Laboratoire no 2

Objectif

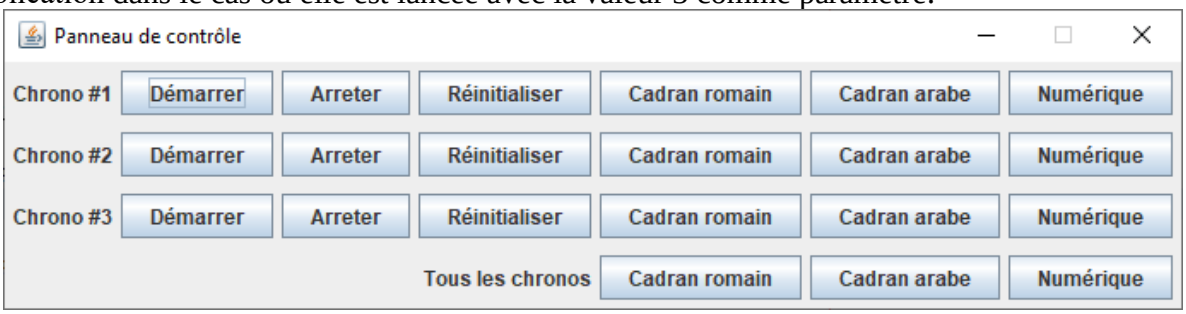
- Utilisation du modèle Observateur.

Donnée

Définir une application graphique qui permette de gérer des chronomètres et d'en afficher la valeur sous différentes formes.

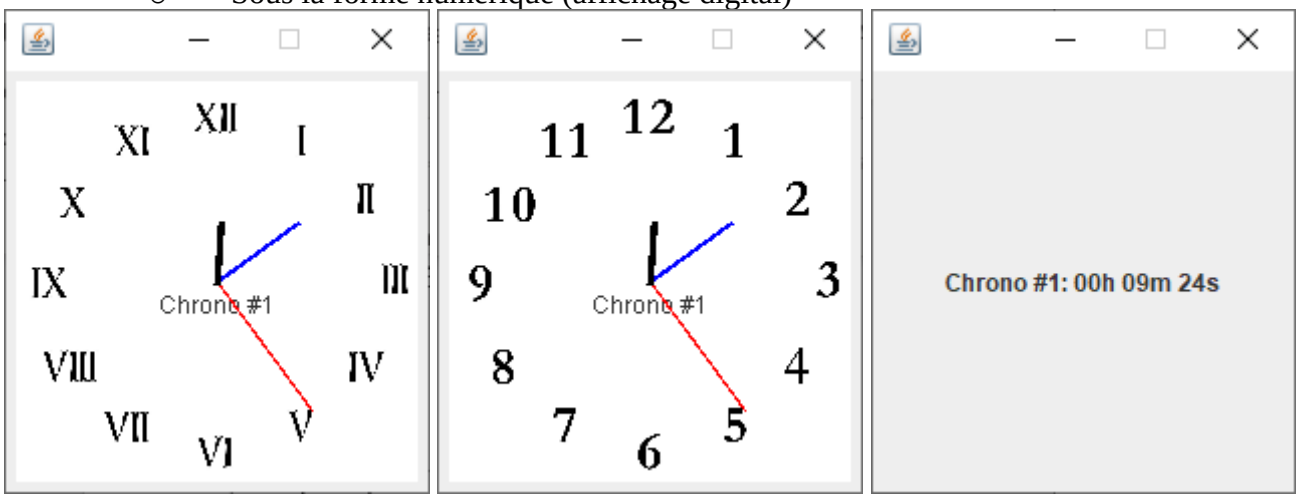
Chaque chronomètre mesure le temps en heures/minutes/secondes et est identifiable par son numéro, incrémenté à partir de 1.

L'application permettra de gérer de 1 à 9 chronomètres, ce nombre de chronomètres sera à passer en argument lors du lancement de l'application. La figure ci-dessous montre l'interface initiale de l'application dans le cas où elle est lancée avec la valeur 3 comme paramètre.

