TP 4 : Thread

On va reprendre le chronomètre des TPs précédents et le rendre fonctionnel.

1°) Modification de l'interface

- Le texte du bouton "Plus" est modifié pour devenir "Start".
- Le texte du bouton "Moins" est modifié pour devenir "Stop".
- Leurs bulles d'aide sont également modifiées en "Lancer le chronomètre" et "Arrêter le chronomètre".
- Au départ le bouton "Stop" est inactif.

Remarque : Pour la question suivante on ne tiendra pas compte du choix de la vitesse de chronométrage et on considèrera qu'il s'agit toujours du mode minutes et secondes (on modifiera cela plus tard).

2°) Ajout de l'avance automatique du temps
Modifier la classe du chronomètre de façon à ce que :

- Le bouton "Start" provoque le démarrage d'un thread (classe à écrire) qui fait fonctionner le chronomètre (les secondes et les minutes avancent).
- Le bouton "Stop" provoque l'arrêt du chronomètre en faisant se terminer le thread.

Remarques :

1. Après avoir arrêté le chronomètre si on le relance il doit continuer à partir de la valeur actuellement affichée. Si on veut qu'il reparte de zéro il faut utiliser le bouton "reset" ou changer de vitesse de chronométrage avant de le relancer.
2. Lorsque le chronomètre est en marche le bouton "Start" et la liste de choix de la vitesse de chronométrage sont inactifs tandis que "Stop" est actif.
3. Lorsque le chronomètre est arrêté le bouton "Start" et la liste de choix de la vitesse de chronométrage sont actifs tandis que "Stop" est inactif.
4. Les boutons "Reset" et "Capture" sont toujours actifs.

Vérifier que le chronomètre fonctionne correctement en particulier que l'on puisse bien l'arrêter et le relancer plusieurs fois et qu'il reparte bien de la valeur affichée. Vérifier également que la capture du temps intermédiaire et la remise à zéro du chronomètre soient possibles qu'il soit ou pas en marche.

3°) Modifier le thread pour que la vitesse de fonctionnement du chronomètre tienne compte du mode choisi (minutes et secondes ou secondes et 1/100).

