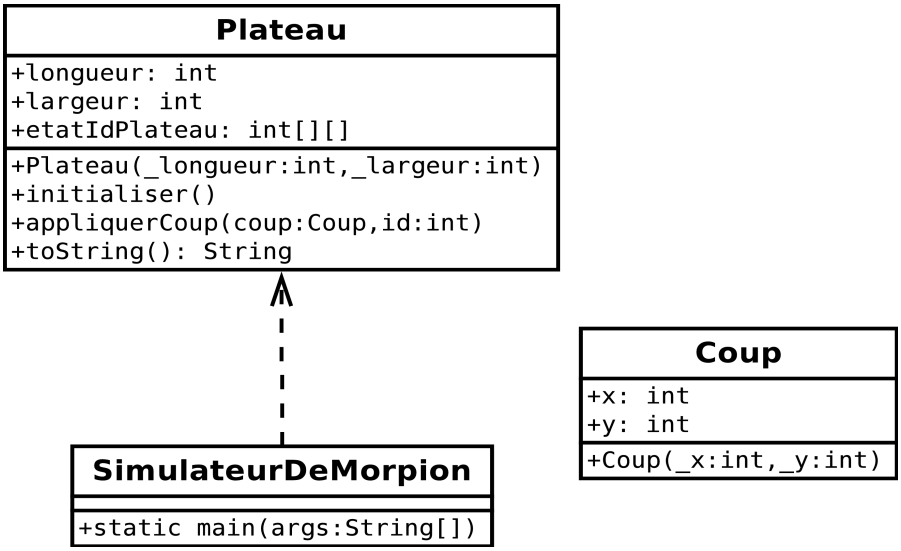
TP3 : Héritage et Polymorphisme – Partie 1

Exercice 1 : Plateau

Soit le diagramme ci-dessous. On choisit de représenter le plateau comme un tableau d'entier : 0 correspond à une case vide, sinon on indique l'identifiant du joueur (entier strictement supérieur à 0).



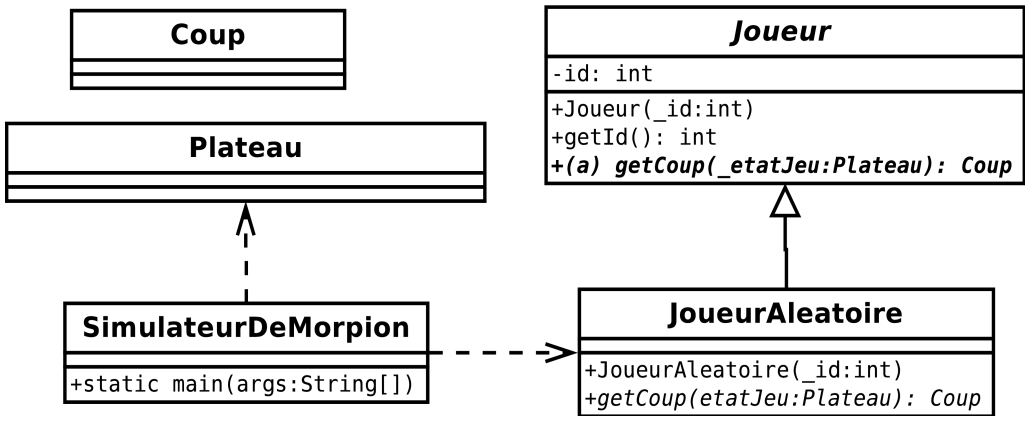
Remarque : la flèche en pointillés (flèche ouverte) est une relation de dépendance, à ne pas confondre avec la flèche représentant l'implémentation (flèche fermée)
Proposez une implémentation. La méthode « main » devra par exemple afficher un plateau 3x3 vide.

Exercice 2 : Joueur aléatoire

Partie 2A
Soit la fonction *random()* de la classe Math (librairie standard, accès par navigateur => rechercher « java api Math »).

Développer la fonction statique monRandom(int max, int min), de votre classe Tool, qui retourne un nombre aléatoire compris entre max et min, max et min inclus.

