INSTITUT UNIVERSITAIRE DE TECHNOLOGIE

Département INFORMATIQUE

(Computer Science Department – Nantes Institute of Technology)

Rapport de Projet Technologique

2015-2016

SweetUP3D

Projet de 2ème année

Réalisé par : **- Cédric Berland**
- Matthieu Fournier
- Jean-Baptiste Lacour

Client : **IUT de Nantes**

Encadrés par : **Sébastien Cannet**
Solen Quiniou

du 05/10/2015 au 18/01/2016

Agenda des phases du travail

Phases	Dates	Commentaires
Réunion de démarrage (avec encadrant)	06 / 10 / 2015	Un projet en deux parties : - Plug-in SweetHome3D -> Sketchup - Serveur local ? pour faire communiquer Scratch et SweetHome3D
Validation du plan de travail proposé par les étudiants	06 / 10 /2015	
Réunion 2	16 / 10 / 2015	Focus sur le plug-in SweetHome3D -> SketchUp
Réunion 3	06 / 01 / 2016	Recherche approfondit sur les différents format de fichier qui peuvent permettre une communication (-> DAE)
. . .		
Rendu de la première version du rapport à l'encadrant	18 / 01 / 2016	
Récupération du rapport annoté par l'encadrant		
. . .		

Présentation de Projet :

Résumé :

Ce projet consiste en réalité en deux sous-projets distincts bien que leurs thèmes se rejoignent, puisqu'il s'agit à chaque fois de communication entre deux logiciels différents.

Le premier sous-projet nécessite de développer un plugin pour Sweet Home 3D (que l'on appellera SH3D par la suite), un logiciel de modélisation 3D, qui permet d'exporter les objets du logiciel dans un format compatible avec le logiciel SketchUp, un autre logiciel de modélisation 3D. En effet, les possibilités actuelles qui consistent simplement à permettre l'importation dans SketchUp des fichiers issus de SH3D sont payantes et donnent un résultat décevant (objets lourds et défauts visuels).

Le second sous-projet consiste à utiliser Scratch, un logiciel destiné aux enfants et aux non-initiés à la programmation, et à créer une communication ouverte avec SH3D permettant alors de modifier des attributs des objets dans ce dernier logiciel.

Abstract :

This project consists of two separate sub-projects that have the same themes, as it is each time communication between two different software.

The first sub-project needs to develop a plugin for Sweet Home 3D (which will be called SH3D thereafter), a 3D modelling software that allows us to export software objects in a format compatible with the SketchUp software (another 3D modelling software). Indeed, the current possibilities consisting to permit the import SketchUp files from SH3D are paid and give a disappointing result (heavy objects and visual defects).

The second sub-project is to use Scratch, a software intended for children and uninitiated to programming, and creating open communication with SH3D then to change the attributes of the objects in this latest software.

INSTITUT UNIVERSITAIRE DE TECHNOLOGIE

3, rue du Maréchal Joffre 44041 cedex 01 - http://www.iut-nantes.univ-nantes.fr

1.Table des matières

2. Introduction

a. Sweet Home 3D

SH3D est un logiciel de modélisation 3D créé en Java en 2005 par une seule personne, un français dénommé Emmanuel Puybaret, et disponible en licence GNU sur SourceForge. Il donne la possibilité à un utilisateur de créer une maison et surtout son intérieur, avec précision et avec un rendu assez réaliste, notamment en ce qui concerne la gestion des lumières. SH3D est le seul logiciel d'aménagement intérieur open source. Installé plus de 10 000 fois par jour d'après le graphique des téléchargements SourceForge, il est encore régulièrement mis à jour bien qu'il ait récemment fêté son 10ème anniversaire.

Annexe 01 : Rendu maison réalisé avec Sweet Home 3D

Annexe 02 : L'interface de Sweet Home 3D

b. SketchUp

SketchUp est un logiciel de modélisation 3D développé en Ruby, il n'est pas libre et n'est donc pas disponible sur SourceForge avec la licence GNU. Initialement édité en 2000 par la société @LastSoftware, il a ensuite été racheté par Google en mars 2006 avant d'être de nouveau racheté, cette fois ci en 2012, par la société Trimble. Le rachat par google a permis de mettre plus en avant l'intégration des objets de SketchUp dans Google Earth et Google Maps, bien qu'elle était déjà possible avant 2006. Très populaire, ce logiciel est également assez fermé, si bien que la plupart des plugins sont payants. SketchUp est un logiciel orienté vers l'architecture et permet de modéliser simplement un environnement à grande échelle. Ainsi, au contraire de SH3D qui se concentre principalement sur la modélisation d'intérieurs, SketchUp est plus approprié pour modéliser rapidement plusieurs maisons souvent peu détaillées mais qui, ensemble, permettent de créer facilement une ville, tout en pouvant rajouter quelques éléments de décor tels que des arbres par exemple. Si SketchUp est donc idéal pour la création d'environnement, on se rend rapidement compte qu'il n'est pas fait pour le design intérieur.

Annexe 03 : Exemple utilisation de SketchUp dans Google Maps

Annexe 04 : L'interface SketchUp

c. Scratch

Scratch, dont la version actuelle est la 2.0 ce qui lui vaut parfois l'appellation de Scratch 2.0, est un logiciel éducatif pour apprendre la programmation et possédant une forte popularité dû à son accessibilité, pour les collégiens par exemple, désireux de découvrir le monde du développement informatique. Le logiciel a été développé par des étudiants de la prestigieuse école MIT aux États-Unis. Aujourd'hui, Scratch comporte de plus de 10,5 millions de projets partagés allant de l'animation à la simulation 3D et le nombre d'utilisateurs enregistrés est de 7,5 millions et ne cesse d'augmenter. Le principe du logiciel est que le code est directement inscrit dans la langue maternelle des utilisateurs (une vingtaine de langues européennes sont disponibles) sous forme de briques en couleurs (par exemple les contrôles sont en orange, les variables en rouge, les mouvements en bleu). Pour les professeurs, Scratch est un excellent moyen d'enseigner la programmation du fait de son aspect ludique.

