

Projet de Programmation

Benoit Donnet
Année Académique 2019 - 2020



1

Agenda

Partie 3: Eléments de Programmation Evénementielle

- Chapitre 1: Introduction aux Interfaces Graphiques
- Chapitre 2: Applications Interactives
- Chapitre 3: Pattern MVC

Agenda

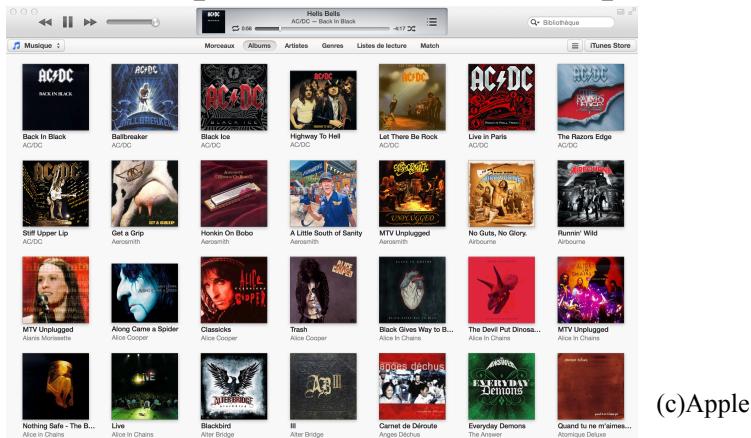
- Chapitre 2: Applications Interactives
 - Principes
 - Structure d'une Application Interactive

Agenda

- Chapitre 2: Applications Interactives
 - Principes
 - Structure d'une Application Interactive

Principes

- **Application interactive?**
 - application avec laquelle l'utilisateur peut interagir
- L'application effectue des opérations en réponse aux actions de l'utilisateur
- Coopération entre le programme et l'utilisateur
 - coopération commandée par l'utilisateur



(c)Apple

Principes (2)

- Programmer une application interactive peut s'avérer complexe car
 - les tâches que l'utilisateur peut effectuer peuvent être complexes
 - il faut prévoir des scénarios d'interaction
 - ✓ et donc les réactions de l'application
 - il faut pouvoir maintenir et réutiliser
- Problème de passage à l'échelle
 - une application interactive est souvent une application importante

Principes (3)

- On dispose de concepts
 - *structuration "formelle"* des applications interactives
 - ✓ front office
 - ✓ back office
 - *modèles d'applications interactives*
 - ✓ modèle-vue-contrôleur
 - › MVC
 - ✓ autres design patterns
- On dispose d'outils
 - langages/environnements de programmation adapté
 - librairies et APIs
 - ✓ GTK

Agenda

- Chapitre 2: Applications Interactives
 - Principes
 - Structure d'une Application Interactive
 - ✓ Principe
 - ✓ Front Office
 - ✓ Back Office
 - ✓ Liaison

Principe

- La structure d'une application interactive est composée de 2 parties
 1. la partie *visible*
 - ✓ **front office**
 - ✓ ce que l'on fait et ce que l'on voit
 - ✓ IHM
 2. la partie *invisible*
 - ✓ **back office**
 - ✓ ce qu'il se passe
 - ✓ toutes les opérations
 - traitements
 - stockage des données
 - communication

Front Office

- L'utilisateur commande l'application à travers l'IHM
 - l'application doit être adaptée à la tâche
 - l'application doit être *réactive*
 - ✓ pas de traitements trop longs
 - ✓ ou alors, il faut notifier l'utilisateur
- Ce que l'on fait correspond aux *entrées*
- Ce que l'on voit correspond aux *sorties*

Front Office (2)

- Les entrées



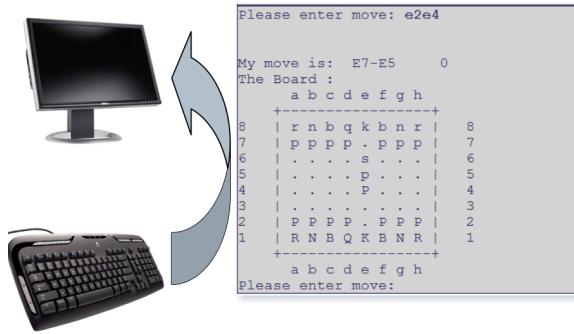
Front Office (3)

- Les sorties

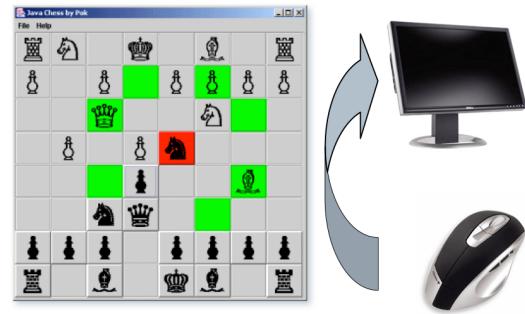


Front Office (4)

- Exemple: jeu d'échec
 - sous-tâches: déplacer les pièces
- Interfaces utilisateur



ligne de commande



graphique

Back Office

- *Noyau* de l'application
- Gère
 - les fonctionnalités
 - les accès aux données
 - le traitement des données
- Produit les résultats aux actions de l'utilisateur

Back Office (2)

- Exemple: jeu d'échec
 - fonctionnalités
 - ✓ jouer une partie
 - déplacer des pièces
 - gérer les tours du jeu
 - joueur virtuel
 - ...
 - ✓ enregistrer une partie
 - ✓ charger une partie
 - données
 - ✓ état de la partie en cours
 - ✓ parties sauvegardées
 - ✓ ...



Liaison

- Comment lier les deux parties de l'application?



Liaison (2)

- Méthode 1

- en vrac
- on gère les choses comme on peut
- on assemble les différentes parties en fonction de notre intuition du moment



Liaison (3)

- Méthode 2:

- programmation selon un modèle organisé
- *MVC*

