Jouer aux cartes

S'il vous est déjà arrivé de jouer aux cartes, vous avez probablement :

- ajouté des cartes dans votre main,
- trié ces cartes selon un critère bien à vous,
- joué les cartes, soit une par une comme à la *belote*, soit par plusieurs d'un coup comme dans le jeu du *président* par exemple.

Dans ce DM vous allez créer un simulateur de Main, c'est à dire un outil qui permette de créer une main (piocher des cartes), jouer une ou plusieurs cartes, trier les cartes..

Mise en place

Vous allez devoir écrire deux classes :

- `Carte.java` permettant de modéliser une carte (par sa valeur et sa couleur),
- `Main.java` permettant de modéliser une main, c'est à dire une liste de cartes et des méthodes associées.

Pour que ce soit "joli", on vous donne du code qui gère l'affichage. Récupérez le fichier zip suivant : sourcePourEtuVide.zip

Il contient :

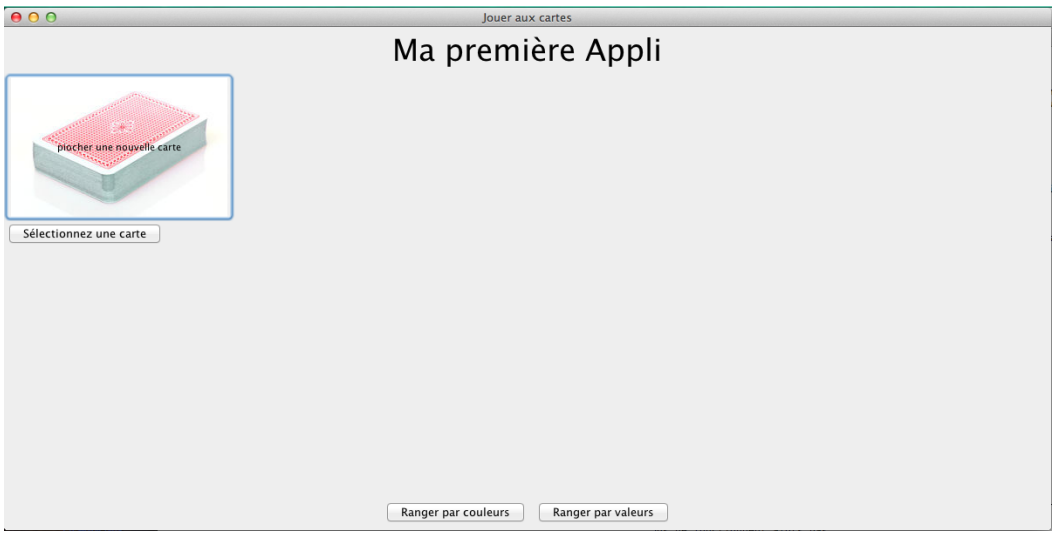
- `Executable.java` : l'exécutable,
- `Controleur.java` : inutile de le regarder. C'est la classe qui s'occupe de faire le lien entre vos classes (Main et Carte) et l'affichage : lorsque par exemple vous cliquez sur le bouton "trier carte", le controleur s'occupe de 1) appeler la méthode de tri des cartes de la classe Main (que vous écrirez) et de 2) demander à la vue de mettre à jour l'affichage.
- `Vue_JeuDeCartes.java` : inutile de le regarder. C'est la classe qui gère l'affichage, la création des boutons, etc.
- un dossier `images` contenant les images utiles à l'affichage,
- un dossier `doc` contenant des informations sur les classes que vous devez implémenter. Vous ouvrirez le fichier `index.html` pour lire cette documentation.

Les fichiers `Carte.java` et `Main.java` n'existent pas, vous ne pouvez donc pas encore compiler ! En regardant la documentation, vous verrez qu'il faut que la classe Carte contienne :

- Un constructeur : `public Carte(int valeur, String couleur)`,
- une méthode `public String getCouleur()`,
- une méthode `public String getValeur()`,
- une méthode `public int getValeurInt()`.

