Rapport Technique

Florian Barrois Nicolas Devillers Valentin Jeanroy Mehdi Loisel
Jean Mercadier Ismail Taleb Willeme Verdeaux

22 mars 2014

Table des matières

Introduction

Chapitre 1

Installation

1.1 Installation

1.1.1 Prérequis

Pour lancer l'application, il est obligatoire d'avoir a disposition un serveur MySQL. Ainsi au lancement le programme chargera ses information dans la base de donné. Mais si la base de donné n'existe pas celui ci sera crée avec le nom de stcal.

Cette base de donnée permet ainsi à l'application de sauvegarder et recharger ses donné au lancement et à la fin de l'execution du programe.

note : Il est conseillé de créer un utilisateur spécifique au programme qui aura tout les privilleges sur la base de donné stcal.

1.1.2 Obtenir l'application

L'integralité de l'application se trouve sur GitHub sur le repo stcal : https://github.com/Ricain/stcal

Executable Pour obtenir un exécutable de l'application, soit un fichier jar il suffit de le telecharger sur GitHub à l'url suivant : https://raw.githubusercontent.com/Ricain/stcal/Main/stcal.jar

Code source Pour obtenir le code source d'une maniere "propre" (sans le telecharger directent dans un fichier zip sur GitHub), il suffit de cloner le projet :

```
$ git clone https://github.com/Ricain/stcal
```

L'integralité du projet sera cloné dans un repertoire stcal sous le repertoire courant. Cela nécessite d'avoir git d'installé. Dans un enviroment linux, le gestionnaire de packet permet de l'installer. Sur un mac, il faut installer executer la commande suivante apres avoir installé Xcode :

```
$ xcode-select -install
```

Une fois git installé il est possible de faire des "commit" et des "pull request".

IDEA Il est cependant possible de cloner dirrectement le projet à partir d'IntelliJ IDEA. Pour cela il suffit d'importer un projet à partir d'un VCS (Version Control System). L'IDE vous demandera le lien GitHub de l'application ennoncé plus haut.

Chapitre 2

Structure

2.1 MVC

L'arboressance du projet a été pensé pour respecter au mieux le modele vue controleur. Sous le repertoire src on trouve trois autres repertoire :

control contient l'enssemble des class moteur à l'application (controleur).

fen abreviation de fenettre, contient l'enssemble des class permettant de dessiner l'application (vue).

don abrevaiation de donné, contient l'eesemble des class concernnat les données de l'application (modele).

2.1.1 Modele

Le modele constitue les données de l'applications. L'essemble des classes prevue à cette effet sonst stocké sous le repertoire *don*. On y trouve des classes comme :

- Etudiant
- Soutenance
- Prof
- Agenda
- etc

Sauvergarder les données de l'application permet non seulement de garder un historique des stages mais égualement de réouvrire l'application et de la retrouver tel qu'on l'a fermé. A cet effet on trouve un sous repertoire *manager* qui contient un enssemble de classe permettant de faire le lien entre la base de donné et le modele de donné de l'application. Chanque manager **doit** implement l'interface manager afin de respecté le concepete de JBDD (enseigné en S4).

Le script qui permet d'installer la base de donné se trouve dans le re repertoir *res* (ressource).

Les classe constituant le repertoir *don* on était pensé sur un modele bien precis. On verra le MCD et l'UML dans la partie modelisation de ce rapport.

2.1.2 Vue

La vue est constitué des classes sous le repertoire *fen* elles permettent de dessiner les fenetre. La bibliotheque graphqiue utilisé ici est *Swing*.

La fenetre principale est dessiné par la class *FInterface*. On lui ajoute des object de type *FTab* afin de creer des onglets. Il est important de noter que aucune des class fenetres (class dans le repertoir *fen*) n'herite d'un quelconque objet de la bibliotheque graphique, elles sont plustôt composé de ces objets.

La classe *Main* contient une methode *mac*. Cette methode est importoante pour adapetre la partie graphique au systeme OS X.

2.1.3 Controleur

Les classes concerant le controleur sont situé dans dans le repertoire *control*. Ces class font le moteur de l'application et servent de lien entre la partie modele et la partie vue. La class *Main* est une exception car elle fait partie du controler mais ne se trouve pas dans le meme repertoire que les autres class.

Les class *Datas*, *DBTools*, *DBSEtting* et *ScriptRunner* permettent de gerer la partie donné de l'application. Les methodes `load()` et `save()` dans la class *Datas* permettent respectivement de charger et sauvergarder le model dans la base de donnée. Ces deux methodes sont respectivement appellé aux debut et à la fin du programe. La class *DBTools* permet de faire des operation sur la base de donnée et *DBSetting* contient les information de connection à la base de donné et permet d'obtenir un connection valide à celle ci. *ScriptRunner* est un classe provenant du projet Ibatis et permet d'executer des scripts SQL comme le script d'installation de la base de donné dans le repertoir *res*.

2.2 Detail de class

2.2.1 Setting

Les class herité de *Setting* permettent de stocker des information, celci sont stocké dans le répertoir `/.stcal` dans le répertoir pernonelle de l'utilisateur. Même si l'application est lancé sur Windows, dans ce cas le dossier ne sera pas caché.

Ce systeme de stocker des informations est utilisé pour les identifiant de connection à la base de donné (*DBSEtting*) et sauvegarder les choix de l'utilistateur rentré dans le formulaire du calendrier (*CALsetting*). Il est important de noter que tout mot de passe dans ces fichiers ne sont pas crypter. C'est pour quoi il est preferable d'avoir un utilisateur de base de donnée propre à l'application et surtout de ne pas mettre le mot de passe root.

2.2.2 Datas et DBTools

Datas est le lien entre la partie donnée et la partie controle. Elle contient donc des listes d'object qui caractereise la partie model. Ces listes sont static et publique afin d'etre utilisé par l'enssemble de la partie controle. La methode `load()` et la methode `save()` sont appellé aux debut du programme afin de charger l'integralité des donnéeet les stocker dans cette class. Et l'inverse à la fermuture du programme.

DBTools et une class qui effectue des operations sur la base de donné. Elle permet de (re)creer la base de donnée à partir du script SQL mais aussi de gerrer les parametre de connection et donc la connection.

2.2.3 Message

Cette class gere les popups, la sortie stendart, et la sortie d'erreure.

Le principe des popup est pensé pour se rapprocher du javascript. En effet, afficher un popup consiste juste a appeller une methode avec le message en parametre. Pendant que le message est à l'ecran, l'application s'arrette. De plus il est possible de passer une exeption en parametre afin d'afficher le message de cette derniere. Il y a quatre type de popup :

notice est un un message d'information

warning affiche un avertissement

error affiche un message d'erreur

question affiche une question à l'utilisateur et celui-ci peut cliquer oui ou non

Dans cette class on recupere egalement la sortie standart et la sortie standart d'erreure. Cela permet si besoin de rediriger tout ce qu'ecrit l'application dans des fichiers comme des logs.

Chapitre 3

Modelisation

3.1 UML

3.2 Base de donnée

Conclusion