Annexe 05 : Scratch et son système de brique de couleur. A gauche le résultat des instructions à droite.

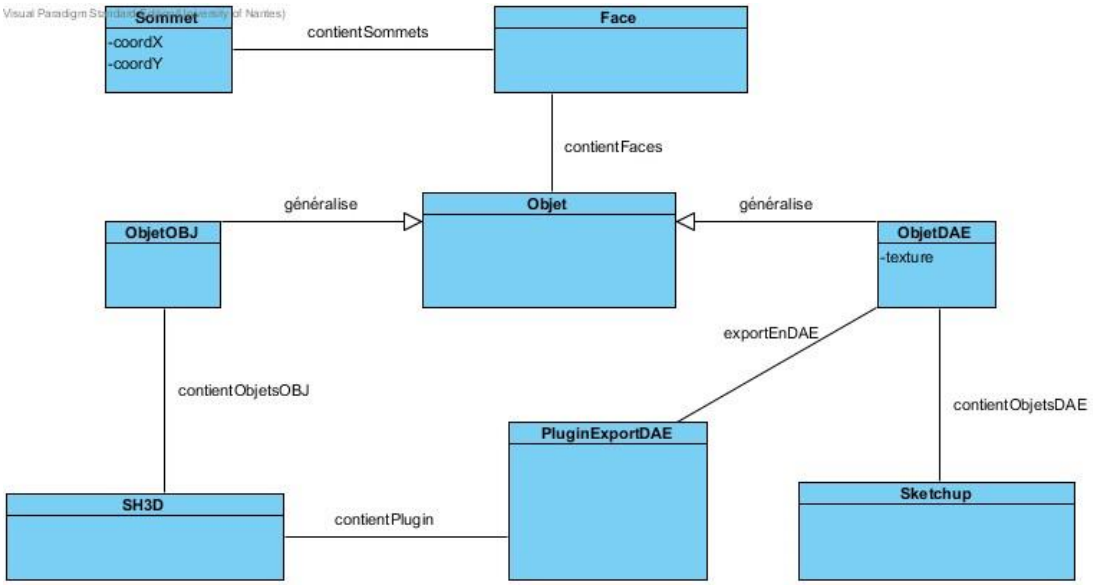
3.SweetUp3D

Le projet SweetUp3D correspond au premier sous-projet et consiste donc à améliorer le transfert de modèles 3D depuis SH3D vers SketchUp.

En effet, comme nous l'avons expliqué précédemment, si SketchUp est très intéressant pour modéliser facilement et rapidement des environnements 3D, il n'est pas adapté à la modélisation d'intérieurs. Cette tâche est plutôt réservée à SH3D qui lui est spécialisé dans l'aménagement. L'idée du projet est donc la suivante : utiliser SH3D pour créer une maison détaillée avec différentes pièces, des meubles et même divers objets puis de pouvoir utiliser cette maison dans SketchUp pour l'intégrer à un environnement plus vaste.

Pour ce faire, l'idée initiale à laquelle nous avons songé consiste à créer un plugin pour SH3D et qui permet d'exporter les objets du logiciel dans un fichier dont le format est compatible avec SketchUp, tout en corrigeant les problèmes à l'origine des défauts visuels après l'importation dans SketchUp. Effectivement, le format par défaut de SH3D est le format OBJ qui ne peut pas être importé dans SketchUp, à moins d'utiliser des plugins payant. Le format par défaut de SketchUp est, lui, le format COLLADA, identifié par l'extension de fichier "DAE". Le plugin SweetUp3D devait donc, initialement, être capable de modifier les objets de SH3D dans le but de retirer les défauts visuels observés dans SketchUp puis d'exporter ces objets en DAE. Nous insistons sur le "initialement" car il se trouve que la partie d'exportation en DAE a été retirée à l'état d'avancement où nous sommes car nous avons depuis pensé à une autre façon de procéder et ainsi seule la partie de modification des objets de SH3D nous intéresse désormais pour le plugin.

Ceci étant, pour mieux appréhender le projet et les différentes relations qui existent entre les éléments concernés, voici un schéma récapitulatif des interactions telles que nous nous les représentions lorsque nous pensions à un plugin permettant l'export en DAE :