1. Créez le fichier `Carte.java` contenant ce code.
2. Pour la classe `Main.java`, trouvez les méthodes à implémenter dans la documentation. Créez un fichier `Main.java` avec les définitions de méthodes, mais en mettant du code *qui ne fait rien* (souvent pas de code) dans le corps des méthodes. On vous guidera dans la suite pour l'ajout des fonctionnalités
3. Compilez. Exécutez en utilisant la commande `java -ea Executable`. L'option `-ea` (*enable assertions*) permet d'activer les assertions, et donner des messages d'erreur supplémentaires.

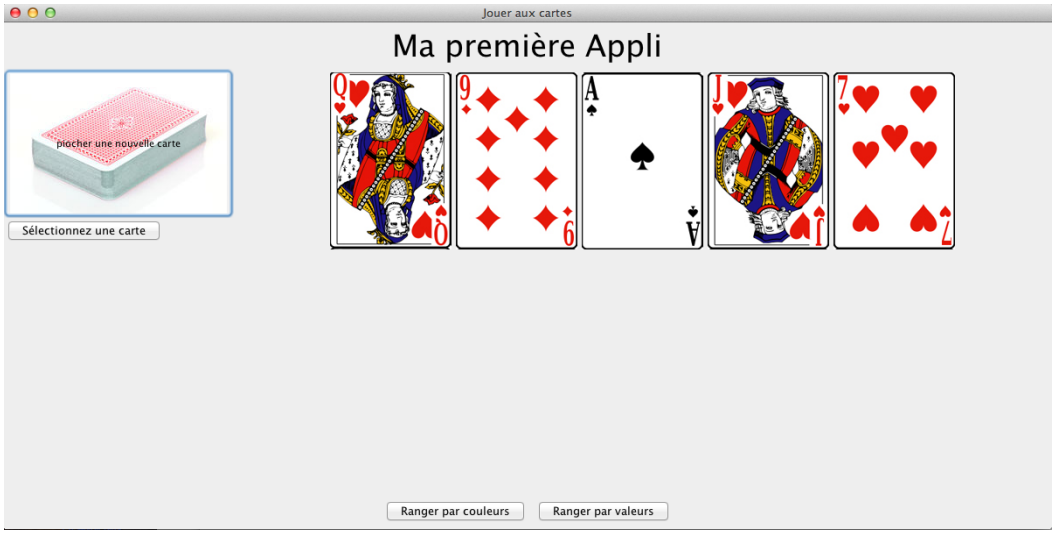
Vous devez avoir un affichage ressemblant à la capture suivante. Les boutons ne fonctionnent alors pas.



Ajout de fonctionnalités

1. Ajoutez le code pour que les méthodes `add` et `getMesCartes` de Main soient fonctionnelles. Pour cela, vous aurez besoin d'un attribut de type `ArrayList<Carte>` qu'il faudra instancier dans le constructeur de Main.

Vous devez alors avoir quelque chose comme :



2. Dans l'ordre, écrivez les méthodes :
   - `setCarteSelectionee`,
   - `getCarteSelectionnee`,
   - `jouer` (utilisez la méthode remove des ArrayList)
   - `jouerToutesValeurs` et `jouerToutesCouleurs`. Attention, ne supprimez pas une valeur d'une liste que vous êtes en train de parcourir. Vous pouvez par exemple commencer par créer la liste des cartes à supprimer, puis supprimez ces cartes. Ou bien vous pouvez commencez par créer la liste des cartes à garder, puis assignez cette valeur à la liste des cartes de votre main.
3. La méthode piocher doit renvoyer une carte choisie aléatoirement. Écrivez cette méthode; pour cela, inspirez vous du code suivant :

```
Random generateurDeNombresAleatoires = new Random();
int nombre = generateurDeNombresAleatoires.nextInt(12) + 1;
// ici nombre vaut un nombre aléatoire choisi entre 1 et 12
```

4. Écrivez les méthodes `triParValeurs` et `triParCouleur`. Pour ces deux dernières méthodes vous pouvez soit 1) écrire votre propre code de tri, soit 2) utiliser la bibliothèque en implémentant l'interface Comparator .