SOLUTION

```
package tp4;
import javax.swing.JFrame;
import java.awt.BorderLayout;
import javax.swing.JPanel;
import java.awt.FlowLayout;
import javax.swing.JTextField;
import javax.swing.JLabel;
import java.awt.GridLayout;
import javax.swing.JButton;
import javax.swing.JComboBox;
import java.awt.Font;
import java.awt.Color;
import javax.swing.SwingConstants;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

public class Chronometre extends JFrame {
	private JTextField minutes;
	private JTextField secondes;
	private JTextField tempsIntermediaire;
	private JButton start, stop, capture, reset;
	private JComboBox<String> mode;
	private JLabel unite1, unite2;
	private boolean minsec;
	private Moteur moteur;

	public Chronometre() {
		super("Chronomètre");
		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		minsec=true;
		getContentPane().setLayout(new BorderLayout(0, 0));

		JPanel panel = new JPanel();
		getContentPane().add(panel, BorderLayout.CENTER);
		FlowLayout fl_panel = new FlowLayout(FlowLayout.CENTER, 10, 15);
		panel.setLayout(fl_panel);

		minutes = new JTextField();
		minutes.setToolTipText("Minutes ou secondes");
		minutes.setHorizontalAlignment(SwingConstants.CENTER);
		minutes.setText("00");
		minutes.setForeground(Color.RED);
		minutes.setBackground(Color.WHITE);
		minutes.setFont(new Font("Tahoma", Font.PLAIN, 36));
		minutes.setEditable(false);
		panel.add(minutes);
		minutes.setColumns(2);

		unite1 = new JLabel("Min");
		panel.add(unite1);

		secondes = new JTextField();
		secondes.setToolTipText("Secondes ou 1/100 de seconde");
		secondes.setText("00");
		secondes.setHorizontalAlignment(SwingConstants.CENTER);
		secondes.setForeground(Color.RED);
		secondes.setBackground(Color.WHITE);
		secondes.setEditable(false);
```

```
        secondes.setFont(new Font("Tahoma", Font.PLAIN, 36));
        panel.add(secondes);
        secondes.setColumns(2);

        unite2 = new JLabel("Sec  ");
        panel.add(unite2);

        JPanel panel_1 = new JPanel();
        getContentPane().add(panel_1, BorderLayout.EAST);
        panel_1.setLayout(new GridLayout(3, 1, 1, 4));

        start = new JButton("Start");
        start.addActionListener(new ActionStart());
        start.setToolTipText("Lancer le chronom\u00E8tre");
        panel_1.add(start);

        stop = new JButton("Stop");
        stop.setEnabled(false);
        stop.addActionListener(new ActionStop());
        stop.setToolTipText("Arr\u00EAter le chronom\u00E8tre");
        panel_1.add(stop);

        reset = new JButton("Reset");
        reset.addActionListener(new ActionReset());
        reset.setToolTipText("Remise \u00E0 z\u00E9ro du chronom\u00E8tre");
        panel_1.add(reset);

        JPanel panel_2 = new JPanel();
        getContentPane().add(panel_2, BorderLayout.SOUTH);
        FlowLayout fl_panel_2 = new FlowLayout(FlowLayout.CENTER, 10, 5);
        panel_2.setLayout(fl_panel_2);

        tempsIntermediaire = new JTextField();
        tempsIntermediaire.setToolTipText("Temps interm\u00E9diaire");
        tempsIntermediaire.setText("00:00");
        tempsIntermediaire.setHorizontalAlignment(SwingConstants.CENTER);
        tempsIntermediaire.setBackground(Color.WHITE);
        tempsIntermediaire.setEditable(false);
        tempsIntermediaire.setFont(new Font("Tahoma", Font.PLAIN, 20));
        panel_2.add(tempsIntermediaire);
        tempsIntermediaire.setColumns(5);

        capture = new JButton("Capture");
        capture.addActionListener(new ActionCapture());
        capture.setToolTipText("Capture du temps interm\u00E9diaire");
        panel_2.add(capture);

        mode = new JComboBox<String>();
        mode.addActionListener(new ActionChoix());
        mode.setToolTipText("Choix du mode (min:sec ou sec:1/100)");
        mode.setFont(new Font("Tahoma", Font.ITALIC, 13));
        mode.addItem("Min:Sec");
        mode.addItem("Sec:1/100");
        panel_2.add(mode);

        pack();
        setVisible(true);
    }
```

```
	private class ActionStart implements ActionListener {
		public void actionPerformed(ActionEvent e) {
			start.setEnabled(false);
			mode.setEnabled(false);
			stop.setEnabled(true);
			moteur=new Moteur(minutes, secondes, !minsec);
			moteur.start();
		}
	}

	private class ActionStop implements ActionListener {
		public void actionPerformed(ActionEvent e) {
			start.setEnabled(true);
			mode.setEnabled(true);
			stop.setEnabled(false);
			moteur.arreter();
		}
	}

	private class ActionReset implements ActionListener {
		public void actionPerformed(ActionEvent e) {
			minutes.setText("00");
			secondes.setText("00");
			tempsIntermediaire.setText("00:00");
		}
	}

	private class ActionCapture implements ActionListener {
		public void actionPerformed(ActionEvent e) {
			String v1,v2;
			v1 = minutes.getText();
			v2 = secondes.getText();
			tempsIntermediaire.setText(v1+":"+v2);
		}
	}

	private class ActionChoix implements ActionListener  {
		public void actionPerformed(ActionEvent e) {
			switch (mode.getSelectedIndex()) {
			case 0: unite1.setText("Min");
				unite2.setText("Sec");
				minsec=true;
				break;
			case 1: unite1.setText("Sec");
			unite2.setText("1/100");
			minsec=false;
			break;
			}
			minutes.setText("00");
			secondes.setText("00");
		}
	}

	public static void main(String[] args) {
		new Chronometre();
	}

}
```

```java
package tp4;
import javax.swing.JTextField;

public class Moteur extends Thread {
	private boolean rapide;
	private JTextField champ1, champ2;
	private boolean enMarche;

	public Moteur(JTextField c1, JTextField c2, boolean vitesse) {
		champ1=c1; champ2=c2;
		rapide=vitesse;
		enMarche = true;
	}

	public void run() {
		int limite, delai;
		if (rapide) {
			limite = 100;
			delai=10;
		}
		else {
			limite = 60;
			delai=1000;
		}
		while (enMarche) {
			try {
				sleep(delai);
				int sec;
				try { sec=Integer.parseInt(champ2.getText()); }
				catch (NumberFormatException nfem) {sec=-1;}
				sec++;
				if (sec==limite) {
					sec=0;
					int m;
					try { m=Integer.parseInt(champ1.getText()); }
					catch (NumberFormatException nfem) {m=-1;}
					m++;
					if (m==limite) {
						m=0;
					}
					if (m<10) champ1.setText("0"+String.valueOf(m));
					else champ1.setText(String.valueOf(m));
				}
				if (sec<10) champ2.setText("0"+String.valueOf(sec));
				else champ2.setText(String.valueOf(sec));
			}
			catch (InterruptedException ie) {}
		}
	}

	public void arreter() {
		enMarche=false;
		interrupt();
		}

}
```