Projet Chronomètre

Introduction

L'objectif principal du Projet Chronomètre est de construire une interface graphique permettant de visualiser les heures, minutes et secondes qui défilent selon trois modes : chronomètre, compte à rebours et minuterie. De plus, l'utilisateur de l'application doit pouvoir sectionner le temps défilant.

Caractéristiques du projet :

- ❑ Interface graphique avec l'affichage numérique de l'heure et un panneau de visualisation du temps sectionné
- ❑ Fonction de chronomètre : de zéro jusqu'à l'infini
- ❑ Fonction du compte à rebours : temps saisi jusqu'à zéro
- ❑ Fonction de minuterie : de zéro jusqu'au temps saisi
- ❑ Fonction de suppression d'un temps sectionné sélectionné

Aspects spécifiques :

Langage : Java version 8

Logiciel de développement : Netbeans IDE 8.2

Outil de communication des tâches à faire, en cours et terminées : Trello

Ressources utilisées : projets effectués dans le cadre de mes études, recherche de solutions techniques sur Internet

Résolution technique :

Dans ce projet, j'ai utilisé la classe Thread de Java. Le cycle de vie du Thread décrit ci-dessous, suit plusieurs étapes.

```
// start a thread when start button is clicked
if (e.getSource() == start) {
    reset();
    changeLabel();

    // stopwatch is on
    on = true;
    // starting thread
    new Thread(this, "StopWatch").start();
    cutter.setEnabled(on);

    start.setVisible(false);
    stop.setVisible(on);

}
```

Image 1 : Démarrage du thread

A chaque pression sur le bouton « Commencer », un thread en Java démarre dans un processus par la création d'un objet Thread avec l'opérateur *new*, puis appelle sa méthode start(). Le thread est alors à l'état « Runnable », prêt à fonctionner.

Ensuite, lorsque le CPU commence à exécuter le thread, l'état de ce dernier passe à « Running ». La méthode run() est alors exécutée.

```
// thread.run method
public void run()
{
    if(measurement.isSelected()){
        // while the stopwatch is on
        while (on) {
            try {
                // pause 1000 milliseconds
                Thread.sleep(1000);
                // update the time
                update();
                // Properly formatting the display of the timer
                changeLabel();
            }
            catch (InterruptedException e) {
                System.out.println(e);
            }
        }
    }
```

Image 2 : Actions du mode chronomètre lorsque le thread est en cours d'exécution

```
// update the timer
public void update()
{

    second++;
    if (second == 60) {
        second = 0;
        minute++;
        if (minute == 60) {
            minute = 0;
            hour++;
        }
    }

}
```

Image 3 : Incrémentation des secondes, des minutes jusqu'aux heures pour le mode chronomètre

Une fois que la variable booléenne *on* du thread passe à la valeur « false », le thread, lui, a fini de s'exécuter et passe ainsi à l'état « Dead ».

Résultats obtenus :

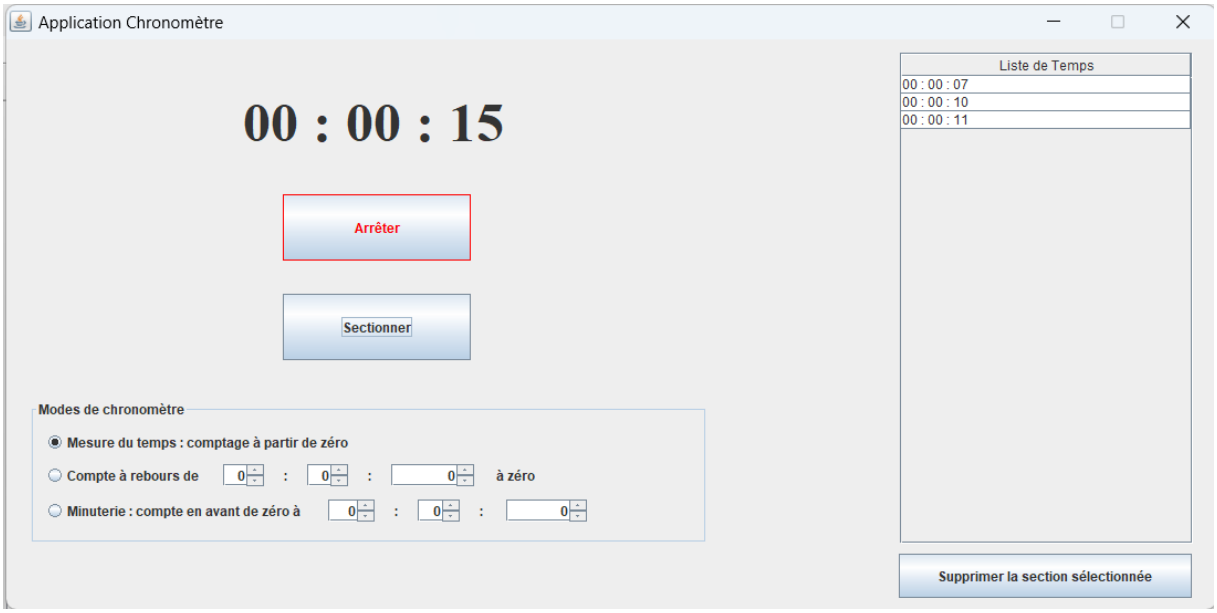

Image 4 : Visuel de l'interface graphique du mode chronomètre

Les trois modes de chronométrage : chronomètre, compte à rebours et minuterie fonctionnent tous correctement. De plus, pour chaque mode, quand le temps défile, un bouton permet de sectionner le temps et de l'envoyer dans un panneau dédié. Dans le panneau de liste de temps, la suppression d'un temps sectionné sélectionné est également fonctionnel.