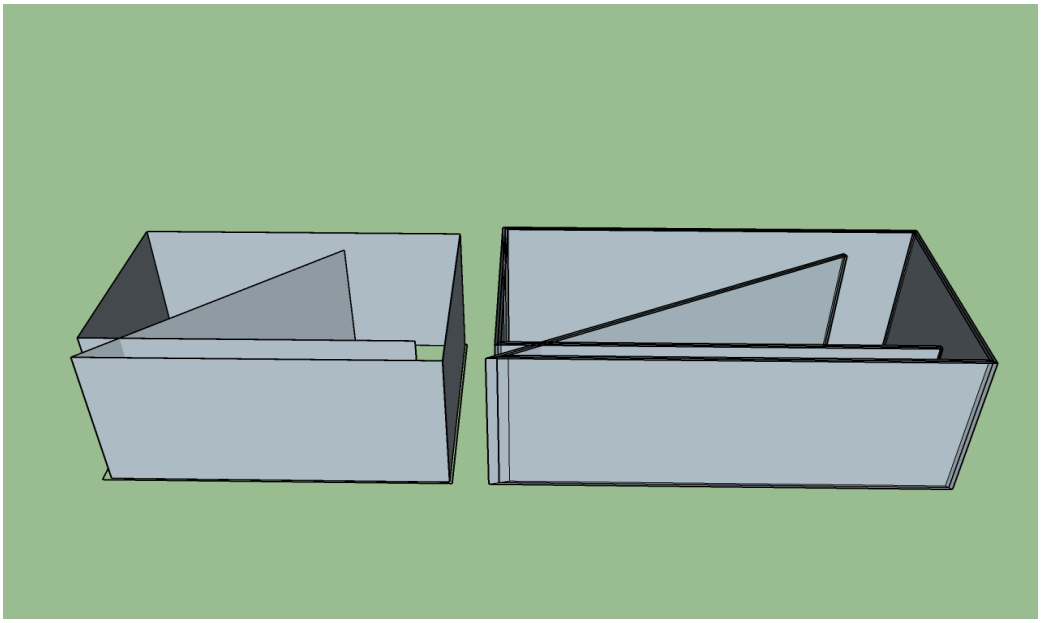
On voit ainsi sur ce schéma que SH3D et SketchUp contiennent tout deux des objets, respectivement des objets en OBJ et des objets en DAE. Ces objets ont des faces, elles-mêmes contenant des sommets qui sont les points qui les constituent. Enfin, et c'est particulièrement ce qui nous intéresse, SH3D contient notre plugin permettant une exportation propre en DAE, c'est-à-dire produisant un fichier et donc un objet DAE compatible totalement avec SketchUp.

Comme nous l'avons expliqué précédemment, une des fonctionnalités, quoi qu'il en soit, primordiales de notre plugin est qu'il soit capable de modifier les objets de SH3D de telle sorte que cela retire les défauts visuels observés après l'importation dans SketchUp.

De ce fait, le plugin doit récupérer l'ensemble des objets présents dans l'environnement 3D du logiciel puis modifier certaines de leurs caractéristiques pour que les modèles conviennent à SketchUp. Ensuite, dans ce nous imaginions au début, pour chaque objet, le plugin aurait dû récupérer les informations le concernant afin de les enregistrer dans un fichier en suivant le formalisme DAE.

En effet, si nous ne nous intéressions qu'à la conversion des fichiers OBJ en fichiers DAE, l'importation fonctionnerait dans SketchUp mais des défauts seraient encore visibles. Pour palier cela, nous avons donc étudié les différences principales entre les deux logiciels et qui peuvent être à l'origine des problèmes. Nous avons compris que là ou SH3D gère l'épaisseur de ses objets, ce n'est pas le cas de SketchUp pour lequel l'épaisseur des modèles n'est pas prise en compte, c'est-à-dire que tous les objets de SketchUp consistent en de simples faces à l'origine de la forme des objets. De ce fait, un mur dans SketchUp n'a aucune épaisseur mais n'est qu'une simple face. Tenter d'importer des objets dotés d'une épaisseur dans SketchUp était donc la source des problèmes.

Nous avons donc cherché une façon de retirer l'épaisseur des objets de SH3D puis nous avons utilisé un convertisseur externe de OBJ vers DAE pour pouvoir tester dans SketchUp et ainsi constater que le problème était résolu. Vous pouvez d'ailleurs observer ci-dessous le résultat obtenu dans SketchUp sans et avec la modification des objets dans SH3D :



Une fois le problème des différences entre SH3D et SketchUp concernant la représentation des objets réglé, il fallait désormais être capable d'exporter les objets OBJ ainsi modifiés en DAE, le format de SketchUp. Du moins cela était-il ce que nous avions initialement prévu. En effet, il existe finalement plusieurs possibilités pour résoudre le problème. Au fil de l'avancée du projet, nous en sommes donc venus à penser à une autre solution, que nous présentons ci-après. Voici décrites les deux solutions que nous avons envisagées :

- L'exportation directe en .dae depuis SH3D

Il s'agit de l'idée initiale du projet car il s'agissait de la plus évidente. Au début, pour tester que nos modifications d'objets dans SH3D permettaient un meilleur résultat dans SketchUp, nous avons utilisé un programme externe pour effectuer la conversion. Bien que cela fonctionne tout à fait, il était préférable que notre plugin le permette lui-même. Cependant, le problème majeur auquel nous avons été vite confrontés a été la nécessité de comprendre parfaitement le format .dae qui est bien plus complexe et fourni que le format OBJ. Une erreur de compréhension aurait ainsi pu empêcher l'importation sur SketchUp, et s'avèrerait très difficile à maintenir. C'est face à cette difficulté que nous nous sommes interrogés sur d'autres solutions.