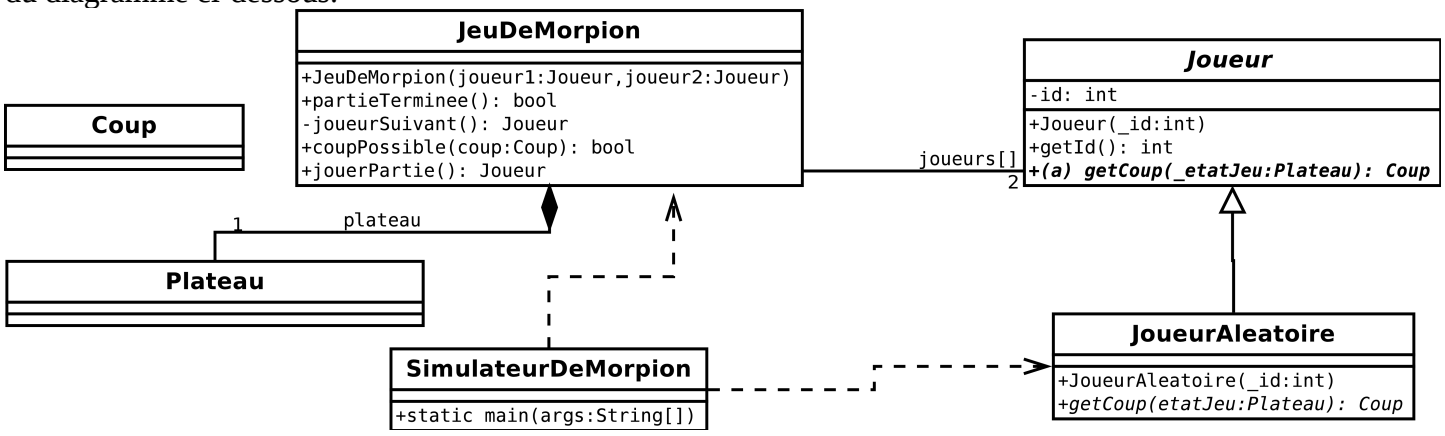
Partie 2B
Aidez vous du diagramme de classe pour implémenter un joueur qui génère aléatoirement des coups sur un plateau (on veillera à ce que « getCoup » ne retourne pas une case déjà occupée).



Remarque : (a) sur le diagramme signifie que la fonction est abstraite (équivalent à une écriture en Italique)

Exercice 3 : Jeu de Morpion

Implémentez la classe JeuDeMorpion qui assure le bon déroulement d'une partie de Morpion en vous inspirant du diagramme ci-dessous.



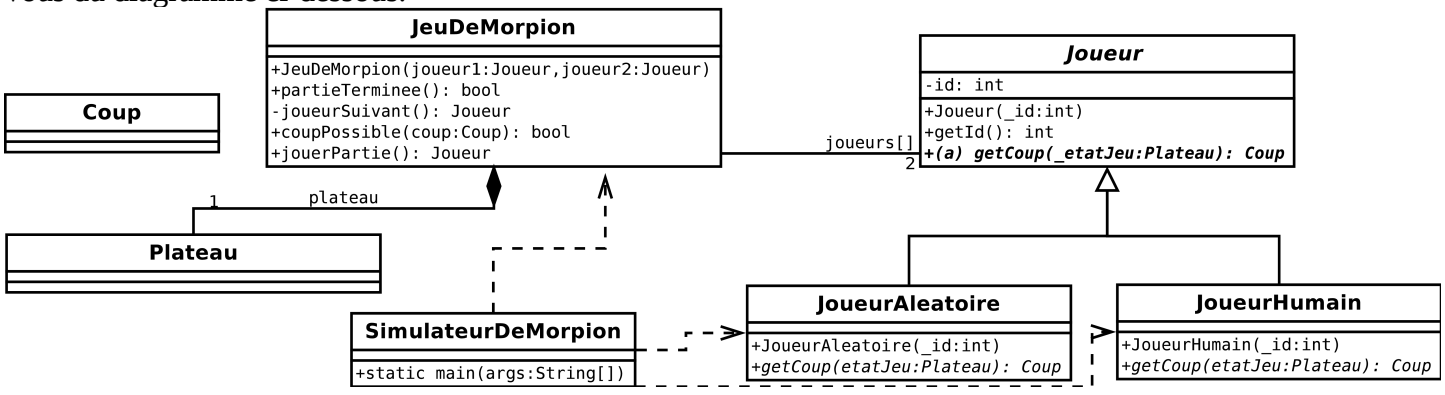
Soit la définition de la fonction **jouerPartie** de la classe **JeuDeMorpion** :

```
public Joueur jouerPartie() {
      Joueur retour = null; // utilisé comme variable de parcours,et renvoyé comme étant
                            // le joueur gagnant
      while (!partieTerminee()) {
            retour = getJoueurSuivant();
            Coup c = retour.getCoup(plateau);
            if (coupPossible(c))
                  plateau.appliquerCoup(c,retour.getId());
            else
                  System.err.print(« erreur ») ;
                  System.exit(1) ;
      }
      return retour;
}
```

Remarque : dans un premier temps, la fonction partieTerminee() pourra simplement vérifier qu'il n'y a plus aucune case vide sur le plateau.

Exercice 4 : Polymorphisme des joueurs

Implémentez un joueur humain qui devra à chaque tour saisir les coordonnées de son prochain coup. Inspirez vous du diagramme ci-dessous.



Remarque : pour la saisie de donnée, on utilisera java.util.Scanner pour lire des entiers à partir « System.in »