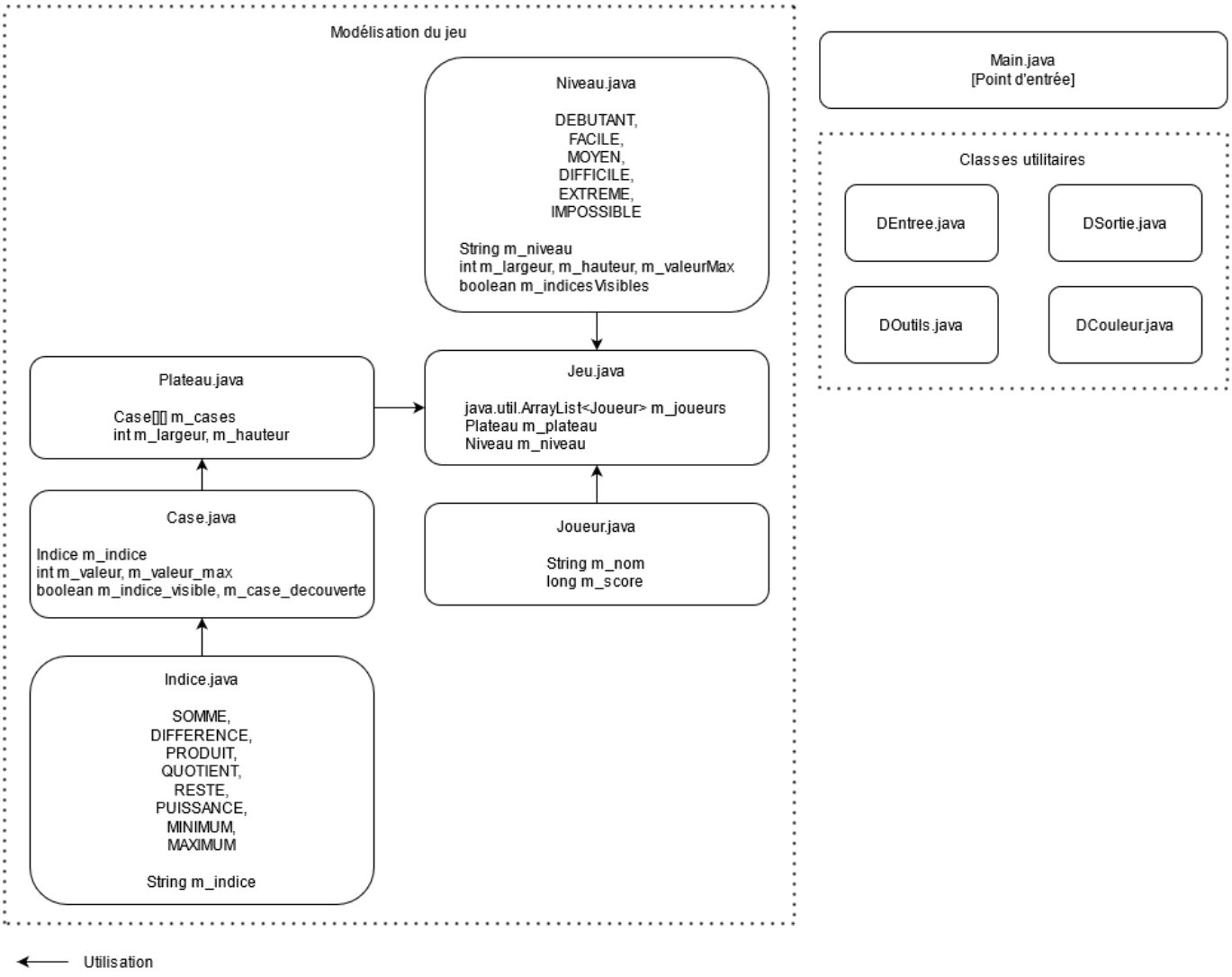
Guillaume BELDILMI
avec Ryan AIT CHEIKH
L1-IE2-31

Projet Info2A

Pré-rapport

Table des matières

Diagramme



Descriptions

Case.java

Cette classe sert à représenter une case du plateau.