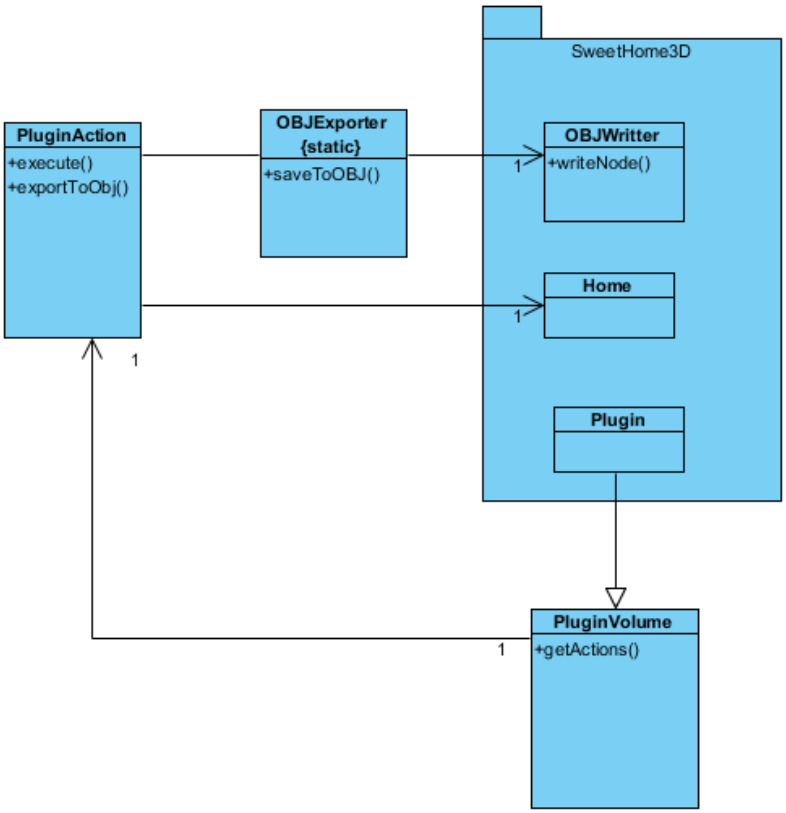
- L'importation du .obj sur SketchUp

Si la conversion du format OBJ vers le format DAE nous a semblé difficile, d'autant que nous n'avons pas trouvé de plugin pour SH3D le faisant déjà, nous avons pensé à faire l'inverse, c'est-à-dire importé de l'OBJ dans SketchUp. Si SketchUp permet lui aussi la création de plugin, la solution d'en développer un nous-mêmes ne nous convenait pas car SketchUp est écrit en Ruby, un langage que nous ne maîtrisons pas et nous aurions dû nous familiariser avec tout le code de SketchUp, en plus de celui de SH3D, et il s'agit d'un travail chronophage. Dès lors, nous avons songé à chercher des plugins pour SketchUp permettant d'importer d'autres formats que ceux disponibles de base. Ainsi, cela nous permettrait d'importer les .obj que produit SH3D directement dans SketchUp. Le problème qui s'est alors posé est que les plugins déjà existant que nous avons trouvé et permettant cette importation sont des plugins nécessitant d'acheter une licence afin de les utiliser au-delà de la période d'essai, ce qui ne nous convient donc pas. En attendant de trouver une solution gratuite fonctionnelle, nous avons déjà fait des essais avec un plugin gratuit durant les 15 premiers jours, FluidImporter, et cela fonctionnait bien.

Au passage, il convient de parler des textures. En effet, le format OBJ ne prend pas en compte les textures, donc il nous a fallu importer également, à l'aide de FluidImporter, le fichier .mlt qui décrit les textures qui sont des images au format .png.

Maintenant que nous travaillons plus particulièrement sur la possibilité d'importer de l'OBJ, nous pouvons vous présenter les diagrammes de classes puis de séquence liés à cette nouvelle idée et donc vraiment proches de la réalité actuelle.

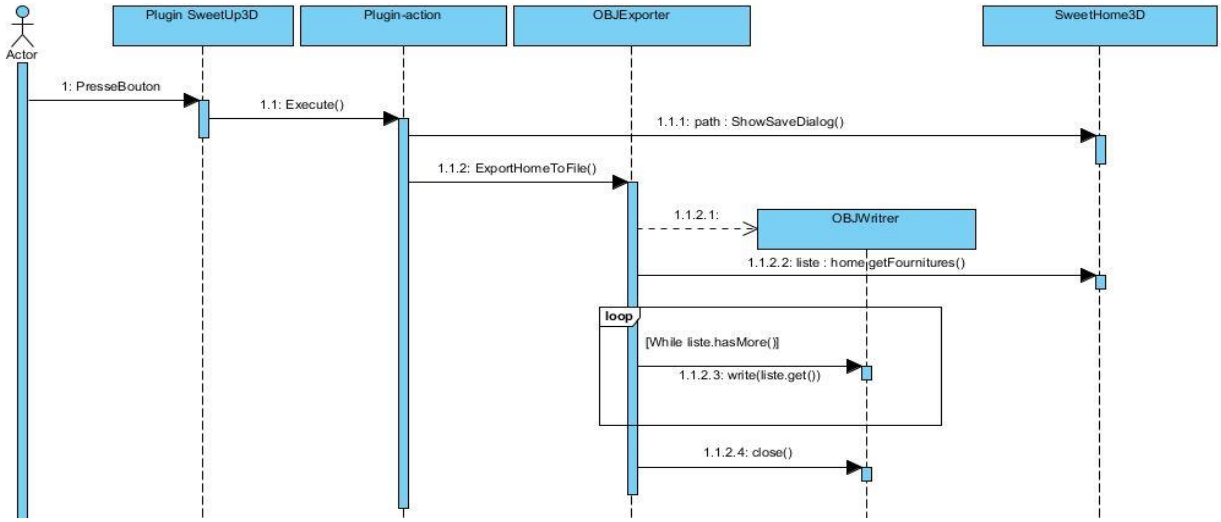
Voici donc, ci-dessous, le diagramme de classes montrant la relation qui existe entre le plugin que nous avons créé et le logiciel SH3D :



Comme le montre ce diagramme de classes, le plugin utilise des fonctions déjà présentes dans le code source de SH3D, afin d'ajouter un menu déroulant à l'interface de SH3D. Lors de l'utilisation de ce bouton, l'utilisateur va déclencher une série d'évènement afin de pouvoir sauvegarder la maison.

