la 3D . Il existe Qt3D mais on s'est dirigé vers Jmonkey
— Bloqué a la version 4.6

1.3.3 Librairie 3D

Nous choisissons d'utiliser jMonkeyEngine comme librairie 3D pour ces raisons :

— Elle est sous license BSD
— Une des interfaces les plus high level
— Community driven
— Développement encore actif
— Documentation extensive
— Engine complet
— Refonte complête de JME2 pour le mieux
— Intégration facile dans un GUI swing : http ://hub.jmonkeyengine.org/wiki/doku.php/jme3 :advanced :swing_canvas

Justification de l'écartement d'autres librairies :

JOGL :

— Utilise SWT comme système de fenêtrage

GL4Java :

— Vieux et obsolète

Java3D :

— Abandonné

Ardor3D :

— Prise en main difficile et risque de requérir beaucoup plus de temps

Annexe A

PV des réunions

A.1 Réunion Kick-off

22 octobre 2014

Présents : Walter, Pierre, Bruno, Titouan, Julian, Florentin

Ordre du jour

— Partage d'expérience
— Itération 1 : objectifs & planning
— Répartition des tâches
— Réunions hebdomadaires
— Conventions

Partage d'expérience

— Communication et honnêteté super importantes
— Savoir tout justifier dans son code
— Toucher à un maximum de parties du code, au moins connaître leur architecture
— Tester un maximum (TDD pour ce projet en particulier)
— Définir les objectifs de chacun clairement
— Coder en groupe un maximum, physiquement, pourquoi pas au hackerspace

Itération 1 : objectifs & planning

On va commencer par l'histoire no 1 pour avoir une "vraie" base. Les top-level tasks seront fixées plus tard mais on peut déjà discerner les suivantes :
— Enregistrement d'un projet
— GUI
— Affichage du monde 2D/3D
— Navigation dans le monde 2D/3D
— Modification de la géométrie de la pièce
— Création d'un default/demo project

Répartition des tâches

Pour l'échéance **Gestion de Projet 1**, il faudra fixer, développer et assigner les top-level tasks. Il faudra aussi se renseigner sur les librairires à utiliser.

— Librairie 3D : Bruno et Florentin
— Librairie GUI : Julian
— Enregistrement de fichiers 3D : Titou

Réunions hebdomadaires

Il a été décidé, outre le fait de se retrouver un maximum pour travailler ensemble, de se retrouver de manière hebdomadaire obligatoirement, pour tout mettre au point, les jeudis à 11h au hackerspace.

Conventions

Le groupe a accepté à l'unanimité l'utilisation de plusieurs conventions :

— Java 1.6
— camelCase, _ au début de variables/fonctions privées, majuscule au début des noms de classes
— Petits commits (quelques dizaines de lignes maximum)
— 1 feature = 1 branche
— Commentaires de commit commençant par "[fix]", "[enh]" (enhance), "[add]", "[rem]" (remove), "[ref]" (refactoring), "[test]"
— Utilisation des tags git
— Tout ce qui est pushé sur le remote doit être documenté et testé
— Utilisation de jUnit et de Javadoc
— Utilisation massive (tant que possible) des exceptions et des assertions
— Utilisation de DIA pour les diagrammes UML
— Le choix d'une lib se fait à au moins 3 personnes du groupe, et est motivé et expliqué
— Pré-échéances 1h avant l'échéance officielle. Plus rien n'est rajouté après cette pré-échéance et on teste le fonctionnement du produit