Indice m_indice

Cet attribut servira à représenter l'indice que possède la case.

int m_valeur

Cet attribut sert à représenter la valeur de la case.

int m_valeur_max

Cet attribut sert à représenter la valeur maximale que peut prendre la case.

boolean m_indice_visible

Cet attribut sert à savoir si l'indice de la case doit être affiché ou non.

boolean m_case_decouverte
Cet attribut sert à savoir si la case a été découverte ou non.

Case(int max, boolean indiceVisible)
Il s'agit du constructeur de la classe. Pour créer un objet Case, il suffit de lui passer en paramètre la valeur maximale que peu prendre la case ainsi qu'un booléen ayant la valeur 'true' si l'indice de la case devra être affiché tant que la case n'est pas découverte.

void genCase()
Cette méthode est appelée par le constructeur et sert à attribuer un indice et une valeur aléatoires à la case.

String toString()
Cette méthode retourne une chaine de caractères pour représenter la case sous la forme '<valeur><indice>'. Si certains attributs de la case ne doivent pas être attribuer, ils seront remplacés par des caractères '#'.

void setIndice(Indice i)
Cet accesseur en écriture sert à attribuer un indice à la case, il sera stocké dans l'attribut 'm_indice'.

void setValeur(int v)
Cet accesseur en écriture sert à attribuer une valeur à la case, elle sera stockée dans l'attribut 'm_valeur'.

void setValeurMax(int vm)
Cet accesseur en écriture sert à attribuer une valeur maximale à la case, elle sera stockée dans l'attribut 'm_valeur_max'.

void setIndiceVisible(boolean iv)
Cet accesseur en écriture sert à spécifier l'état que doit prendre l'attribut 'm_indice_visible'.

void setCaseDecouverte(boolean d)
Cet accesseur en écriture sert à spécifier l'état que doit prendre l'attribut 'm_case_decouverte'.

Indice getIndice()
Cet accesseur en lecture sert à renvoyer l'indice de la case, stocké dans l'attribut 'm_indice'.

int getValeur()
Cet accesseur en lecture sert à renvoyer la valeur de la case, stockée dans l'attribut 'm_valeur'.

int getValeurMax()
Cet accesseur en lecture sert à renvoyer la valeur maximale de la case, stockée dans l'attribut 'm_valeur_max'.

boolean isIndiceVisible()
Cet accesseur en lecture sert à renvoyer l'état du booléen stocké dans l'attribut 'm_indice_visible'.

boolean isCaseDecouverte()
Cet accesseur en lecture sert à renvoyer l'état du booléen stocké dans l'attribut 'm_case_decouverte'.

boolean isValeur(int v)
Cette méthode sert à vérifier si la valeur passée en paramètre est égale à la valeur de la case, et renvoie ‘true’ si c’est bien le cas.

DCouleur.java
Cette énumération fait partie des clases utilitaires du projet, elle ne sert qu’à contenir les séquences d'échappement ANSI permettant de changer la couleur d’écriture du texte dans le terminal.

‘NOIR’, ‘ROUGE’, ‘VERT’, ‘JAUNE’, ‘BLEU’, ‘MAGENTA’, ‘CYAN’ et ‘BLANC’ permettent respectivement de changer la couleur du texte en noir, rouge, vert, jaune, bleu, magenta, cyan et blanc. Tandis que REINITIALISATION permet de réinitialiser la couleur du texte dans le terminal.

String m_couleur
Il s’agit de l’attribut qui permettra de stocker la séquence d’échappement de chaque élément de l’énumération.

DCouleur(String couleur)
Le constructeur de l’énumération prenant en paramètre la séquence d’échappement de la couleur choisie.

String toString()
Cette méthode permet de renvoyer la une chaine de caractères contenant la séquence d’échappement.

void setCouleur(String c)
Cet accesseur en écriture sert à attribuer un code d’échappement, stocké dans l’attribut ‘m_couleur’.

String getCouleur()
Cet accesseur en lecture sert à renvoyer le code d’échappement, stocké dans l’attribut ‘m_couleur’.

DEntree.java
Cette classe fait partie des classes utilitaires du projet, elle sert à recueillir ce que l’utilisateur saisi dans la console.

