Calculatrice Graphique

komlan godwin AMEGAH[1] and amadou Diattara[1]

[1]Licence-2 UFR-Maths-info

May 2021

1 Introduction

Au cours de notre deuxième année, nous avions une matière appelée **Programmation Orienté Objet** (POO). Durant l'évolution dans cette UE il nous ait demandé de réaliser un projet intitulé « **Calculatrice Graphique** ». En effet le but de notre mini-projet est de réaliser une calculatrice scientifique permettant de réaliser des opérations de calcul. Le code du projet doit être réalisé en langage `JAVA`.

2 Choix d'implémentation

2.1 Le cahier de charge

Le projet s'intéresse à créer une calculatrice ayant un certains nombre de caractéristique. La calculatrice que nous souhaitons obtenir aura les caractéristiques suivantes :

1. Permette les différentes conversions entre les bases décimales, binaires, octales et hexadécimales.
2. Elle prend en charge la gestion des erreurs de saisie.
3. Donne la main de se basculer entre le mode standard et le mode scientifique en cliquant sur simple bouton radio.

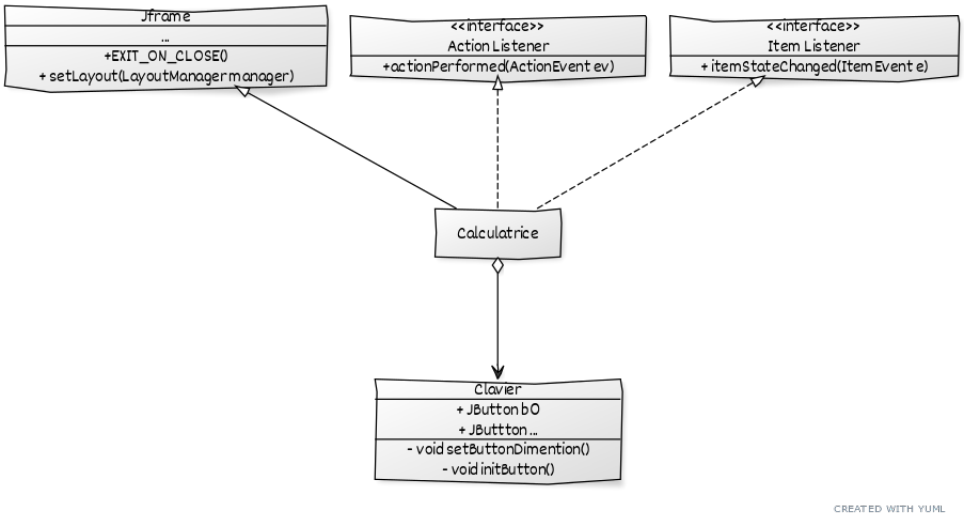
Ces deux modes sont décrits comme suit :

1. Mode standard : Il s'agit des boutons qui définissent les chiffres et les opérations de calcul tel que Le produit, la somme, la soustraction, la division, un bouton pour définir PI=3.14, un bouton pour définir la racine carrée etc...

2. Mode scientifique : Concernant ce mode, Nous avions ajoutez des fonctions les plus utilisées par exemple : les opérations logarithmiques, Triangulaires (sin, cos, tan), Inverse (1/x), Exponentielle. Aussi nous avions ajoutez le codage des chiffres dans différentes bases :Décimale, Binaire, Octale, hexadécimale

2.2 La modélisation

Pour une modélisation propre de notre `calculatrice`, nous avions utlisé un langage de modélisation: `UML`.
Tout d'abord une calculatrice est un programme (appareil) qui permet de réaliser des opérations mathématiques. Il est entre autre composé d'un ensemble de touches représentant les opérations de calcul et un écran permettant d'afficher les résultats des opérations, graphes etc...
Nous disposons de trois classes: `Clavier`, `Calculatrice` et `Main`.
La `Clavier` modélise tous les boutons de notre calculatrice. La classe `Calculatrice` contient un champs de type clavier et étend une `Jframe`.



