TP7 – Les langages de programmation

Les acteurs en Scala

UV IDL

Correction

Objectifs

À travers le problème des lecteurs rédacteurs, nous allons découvrir les acteurs de scala.

Problème des Lecteurs-Rédacteurs

Une base de données est accédée par des lecteurs et par des rédacteurs. Plusieurs lecteurs peuvent accéder à la base en même temps. Un seul rédacteur peut écrire à la fois. Il y a exclusion entre lecteurs et rédacteurs. Les rédacteurs sont prioritaires. Cela signifie qu'un lecteur n'obtient le droit d'accès à la ressource que si aucun rédacteur n'y accède ou n'a envie d'y accéder. Il n'y a cependant pas préemption de la ressource par les rédacteurs lorsque celle-ci est occupée par des lecteurs.

Interface graphique utilisée

L'interface graphique (cf figure 1) se compose de deux boutons (« Démarrer »et « Quitter ») et de trois zones contenant chacune une liste de *threads* :

- la liste de lecteurs en attente,
- la liste des lecteurs ou du rédacteur se trouvant à l'intérieur de la base de données.
- et la liste de rédacteurs en attente.

Lorsque vous appuyez sur « Démarrer », 5 *threads* sont créés et entrent en compétition pour utiliser la base de données (3 lecteurs et 2 rédacteurs). Le bouton « Quitter »permet de quitter l'application.

FIGURE 1 – Interface graphique de l'application

Classes pour l'interface homme-machine

L'objet `Main` crée l'interface graphique et met à jour les listes de lecteurs et de rédacteurs en attente ou dans la base de données.

```
package readerWriter
import scala.swing._
import scala.swing.event._
import mvc._
import java.awt.Color
object Main extends SimpleSwingApplication {
  def top = new MainFrame {
    title = "Lecteur Redacteur"
    preferredSize = new Dimension(600, 300)
    val listeLecteur = new ListModel[String]
    val listeRedacteur = new ListModel[String]
    val listeBD = new ListModel[String]
    val startButton = new Button("Demarrer")
    val quitButton = new Button("Quitter")
    val pReader = new BorderPanel {
      preferredSize = new Dimension(200, 200)
      background = Color.GREEN
      add(new Label("Lecteurs en attente"), BorderPanel.Position.North)
      add(Component.wrap(new TextView(listeLecteur)), BorderPanel.Position.Center)
    }
    val pWriter = new BorderPanel {
      preferredSize = new Dimension(200, 200)
      background = Color.BLUE
      add(new Label("Redacteurs en attente"), BorderPanel.Position.North)
      add(Component.wrap(new TextView(listeRedacteur)), BorderPanel.Position.Center)
    }
    val pBD = new BorderPanel {
      preferredSize = new Dimension(200, 200)
      background = Color.YELLOW
      add(new Label("A l'interieur de la BD"), BorderPanel.Position.North)
      add(Component.wrap(new TextView(listeBD)), BorderPanel.Position.Center)
    }
    contents = new BorderPanel {
      add(startButton, BorderPanel.Position.North)
```

```
      add(quitButton, BorderPanel.Position.South)
      add(pReader, BorderPanel.Position.West)
      add(pBD, BorderPanel.Position.Center)
      add(pWriter, BorderPanel.Position.East)
    }
    listenTo(startButton)
    reactions += {
      case ButtonClicked(b) =>
        if (b.equals(startButton)) {
          println("starting")
        }
    }
    listenTo(quitButton)
    reactions += {
      case ButtonClicked(b) =>
        if (b.equals(quitButton))
          exit(0)
    }
  }
}
```

Exercice 1

▷ **Réalisez les classes nécessaires.**