String lire()
Cette méthode sert à récupérer ce que saisi l’utilisateur et le renvoie sous forme de chaine de caractères. Elle n’est accessible que par les autres méthodes de la classe ‘DEntree’.

void pause()
Cette méthode sert à faire une pause dans le programme, en affichant un message à l’utilisateur lui demandant d’appuyer sur la touche entrée pour continuer.

String getString()
Cette méthode permet de demander à l’utilisateur d’entrer quelque chose et de renvoyer sa saisie sous forme d’une chaine de caractères.

int getInt()
Cette méthode permet de demander à l’utilisateur d’entrer quelque chose et de renvoyer sa saisie sous la forme d’un entier, redemande une entrée à chaque fois que l’utilisateur rentre une saisie incorrecte.

int getInt(int min)

Cette méthode permet de demander à l'utilisateur d'entrer quelque chose et de renvoyer sa saisie sous la forme d'un entier, redemande une entrée à chaque fois que l'utilisateur rentre une saisie incorrecte ou inférieur à la valeur 'min' passée en paramètre.

int getInt(int min, int max)

Cette méthode permet de demander à l'utilisateur d'entrer quelque chose et de renvoyer sa saisie sous la forme d'un entier, redemande une entrée à chaque fois que l'utilisateur rentre une saisie incorrecte, inférieur à la valeur 'min' ou supérieure à 'max' passées en paramètre.

boolean getConfirmation()

Cette méthode sert à obtenir une réponse négative ou positive de l'utilisateur en entrant 'o' pour une réponse positive et 'n' pour une réponse négative en étant insensible à la casse, une entrée vide ou incorrecte considèrera la réponse comme positive par défaut.

DOutils.java

Cette classe fait partie des classes utilitaires du projet, elle sert à recueillir diverses méthodes n'ayant pas directement un rapport avec le projet mais qui peuvent être utilisés de diverses façons.

int getNbChiffre(long i)

Cette méthode retourne le nombre de chiffre contenus dans un entier.

DSortie.java

Cette classe fait partie des classes utilitaires du projet, elle sert à gérer l'affichage dans la console.

void affiche(Object x)

Cette méthode est juste un enrobeur pour la méthode 'System.out.print()' mais est plus court à écrire.

void afficheLigne(Object x)

Comme la méthode précédente, cette méthode est juste un enrobeur pour la méthode 'System.out.println()' mais est plus court à écrire.

void afficheErreur(Object e)

Cette méthode permet d'afficher une erreur dans la console avec une mise en forme particulière.

String noir(Object x)

Cette méthode ajoute les séquences d'échappement nécessaire pour que l'argument passé en paramètre apparaisse en noir une fois dans la console.

String rouge(Object x)

Cette méthode ajoute les séquences d'échappement nécessaire pour que l'argument passé en paramètre apparaisse en rouge une fois dans la console.

String vert(Object x)

Cette méthode ajoute les séquences d'échappement nécessaire pour que l'argument passé en paramètre apparaisse en vert une fois dans la console.

String jaune(Object x)

Cette méthode ajoute les séquences d'échappement nécessaire pour que l'argument passé en paramètre apparaisse en jaune une fois dans la console.

String bleu(Object x)
Cette méthode ajoute les séquences d'échappement nécessaire pour que l'argument passé en paramètre apparaisse en bleu une fois dans la console.

String magenta(Object x)
Cette méthode ajoute les séquences d'échappement nécessaire pour que l'argument passé en paramètre apparaisse en magenta une fois dans la console.

String cyan(Object x)
Cette méthode ajoute les séquences d'échappement nécessaire pour que l'argument passé en paramètre apparaisse en cyan une fois dans la console.

String blanc(Object x)
Cette méthode ajoute les séquences d'échappement nécessaire pour que l'argument passé en paramètre apparaisse en blanc une fois dans la console.

Indice.java
Cette énumération sert à représenter les différents types d'indices que peut prendre une case.

Il n'existe que huit types d'indices pour l'instant : 'SOMME', 'DIFFERENCE', 'PRODUIT', 'QUOTIENT', 'RESTE', 'PUISSANCE', 'MINIMUM' et 'MAXIMUM'.

