

Interfaces Graphiques

« La concurrence est rude mais nécessaire,
nul ne doit y échapper... »

Proverbe ultra-libéral

Jean-Baptiste.Yunes@univ-paris-diderot.fr

Université Paris Diderot

©2014



- Swing n'est pas thread-safe!!!
- pour en savoir plus, lisez « Swing's Threading Policy »
- In general **Swing is not thread safe**. All Swing components and related classes, unless otherwise documented, must be accessed on the event dispatching thread.

- Réactivité
- Concurrency
 - threads *initiaux*
 - thread principal de l'interface (distribution)
 - threads annexes de travail
- Ces threads sont automatiquement créés...

Thread principal



- interface réactive
 - à réception d'événements le thread principal ne doit pas exécuter de code trop long
 - sinon, il faut utiliser des threads de travail

- Threads initiaux
 - Application
 - un seul thread appelant `main ()`
 - Applets
 - éventuellement plusieurs threads appelant
 - `init ()`
 - `start ()`

- dans ces threads initiaux
 - on initialise l'application
- l'initialisation de l'interface doit être réalisée sous le contrôle du thread principal
 - créer un Runnable
 - demander à l'exécuter par le thread principal
 - `SwingUtilities.invokeLater(Runnable)`
 - `SwingUtilities.invokeAndWait(Runnable)`
 - une simple encapsulation des méthodes correspondantes de `java.awt.EventQueue`

- Attention pour une `Applet`
- la création de l'interface doit s'effectuer dans la méthode `init`
- elle doit être terminée avant le retour de celle-ci
- nécessaire d'utiliser `invokeAndWait`

- Threads de travail
 - s'occupent de réaliser des tâches longues
 - sont en arrière-plan
 - une telle tâche peut être représentée par une instance de `SwingWorker`

- mécanismes
 - exécution d'une méthode à la terminaison
 - fournit un résultat asynchrone (un futur)
 - peut exécuter du code sous contrôle du thread principal