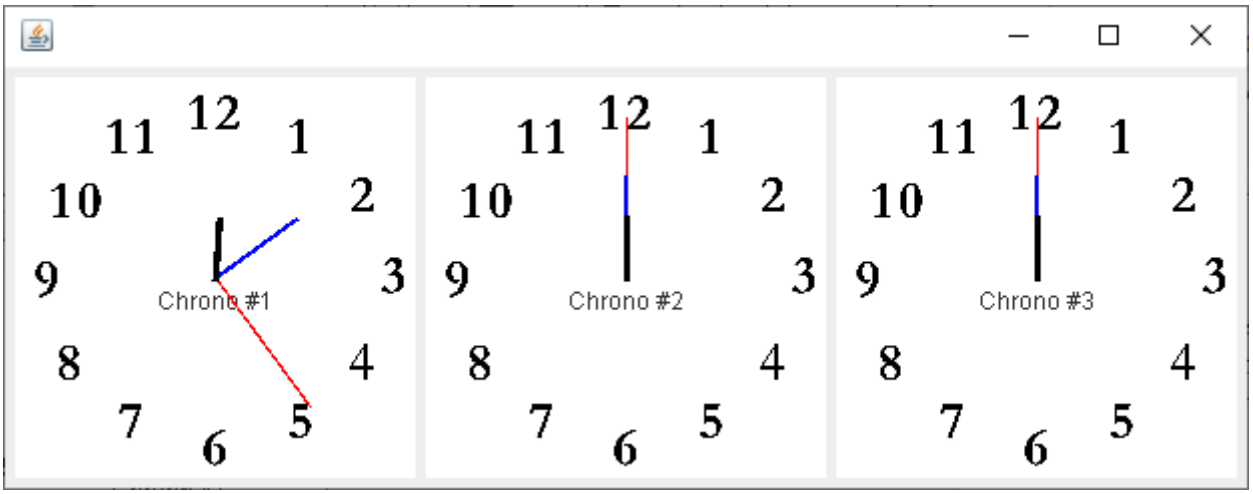


Pour chaque chronomètre on pourra, dans l'ordre des boutons affichés :

- Le (re)démarrer
- L'arrêter (le mettre en pause)
- Le réinitialiser (le remettre à la valeur 0)
- Chacun des 3 boutons suivants créera une fenêtre, non redimensionnable, affichant le chronomètre :
    - Sous la forme analogique (aiguilles) sur un cadran de chiffres romains
    - Sous la forme analogique (aiguilles) sur un cadran de chiffres arabes
    - Sous la forme numérique (affichage digital)



La dernière « ligne » du panneau de contrôle permet d'ouvrir des fenêtres qui afficheront, chacune, tous les chronomètres sous la forme voulue. Ces fenêtres doivent être redimensionnables, les chronomètres se réaligneront selon la place disponible (voir affichage vertical en annexe).



La taille de l'affichage de chaque chronomètre sera fixe et de 200x200. Chacun affichera son nom (Chrono + #numéro), avant sa valeur dans un affichage numérique, centré horizontalement et juste un dessous du centre dans les affichages analogiques.

On doit pouvoir créer autant de fenêtres d'affichage de chronomètres que l'on veut. On peut donc cliquer plusieurs fois sur le même bouton qui crée une fenêtre d'affichage de chronomètre(s), cela créera à chaque fois une nouvelle fenêtre sans clore les autres.

Cliquer sur l'affichage d'un chronomètre mettra ce dernier en pause s'il était en marche, le redémarrera s'il était à l'arrêt.

Chaque fenêtre peut être fermée, si on ferme le panneau de contrôle cela quitte l'application.

Implémentation

- Vous devez implémenter votre propre version du pattern Observateur, il ne faut pas utiliser l'implémentation fournie dans l'API Java
- Pour gérer le temps qui s'écoule dans chaque chronomètre, utiliser un `Timer` de l'API Java
- Les images des cadrans avec chiffres romains et arabes vous sont fournies

Indications AWT/Swing

- `Toolkit.getDefaultToolkit().getImage(String fileName)` permet d'obtenir un objet `Image` à partir du contenu du fichier ayant le nom `fileName`
- `getScaledInstance(int width, int height, int hints)` permet ensuite de redimensionner l'objet de type `Image`
- Pour dessiner l'objet `Image` dans un `Graphics`, on peut utiliser la méthode `drawImage(Image img, int x, int y, ImageObserver observer)`
- Le plus simple est de faire que chaque fenêtre affichant un ou plusieurs chronomètres soit une `JFrame` et contienne un `JPanel` par chronomètre
- Le texte de l'horloge numérique (nom + valeur) peut être affiché dans un `JLabel`
- Pour afficher le nom du chronomètre dans un affichage avec un cadran, on peut utiliser la méthode `drawString(String str, int x, int y)` sur un `Graphics` (celui du `JPanel` contenant l'`Image` du cadran)
- Pour le panneau de contrôle des chronomètres, il peut être intéressant de faire un `JPanel` par « ligne »
- Pour positionner les éléments dans les `JFrame` et `JPanel`, il est souvent suffisant d'appeler la méthode `setLayout(LayoutManager mgr)` en lui passant en paramètre un layout adapté (`FlowLayout`, `GridLayout`, …).
  C'est notamment le cas pour positionner les éléments dans le panneau de contrôle et pour que les chronomètres se repositionnent lors d'un redimensionnement des fenêtres en contenant plusieurs
- Pour traiter les clicks sur les boutons, cela se fait dans la méthode `actionPerformed` d'un `ActionListener` que l'on abonne à l'objet `JButton`. On peut le faire comme ci-dessous en créant une classe anonyme :

```
JButton xxx = new JButton("…");
xxx.addActionListener(new ActionListener() {
   @Override
   public void actionPerformed(ActionEvent ae) {
     /* code à exécuter après un click sur le bouton xxx */
   }
});
```

A réaliser

Par groupe de 2

A remettre

- Le diagramme de classes
- Le code source

Travail à rendre le 22.04.2021 à 15h45 sur Cyberlearn

Annexes

Affichage de 3 chronomètres après redimensionnement de la fenêtre pour avoir un affichage vertical.