String m_indice
Cet attribut stockera le signe de l'indice de la case sous la forme d'une chaîne de caractères.

Indice(String indice)
Il s'agit du constructeur de l'énumération, il prend la représentation du type de l'indice sous la forme d'une chaine de caractères en paramètre.

String toString()
Cette méthode renvoie l'indice sous la forme d'une chaîne de caractères.

void setIndice(String i)
Cet accesseur en écriture sert à attribuer un signe à l'indice, et le stock dans l'attribut 'm_indice'.

String getIndice()
Cet accesseur en lecture renvoie la chaine de caractères stockée dans l'attribut 'm_indice'.

double getResultat(Case c1, Case c2)
Cette méthode permet d'obtenir la valeur de l'indice entre deux cases selon le type de l'indice de la première case.

Jeu.java
Cette classe représente une partie et gère son bon déroulement.

java.util.ArrayList<Joueur> m_joueurs
Cet attribut sert à gérer l'ensemble des joueurs peu importe le nombre de ces derniers.

Plateau m_plateau
Cet attribut permet de stocker le plateau du jeu actuel et son avancement.

Niveau m_niveau
Cet attribut représente le niveau de difficulté choisis pour la partie en cours.

Jeu(java.util.ArrayList<Joueur> joueurs, Niveau niveau)

Ce constructeur permet d'initialiser une nouvelle partie en spécifiant l'ensemble de joueurs participants et le niveau de difficulté désiré.

void joue()

Cette méthode sert à simuler le déroulement de la partie.

void afficheCaracteristiquesPartie()

Cette méthode sert à afficher les caractéristiques de la partie en cours.

void afficheCaracteristiquesJoueurs()

Cette méthode sert à afficher les caractéristiques des différents joueurs.

String toString()

Cette méthode renvoie les caractéristiques de la partie sous la forme d'une chaine de caractères.

void setJoueurs(java.util.ArrayList<Joueur> j)

Cet accesseur en écriture permet d'attribuer un ensemble de joueurs à la partie en cours en le stockant dans l'attribut 'm_joueurs'.

void setPlateau(Plateau p)

Cet accesseur en écriture permet d'attribuer un plateau à la partie en cours en le stockant dans l'attribut 'm_plateau'.

void setNiveau(Niveau n)

Cet accesseur en écriture permet d'attribuer un niveau à la partie en cours en le stockant dans l'attribut 'm_niveau'.

java.util.ArrayList<Joueur> getJoueurs()

Cet accesseur en lecture permet d'obtenir l'ensemble des joueurs de la partie en cours stocké dans l'attribut 'm_joueurs'.

int getNbJoueurs()

Cette méthode permet d'obtenir le nombre de joueurs de la partie en cours.

Joueur getJoueur(int i)

Cette méthode permet d'obtenir la référence d'un joueur en spécifiant son rang dans la partie en cours.

Plateau getPlateau()

Cet accesseur en lecture permet d'obtenir la référence du plateau de la partie en cours.

Niveau getNiveau()

Cet accesseur en lecture permet d'obtenir le niveau de la partie en cours.

Joueur.java

Cette classe sert à représenter un joueur.

String m_nom

Cet attribut sert à contenir le nom du joueur.

long m_score

Cet attribut sert à contenir le score actuel du joueur.

Joueur(String nom)
Ce constructeur permet de créer un nouveau joueur en spécifiant son nom.

void joue(Plateau p, Niveau n)
Cette méthode sert à gérer le déroulement du tour d'un joueur et permet d'alléger la classe Jeu.

void reussi(long s)
Cette méthode sert à ajouter le nombre passé en paramètre au score du joueur lorsque ce dernier aura correctement deviné le contenu de deux cases.

String toString()
Cette méthode permet de retourner le nom et le score du joueur sous forme d'une chaine de caractères.

void setNom(String n)
Cet accesseur en écriture permet de changer le nom du joueur.

void setScore(long s)
Cet accesseur en écriture permet de changer la valeur du score du joueur par la valeur passée en paramètre en la stockant dans l'attribut 'm_score'.

String getNom()
Cet accesseur en lecture permet d'obtenir le nom du joueur en retournant la référence à la chaine de caractères contenue dans 'm_nom'.

long getScore()
Cet accesseur en lecture permet d'obtenir le score du joueur en renvoyant la valeur stockée dans l'attribut 'm_score'.