3 Les méthodes principales

3.1 Les composants

3.1.1 Le panneau textuel

Il s'agit du champs de texte ou seront inscrit tout les opérations tapées au clavier.Il est représenté par le composant de type `JTextField txt`.

3.1.2 Le panneau Standard

Ce conteneur regroupe tous les boutons numériques ainsi que ceux représentant les opération de base (*addition, soustraction...*). La *méthode privée* `CreatStandarPannel` permet de créer ce composant tout en y rajoutant les boutons.

3.1.3 Le panneau Scientifique

Il s'agit cette fois-ci d'un conteneur qui regroupe tous les boutons de conversion(*décimal, hexadécimal, etc...*) ainsi que ceux représentant les opération scientifiques(*logarithme, exponentiel,...*). La *méthode privée* `createScientistPanel` permet de créer ce composant tout en y rajoutant les boutons.

3.2 Conversions en base

3.2.1 Activation des buttons

Pour une bonne orientation des utilisateurs nous disposons des fonctions permettant de rendre inutilisable certains boutons. Cette fonctionnalité permet entre autre en fonction de la `base` choisis de manipuler un certain nombre de bouton propre à cette dernière. C'est le cas par exemple de la fonction `activateHexadecimalButton` qui permet de rendre accessible les boutons représentant des caractère hexadécimaux (`A..F`).

3.2.2 Détermination des bases

Pour les opérations de conversion ils nous a fallu déterminer la base de départ ainsi que celle d'arrivé. Cette responsabilité est laissé aux méthodes `getSourceBase` et `getDestinationBase`.

3.3 Les opérations de calcul

Pour effectuer des opérations l'utilisateur devra se servir des boutons. Pour ce faire Nous avions placer des écouteurs au niveau de chaque bouton grâce à la méthodes `listenAllButton`. A chaque bouton est associé une action implémentée dans la méthode `actionPerformed`.

Pour réaliser les opérations il a fallu récupérer les données dans le bon format. Nous avions avions donc effectuer une opération de `parsing` sur chaque données du champ textuel. La méthode de *parsing* utilisée correspond à celui disponible dans les *Class Interger, Double, ...*

Il est important de noter que certains opérations nécessitent un traitement particulier notamment par exemple l'*opération de division*. La division par zero n'étant pas possible, nous avions effectuer une *gestion d'exception* dans des blocs `try..catch`.

4 Difficultés rencontrées

4.1 Le tracé des graphes

Nous avions rencontré des problèmes pour concevoir l'affichage des courbes des fonctions. En effet l'affichage des courbes nécessitait une autre fenêtre graphique. Cependant compte tenu de certains bugs, de la complexité des taches et aussi du deadline, nous avions décider laisser temporairement cette partie et de nous concentrer sur la partie principal de notre application. Evidemment cette version pourra évoluer dans le temps.

5 Répartition des tâches

La répartition des tâches s'est faites assez intuitivement de manière à ce que l'on puisse travailler indépendamment en parallèle.De plus elle s'est faites de telle manière que chacun doive attendre la fin du travail de l'autre pour être sur de travailler de manière séquentielle. Nous nous sommes aussi mis d'accord sur les conventions de nommage des méthodes, de Commentaires et de l' Indentation pour rendre le code plus lisible.

AMADOU s'est charger de toute la partie graphique basé sur sa conception `UML` (voir section *modélisation*). GODWIN s'est charger de la partie implémentation de méthode associé à chaque bouton. Certaines méthodes non détaillés dans ce document ont été implémentés. Il s'agit là dans ce cas d'un travail collectif.

6 Conclusion

En conclusion nous avons étudié les étapes de la réalisation d'une calculatrice de sa conception jusqu'à son exécution. Pour cela nous somme partie d'une modélisation de notre calculatrice qui nous à permis d'identifier en globalité les *Class* et *méthodes* nécessaires. Ensuite nous avons implémenter et détaillé les différentes fonctions et conditions dans la programmation de notre calculatrice scientifique suivant un cahier de charge bien défini.

7 Annexes

Contents