

Projet MDD :

Document de

Conception : Jeu Snakes



Pierre-Louis OGER (p2oger@enib.fr)
Anthony GUILLIER (a2guilli@enib.fr)

Liste des fonctionnalités :

- Créer map
- Afficher map
- Créer snake
- Afficher snake
- Déplacer snake
- Ressuscité snake
- Créer bonbon
- Manger bonbon
- Gagner Partie
- Sauvegarder partie
- Charger partie
- Créer partie

Arbre d'appel des fonctions :

main

mChargerfichier.ChargerFichier

mPa.creer
mPa.setForme
mPa.setTailleRect
mPa.setTailleTrian
mPa.setTailleCroix
mPa.setNomsJoueurs
mPa.setCouleursJoueurs
mPa.setVitesseJeu
mPa.setFinJeu

creerPartie

mParametre.getVitesseJeu
mParametre.getForme
mParametre.getTailleRect
mMap.creerRectangulaire
 creerEncadrement
mParametre.getTailleTrian
mMap.creerTriangulaire
 creerRectangulaire
 creerEncadrement
mMap.creerCroix
 creerEncadrement
mMap.getEtatCase
 mCase.getCase
mSnake.creerSnakeInit
mJoueur.creerJoueur
mJeu.creer
mJeu.getMap
mMap.getTailleEcran

mGraphique.initGraphique
mChargerFichier.chargerPara

jouer

mJoueur.getSnake
mJeu.getJoueur
mSnake.getxSnake
mSnake.getySnake
mMap.setCaseC
mSnake.deplacerSnake1
 getxSnake
 getySnake
 mJoueur.setSnakeJ
mSnake.deplacerSnake2
 getxSnake
 getySnake
 mJoueur.setSnakeJ
mMap.sortiCarte
 getEtatCase
 mCase.getCase
 mCase.getCase
 mSnake.setSnake
 mJoueur.setSnakeJ
 mSnake.getxSnake
 mSnake.getySnake
mSnake.imprimerSnake
 mMap.setCaseC
mGraphique.afficherMap
 mJeu.getMap
 mJoueur.getCouleur
 mCase.getCase
mMap.affichageMap
 mJeu.getJoueur
mMap.bonbon
 mSnake.appSnake
 mGraphique.playsound
 getEtatCase
 mCase.getCase
 setCaseC
 mCase.setCase
mMap.collisionSnakes
 mSnake.ressusciteSnake
 mMap.setCaseC
 mJoueur.setSnakeJ
mMap.collisionSnake
 mSnake.ressusciteSnake
 mMap.setCaseC
 mJoueur.setSnakeJ

finPartie

mParametre.getFinJeu
mJoueur.getSnake
mJoueur.getNom

mSnake.clavierSnake1
mSnake.clavierSnake2
mChargerFicher.sauverPara
mParametre.getVitesseJeu

Versions:

Version 1 :

Cette version permet d'afficher les 2 snakes et de déplacer un snake sur une carte rectangulaire.

Fonctionnalités mises en œuvre :

- Créer map
- Afficher map
- Créer snake
- Afficher snake
- Déplacer snake

Version 2 :

Cette version permet de déplacer 2 snakes et de relier les bords de la carte.

Fonctionnalités modifiées :

- Créer map
- Créer snake
- Afficher snake
- Afficher map

Version 3 :

Cette version permet de ressuscité un snake et sauvegarder/charger une partie dans un fichier, ramasser un bonbon/gagner et choisir sa carte.

Fonctionnalités mises en œuvre :

- Sauvegarder partie
- Ressuscité snake
- Créer bonbon
- Manger bonbon

- Charger partie
- Gagner partie

Fonctionnalités modifiées :

- Charger partie
- Créer map
- Afficher map

Types abstraits :

Jeu caractérisé par :

- Une map
- 2 joueurs
- Une vitesse de déroulement

Map caractérisé par :

- Un tableau de cases

Case caractérisé par :

- Une chaîne de caractère représentant ce qui se trouve sur la case (Snake 1, Snake 2, Un bonbon, Vide, Rien (hors du terrain de jeu))

snake caractérisés par :

- Une liste de positions(x,y)

joueur caractérisés par :

- Un nom
- Une couleur
- Un snake

parametre caractérisé par :

- forme de la map
- les tailles des maps
- le nom des joueurs
- la couleur des joueurs
- la vitesse du jeu
- le score de fin du jeu
- les deux snakes
- les deux derniers déplacements de chacun des snakes

Ce type sert à représenter le fichier lorsqu'il est chargé et également le chargement d'une partie sauvegarder. Il contient donc tous les paramètres nécessaires à caractériser une partie.

Fonctions associées aux types abstraits :

Type **Case** :

Constructeur :

creer : str → case

Prend en entrée l'état de la Case et renvoi une case.

Accesseurs :

getCase : case → str

Renvoie l'état de la case codé par une str.

setCase : str, case → rien

Permet de modifier l'état d'une case.

getCaseX : X, case → bool

Renvoie True si la case est du type X et False sinon.

