Architecture d'une application

martine.gautier@univ-lorraine.fr


Jeu 421
3 coup(s) joué(s)
4
3
5
Jouer
Quitter

- Dans la fonction **start**, construire un panneau
  - un label en haut
  - trois boutons au centre
  - rois boutons en bas
- Ajouter les écouteurs sur chaque bouton
- Ajouter la logique du jeu
  - Mémoriser les valeurs des dés, le nombre de lancers

- La classe Main a trop de responsabilités.
- Aucune structuration : où sont les classes ?
- Une modification/évolution de l'interface graphique est un vrai casse-tête.
  - Ajouter un composant graphique
  - Remplacer un composant par un autre
  - Ajouter des fonctionnalités à l'application
- La logique du jeu (logique métier) est noyée dans l'interface graphique.

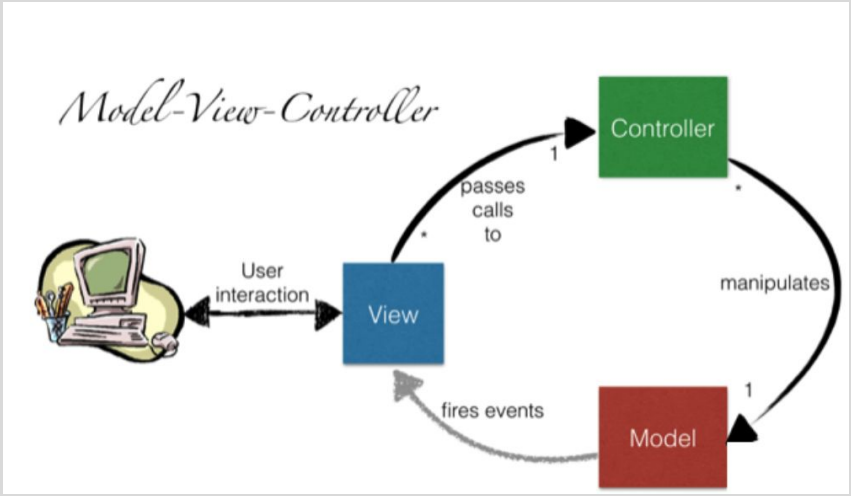
Evolution ?

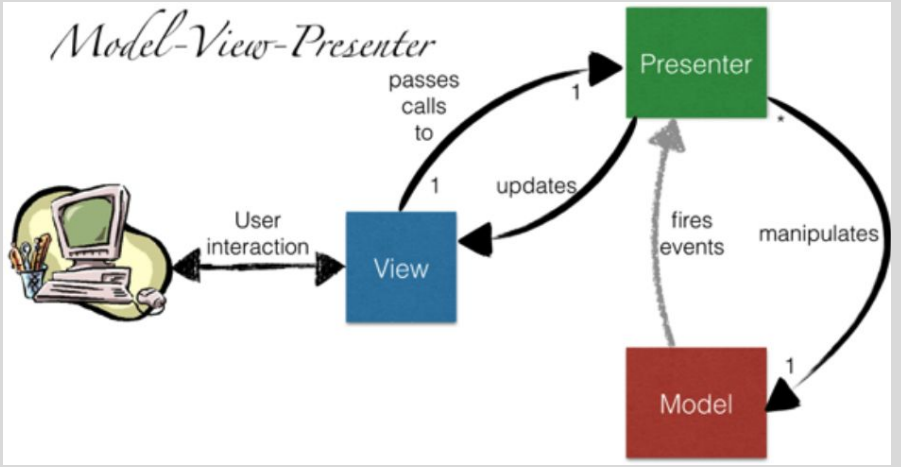
- Application grandeur nature
  - Beaucoup de composants
  - Actions faisant évoluer les données et l'IG

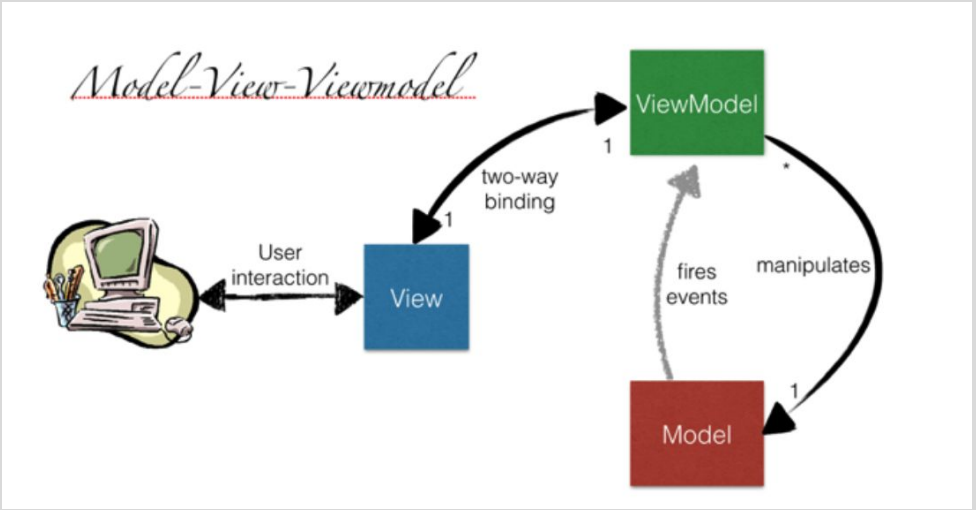
- Structuration indispensable
  - Faciliter le développement
    - … itératif
    - … collaboratif
  - Favoriser l'évolutivité

- Concentrer l'IG dans une classe n'a pas de sens …

Architectures









- Application graphique = Données + Composants réactifs
- Points communs entre ces 3 architectures =
  - Le modèle de données reste indépendant des composants
  - Les vues sont des objets à part entière
- L'IG peut évoluer (ajout/remplacement de composants) sans conséquence sur le modèle

Architecture Model-View-Whatever ?



Principe général

L'utilisateur agit sur l'interface graphique.

Le contrôleur attaché au composant se réveille.

Il transmet l'action au modèle.

Le modèle transforme ses données.

Il notifie toutes les vues qu'il est modifié.

Lorsqu'une vue est notifié d'un changement, elle peut (ou non) se rafraîchir.