Le pluginVolume permet d'ajouter à SH3D un bouton dans la barre d'outils, qui lorsqu'il est pressé lancera la methode execute() présente dans la classe PluginAction. Dans notre plugin, cette méthode va utiliser un OBJExporter qui va pouvoir exporter la Home (c'est-à-dire la classe qui représente les objets présents dans l'environnement 3D) qui a été donnée au PluginAction lors de sa création. L'OBJExporter va donc récupérer chaque objet dans cette Home, et les écrire dans l'OBJWritter.

Voyons maintenant, le diagramme de séquence détaillant les interactions entre ces différentes classes :



Le diagramme de séquence ci-dessus montre les différentes interactions qu'il peut y avoir entre toutes les classes en jeu lorsqu'un utilisateur veut utiliser le plugin SweetUp3D.

Tout d'abord, nous utilisons une fonction (la méthode showSaveDialog()) déjà présente dans SH3D afin de demander à l'utilisateur où il voudra enregistrer le fichier. Ensuite, nous appelons encore une fois une fonction de SH3D, getSelectableViewableItems(), qui, cette fois, va nous renvoyer la liste entière de toutes les fournitures de la maison (c'est-à-dire les objets). Il sera possible, à partir de cette liste, de transformer chaque objet, un à un, en un nouvel objet qui sera lisible par la classe servant à l'écriture. Les objets s'ajoutent donc tous dans un fichier, les uns à la suite des autres. Cependant, avant de les transformer, nous prenons certains objets qui posaient problème dans SketchUp et notamment les murs, et nous leurs donnons des propriétés différentes afin de nous assurer que l'importation dans SketchUp donnera un résultat convenable à l'œil. Une fois chaque objet écrit sans erreur dans le fichier, on ferme la classe permettant l'écriture.

Si vous souhaitez obtenir plus d'informations sur la conception mais également sur les diagrammes de gestion de projet, nous vous invitons à consulter le cahier des charges du projet.

4.ScracthHome

Le second projet, ScratchHome consiste à créer une passerelle entre Scratch, logiciel éducatif, et SH3D. Le but est qu'un message puisse être envoyé depuis Scratch, en donnant en information un meuble à modifier et la méthode de modification. Un plugin sur SH3D devra donc être capable de récupérer ce message, le comprendre et modifier comme voulu le meuble demandé.

Le plugin aura aussi une autre fonctionnalité qui sera de générer des blocs de contrôle sur Scratch, afin de pouvoir créer le moyen de communication entre les deux logiciels .En effet, c'est par la modification des blocs dans Scratch que les propriétés des objets concernés dans SH3D se verront modifiées.

Au départ, nous songions créer en python une application qui servirait de passerelle entre les deux à l'aide de la gestion des sockets qui est assez simple en python. Cependant, la solution du plugin nous intéresse plus de prime abord car, de ce fait, il sera plus facile de modifier les objets de SH3D puisque nous serons directement dans le logiciel concerné.

Pour récupérer une liste d'objets de SH3D, nous avons découvert la présence d'une méthode sur le logiciel permettant l'exportation de la liste des objets sous le format CSV. Cela nous donnerait ainsi une liste des objets de SH3D que nous pourrions alors utiliser pour permettre à Scratch de pouvoir affecter à ses blocs des identifiants d'objets qui correspondront à ceux dans le ficher CSV. Le plugin, qui connaîtra ces identifiants, saura modifier le bon objet avec les nouvelles propriétés spécifiées par Scratch.

Ce projet sera réalisé au semestre 4 car nous avons, pour le semestre 3, favorisé le premier projet, à savoir celui du transfert depuis SH3D vers SketchUp. De ce fait, à part quelques idées de départ à l'instar de notre réflexion sur l'utilisation d'un fichier CSV, nous n'avons, à l'heure actuelle, pas d'avancées particulières à présenter.

5.Conclusion

Pour ce semestre 3, nous avons décidé de nous concentrer entièrement sur le premier sous-projet parmi les deux qui composent le projet principal. En effet, ces deux sous-projets demandent un temps de travail somme toute relativement long et n'ont en outre pas de réel lien entre eux si ce n'est que le logiciel SH3D les concerne tous les deux. De plus, l'absence, dès le début, d'un membre de notre groupe de projet (Maxime HOUDEAU) a réduit l'équipe à trois personnes, ce qui nous a décidés à nous concentrer sur un seul projet dans un premier temps.

Ainsi, pour ce premier sous-projet, celui du transfert de SH3D à SketchUp, il est assez original en ce qu'il diffère d'un projet de développement classique de par le fait que le but premier n'était pas de créer une application, mais de faire évoluer un code déjà existant, en l'occurrence celui de SH3D. Pour ce faire, il a donc fallu une longue période de compréhension et d'étude de l'API de SH3D, à l'aide des diagrammes de classes (voir Annexes 6 et 7) proposés par le créateur du logiciel et avec le code source qu'il propose au téléchargement, SH3D étant libre. Si nous sommes aujourd'hui assez proches d'une solution viable, il nous faut néanmoins rapidement trouver une substitution à FluidImporter, le plugin d'importation de SketchUp qui est fonctionnel mais malheureusement payant. Quand cela sera fait, nous pourrons ainsi exporter, en OBJ, des fichiers créés dans SH3D puis les importer dans SketchUp sans problèmes visuels et également avec les textures incluses.