Type **map** :

Constructeur :

creerRectangulaire : int, int → map

Construit une map rectangulaire à partir de la hauteur et la largeur.

creerTritangulaire : (int, int), (int, int), (int, int) → map

Construit une map triangulaire à partir de trois points

creerCroix : int → map

Construit une map en croix à partir de la taille du côté du carré hébergeant la croix.

CreerEncadrement : map → map

Rajoute un encadrement de cases vides autour de la carte.

Accesseurs :

getCaseC : carte, (int, int) → str

Renvoie l'état de la case (x, y)

setCaseC : carte, (int,int), str → rien

Change l'état de la cas (x,y).

getTailleEcran : carte → str,str

Obtenir la taille de l'écran (hauteur,largeur)

affichageMap : carte → carte

Affichage de la map « non graphique »

sortiCarte : snake1, list,carte,joueur,int,int → snake,joueur

Détection et rebouclage des bords de la map

bonbon : int, int, int, int, snake, joueur, int, int, carte, int → snake, joueur, int, int

Detection si on est sur un bonbon et réactualisation de la map

CollisionSnakes : snake, int, int, int, int, snake, joueur, joueur, jeu, int, str, str → str, str
Detection si collision entre les 2 snakes.

CollisionSnake : snake, int, int, joueur, jeu, int, str, str → str, str
Detection si collision de son propre snake.

Type **jeu** :

Constructeur :

creer : vitesse, map, joueur1, joueur2 → jeu
Fonction qui crée tous les éléments de la partie à partir du fichier .xml .

Accesseurs :

getMap : jeu → map
getJoueur : jeu, int → joueur
getVitesse : jeu → int

Type **parametre** :

Constructeur :

créer : rien → parametre
Renvoie le type parametre avec des données nuls.

Accesseurs :

setForme :	parametre, str	→ rien
setTailleRect :	parametre, (int, int)	→ rien
setTailleTriang :	parametre, (int, int), (int, int), (int, int)	→ rien
setTailleCroix :	parametre, int	→ rien
setNomsJoueurs :	parametre, str, str	→ rien
setCouleursJoueurs :	parametre, str, str	→ rien
setVitesseJeu :	parametre, int	→ rien
setFinJeu :	parametre, int	→ rien
setSnakes :	parametre, snake, snake	→ rien
setDepSnakes :	parametre, list, list	→ rien
getForme :	parametre	→ str
getTailleRect :	parametre	→ list(int, int)
getTailleTriang :	parametre	→ list((int, int), (int, int), (int, int))
getTailleCroix :	parametre	→ int
getNomjoueur :	parametre, int	→ str
getCouleurJoueur :	parametre, int	→ str
getVitesseJeu :	parametre	→ int
getFinJeu :	parametre	→ int
getSnakes :	parametre	→ snake, snake

getDepSnakes : parametre → list, list

Opérateur :

ConvertirEnInt : str → int

Type **Joueur** :

Constructeur :

creerJoueur : str, str, list → joueur

Créer un joueur à partir de sa couleur son nom et son snake.

Accesseurs :

getCouleur : joueur → str

Renvoie la couleur du joueur

getNom : joueur → str

Renvoie le nom du joueur

getSnake : joueur → list

Renvoie le snake du joueur

setSnakeJ : joueur, list → joueur

Met en place le snake d'un joueur

Type **Snake**:

Constructeur:

creerSnake : int, int → snake

Créer un snake à partir de sa position (x,y).

creerSnakeInit : int, int → snake

Créer un snake à partir de sa position initiale (x,y).

Accesseurs :

getXSnake : snake, int → int

Récupération de x d'une partie d'un snake

getYSnake : snake, int → int

Récupération de y d'une partie d'un snake

setSnake : snake, int, int, int → joueur

Mise en place du snake

imprimerSnake : carte, snake, int → rien

Imprimer le snake sur la carte « non graphique ».

deplacerSnake1: list, snake1, joueur1 → list
Déplacement du snake1 sur la carte

deplacerSnake2 : list, snake1, joueur1 → list
Déplacement du snake2 sur la carte

clavierSnake1 : list → rien
Lecture du clavier pour le snake 1

clavierSnake2 : list → rien
Lecture du clavier pour le snake 2

appSnake : snake,int,int → list
Rajoute une case au snake

ressusciteSnake : snake,joueur,jeu → snake
Ressuscite snake apres collision

Type **Graphique**:

initGraphique : int, int,fichier → screen
Initialisation de l'écran et du son

chargerFichierCouleurJoueur: str → str
Chargement de la couleur d'un joueur

afficherMap : jeu, screen → rien
Affichage graphique de la carte

playSound : rien→ rien
Joue le son

Type **Main** :

creerPartie : list → jeu,int,int,int,int
Crée la partie

jouer : list, jeu, screen, int, carte → list, int, int, snake, joueur
Met en mouvement les snakes, délimite le terrain, actualise la map

finPartie : str, parametres, joueur1, joueur2, str, str → str
Met fin au jeu lorsqu'on a gagné ou perdu