|  |
| --- |
| BERCY - JOULIN |
| Jeu de la grenouille |
| TP de programmation |
|  |
| **INFO1** |
| **2012** |

|  |
| --- |
|  |

Table des matières

[2 Présentation 4](#_Toc355371027)

[2.1 Présentation du jeu 4](#_Toc355371028)

[3 Composantes du jeu 4](#_Toc355371029)

[3.1 Personnages 4](#_Toc355371030)

[3.2 Zone de jeu 5](#_Toc355371031)

[3.3 Joueurs 6](#_Toc355371032)

[4 Architecture du jeu 7](#_Toc355371033)

[5 Intelligence artificielle 7](#_Toc355371034)

[5.1 Prévision des déplacements 7](#_Toc355371035)

[5.1.1 Recherche des meilleurs mouvements (pendant la partie) 7](#_Toc355371036)

[5.1.2 Recherche des meilleures mouvements (avant la partie) 8](#_Toc355371037)

[5.1.3 Chances de gagner 8](#_Toc355371038)

[5.2 Solution implémentée 9](#_Toc355371039)

[5.3 Stratégies 9](#_Toc355371040)

# Présentation

## Présentation du jeu

Le jeu oppose deux joueurs et consiste à déplacer des grouilles et des crapauds sur un plateau quadrillé de N lignes et M colonnes. En début de partie, les grenouilles sont attribuées à l’un des joueurs, les crapauds à l’autre. Le détenteur des grenouilles commence, puis les joueurs jouent à tour de rôle. Le premier qui ne peut plus avancer ses batraciens a perdu.  
Les grenouilles avancent horizontalement de la gauche vers la droite, et les crapauds de la droite vers la gauche.  
Si la case adjacente dans le sens de déplacement est libre, la grenouille (ou le crapaud) déplacé(e) vient occuper cette case. Si, par contre, elle est occupée par un crapaud (ou une grenouille) et que la case suivante est libre, le batracien vient occuper cette case libre. Dans tous les autres cas, le batracien ne peut être déplacé.

# Composantes du jeu

## Personnages

Deux solutions permettent de représenter les personnages : la première consiste à créer une classe pour chacun d’entre eux, la deuxième consiste à utiliser une seule classe comportant une propriété qui indique le type de personnage.

|  |  |
| --- | --- |
| Image 3.1 - Schéma de la première solution  Personnage  Grenouille  Crapaud | Dans le schéma ci-contre, Personnage est une classe abstraite, héritée par les classes Grenouille et Crapaud.  Avantages :   * L’architecture reflète la réalité. La classe Personnage a été nommée ainsi pour son application dans le contexte du jeu, mais aurait pu se nommer « Anoures » pour faire référence à la famille des grenouilles et des crapauds dans la nature.   Inconvénients :   * La mise en pratique d’une telle architecture de code conduit à écrire la totalité du code dans la classe abstraite, ou à écrire du code similaire dans les deux classes dérivées. * L’utilisation de cette architecture complique les interactions avec les classes. Par exemple, pour connaître le type d’un personnage donné dans la zone de jeu, il est nécessaire d’avoir recours à instanceOf dans une partie du code où il pourrait être possible de travailler à un plus haut niveau (et par conséquent de s’abstraire de ces constructions de bas niveau). Une alternative serait d’implémenter dans les classes la possibilité de les interroger sur leur type, au quel cas l’intérêt de les séparer se verrait réduit. |

|  |  |
| --- | --- |
| Image 3.2 - Schéma de la seconde solution  Personnage  type  *C*  *G* | Dans cet autre schéma, la classe Personnage est à la fois utilisée pour représenter les grenouilles et les crapauds. Le type de personnage peut par exemple être définit lors de la construction d’un objet, et récupéré via un accesseur.  Avantages :   * Architecture plus simple * Transparence complète pour le code du jeu au plus haut niveau * Création plus simple des personnages   Inconvénients :   * Il est plus difficile de rajouter de nouveaux personnages. Dans le cas du jeu de la grenouille, il est certain que le nombre de personnages sera toujours de deux, le problème ne se posera donc jamais. |

## Zone de jeu

La zone de jeu (« Etang ») est représentée par un tableau 2D de références vers des batraciens.

## Joueurs

Pour chacun des deux types de personnage (grenouilles et crapauds), deux types de joueurs sont possibles (humain ou virtuel). Afin de garantir le maximum de flexibilité, les deux types de joueurs doivent être interchangeables de façon totalement transparente au reste du jeu. Le polymorphisme permet d’arriver à cet idéal.

Image . - Gestion des joueurs

Humain

IA

Joueur grenouilles

Humain

IA

Joueur crapauds

Ce que l’Image 2.4 vise à montrer, est que seuls les systèmes « joueur grenouilles » et « joueur crapauds » sont exposés au reste du jeu. Aucun des autres systèmes n’a besoin de s’avoir si un personnage donné est dirigé par un humain ou par une intelligence artificielle.

# Intelligence artificielle

(A faire)

# Architecture du jeu

(A faire : schéma)

# Utilisation

(A faire : guide d’utilisation du jeu)