6.Annexes

Annexe 00 : Sources

SweetHome 3D : http://www.sweethome3d.com/fr/

Sketchup : https://www.sketchup.com/fr

DAE Collada : https://www.khronos.org/collada/

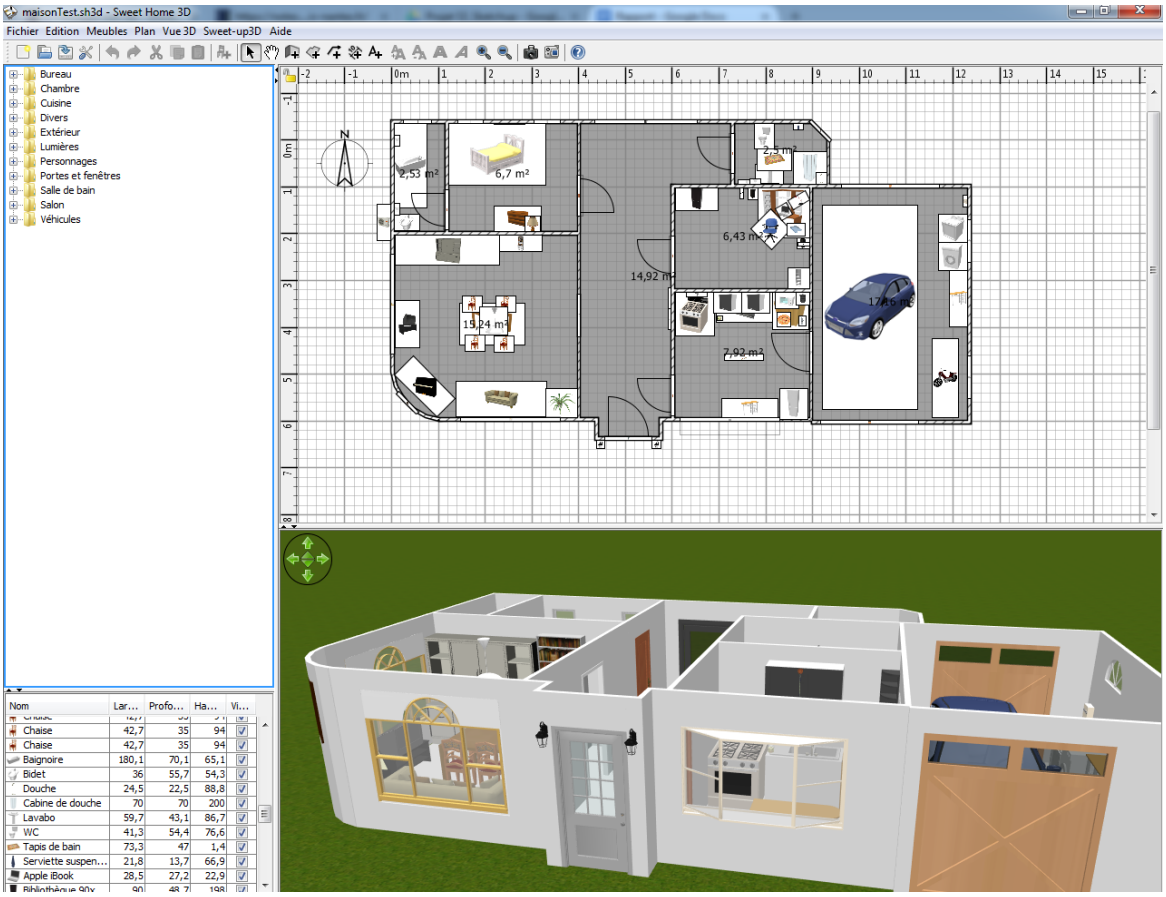
GitHub : https://github.com/zeptoline/sweet-up3d

Wikipedia (formalismes des formats OBJ et DAE) : https://fr.wikipedia.org/

JavaDoc : https://docs.oracle.com/

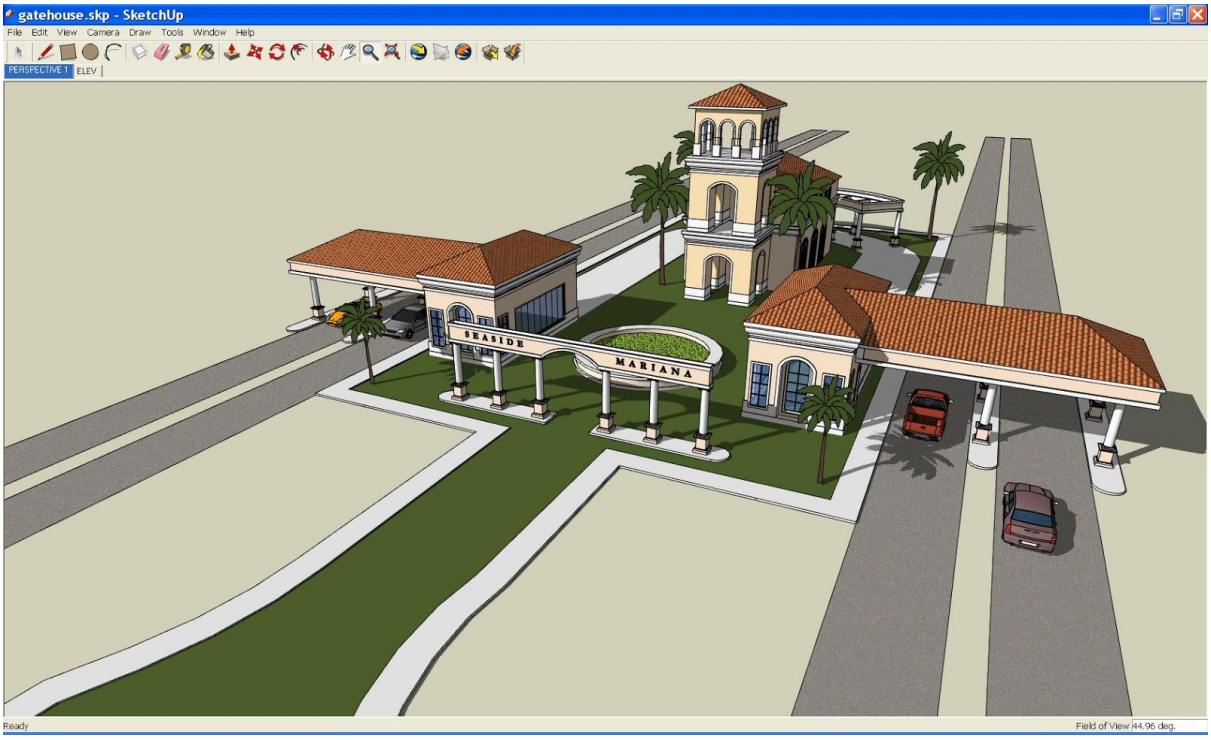
Annexe 01 : Rendu maison réalisé avec Sweet Home 3D

Annexe 02 : L'interface de Sweet Home 3D

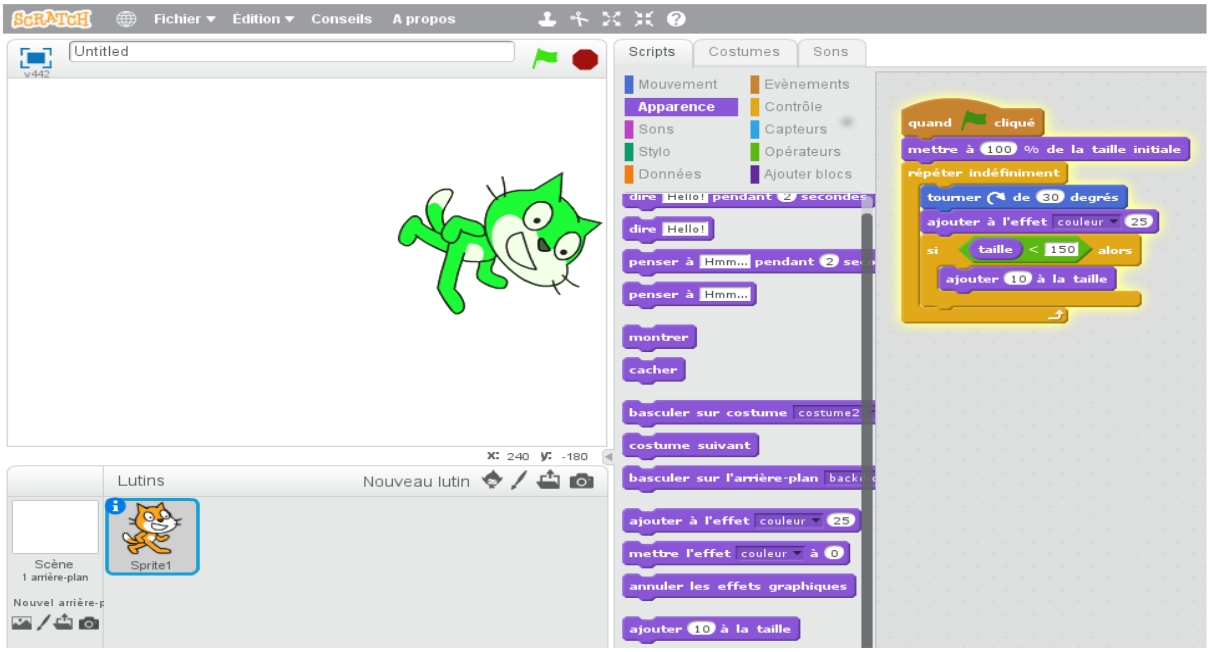


Annexe 03 : Exemple utilisation de SketchUp dans Google Maps

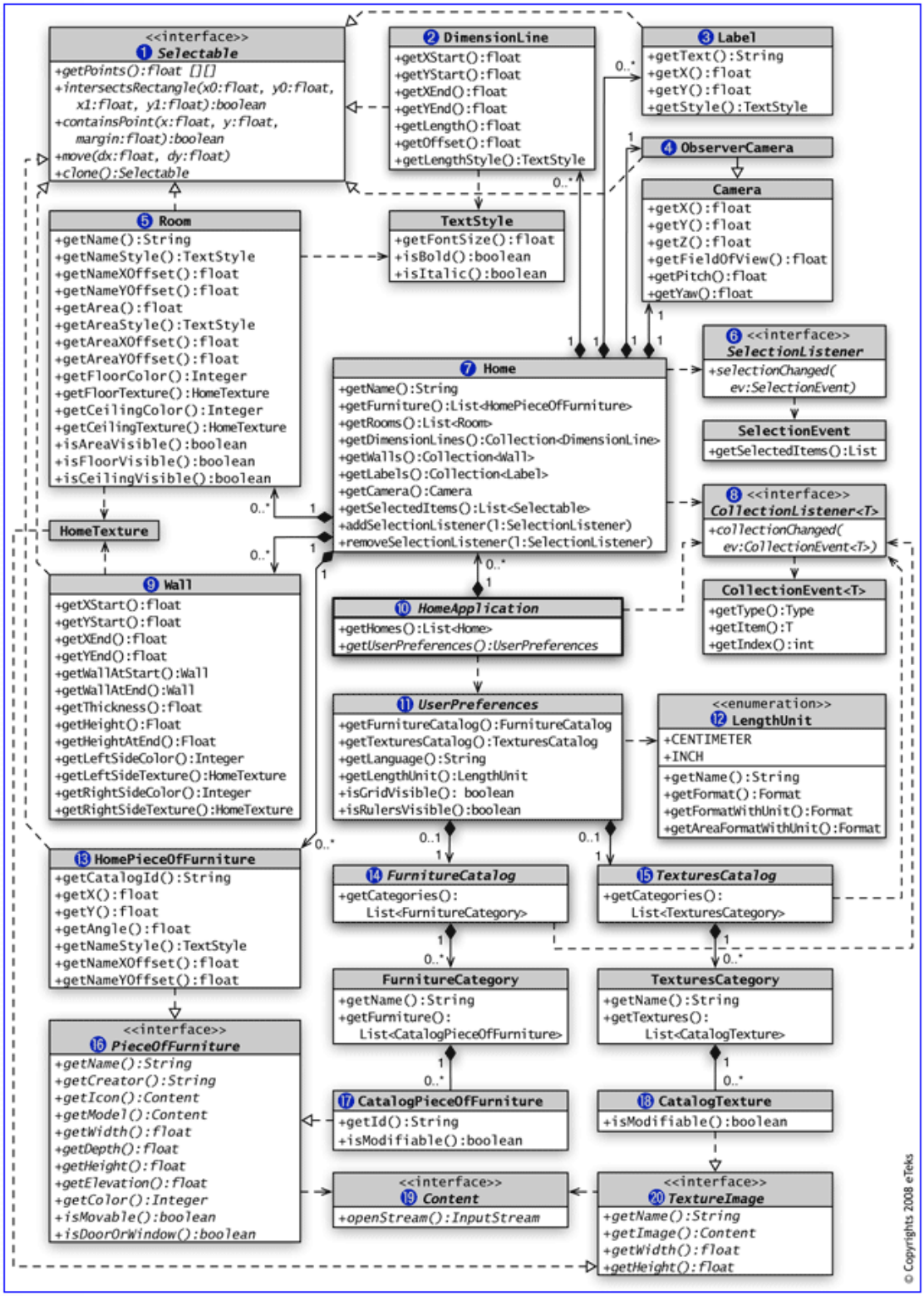
Annexe 04 : L'interface SketchUp



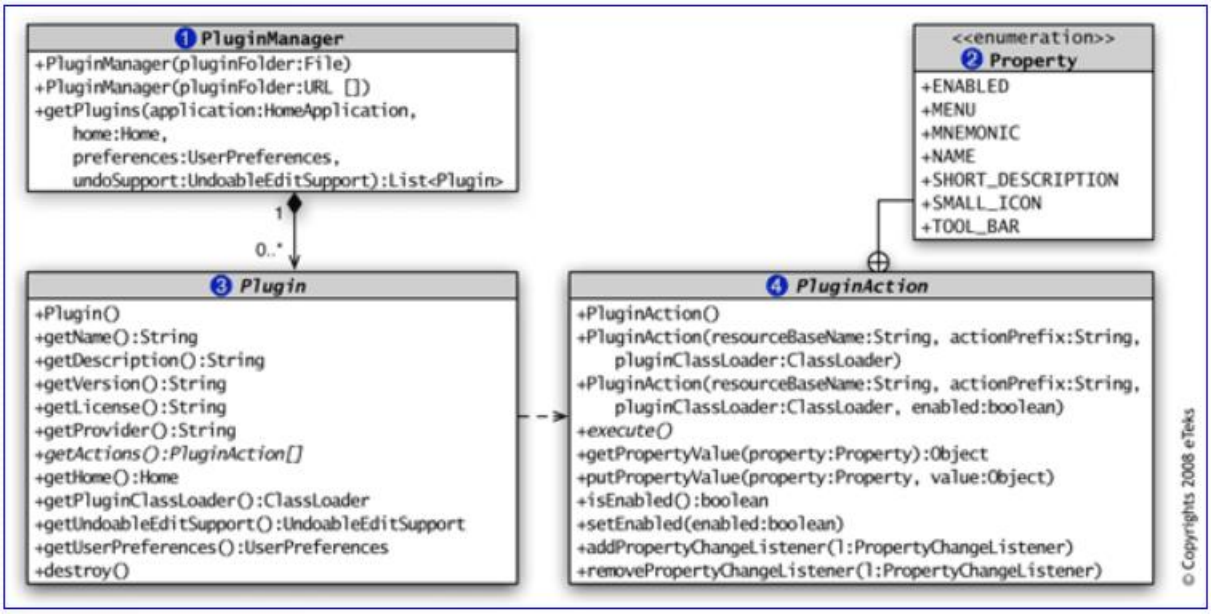
Annexe 05 : Scratch et son système de brique de couleur. A gauche le résultat des instructions à droite.



Annexe 06 : Diagramme des classes de Sweet Home 3D



Annexe 07 : Diagramme de classes des plugins de Sweet Home 3D



Projet SweetUp3D

Rapport de Projet Technologique

Université de Nantes

Institut Universitaire de Technologie

Département Informatique

Nantes, le 18/01/16

INSTITUT UNIVERSITAIRE DE TECHNOLOGIE

3, rue du Maréchal Joffre 44041 cedex 01 - http://www.iut-nantes.univ-nantes.fr