Main.java
Cette classe sert uniquement à contenir le point d'entrée du programme afin de le séparer des autres classes.

void main(String[] args)
Il s'agit du point d'entrée du programme. Cette méthode demande toutes les informations nécessaires à la création d'une nouvelle partie, lance cette dernière et recommence jusqu'à ce que l'utilisateur refuse de faire une nouvelle partie.

Niveau.java
Cette énumération sert à représenter un niveau de difficulté de jeu. Les différents niveaux, par ordre croissant de difficulté, sont les suivants : 'DEBUTANT', 'FACILE', 'MOYEN', 'DIFFICILE', 'EXTREME' et 'IMPOSSIBLE'.

String m_niveau
Cet attribut sert à contenir le nom du niveau de difficulté.

int m_largeur
Cet attribut sert à contenir le nombre de case de la largeur du plateau.

int m_hauteur
Cet attribut sert à contenir le nombre de case de la hauteur du plateau.

int m_valeurMax
Cet attribut sert à contenir la valeur maximale que peut prendre une case du plateau.

boolean m_indicesVisibles
Cet attribut sert à représenter si les indices des cases doivent être visibles ou non depuis le début de la partie ou non.

Niveau(String niveau, int largeur, int hauteur, int valeurMax, boolean indicesVisibles)
Il s'agit du constructeur de l'énumération demandant en paramètre toutes les valeurs des différents attributs.

String toString()
Cette méthode permet de retourner le nom du niveau de difficulté sous forme d'une chaine de caractères.

void setNiveau(String n)
Cet accesseur en écriture permet de changer le nom du niveau de difficulté avec la chaine de caractères passée en paramètre.

void setLargeur(int l)
Cet accesseur en écriture permet d'attribuer la valeur passée en paramètre à l'attribut 'm_largeur'.

void setHauteur(int h)
Cet accesseur en écriture permet d'attribuer la valeur passée en paramètre à l'attribut 'm_hauteur'.

void setValeurMax(int vm)
Cet accesseur en écriture permet d'attribuer la valeur passée en paramètre à l'attribut 'm_valeurMax'.

void setIndicesVisibles(boolean iv)
Cet accesseur en écriture permet d'attribuer la valeur du booléen passé en paramètre à l'attribut 'm_indicesVisibles'.

String getNiveau()
Cet accesseur en lecture renvoie la référence à la chaine de caractères correspondant au nom du niveau de difficulté contenue dans l'attribut 'm_niveau'.

int getLargeur()
Cet accesseur en lecture renvoie la valeur contenue dans l'attribut 'm_largeur'.

int getHauteur()
Cet accesseur en lecture renvoie la valeur contenue dans l'attribut 'm_hauteur'.

int getValeurMax()
Cet accesseur en lecture renvoie la valeur contenue dans l'attribut 'm_valeurMax'.

boolean isIndicesVisibles()
Cet accesseur en lecture renvoie la valeur du booléen contenu dans l'attribut 'm_indicesVisibles'.

Plateau.java
Cette classe permet de représenter un plateau de jeu et de le manipuler.

Case[][] m_cases
Cet attribut servira à contenir l'ensemble des cases du plateau dans un tableau bidimensionnel.

int m_largeur
Cet attribut servira à contenir le nombre de cases que contient le plateau en largeur.

int m_hauteur
Cet attribut servira à contenir le nombre de cases que contient le plateau en hauteur.

Plateau(int largeur, int hauteur, int valeurMax, boolean indicesVisibles)
Ce constructeur permet de créer un nouveau plateau ayant la largeur et la hauteur spécifiées en paramètre. A noter que les deux autres paramètres servent à préciser l'état initial des cases du plateau, respectivement la valeur maximale que peut prendre une case et un booléen valant 'true' si les indices des cases doivent être visibles.

boolean toutesLesCasesSontDecouvertes()
Cette méthode renvoie la valeur 'false' tant que toutes les valeurs des cases n'ont pas toutes été trouvées.

void affiche()
Cette méthode permet d'afficher le plateau.

String toString()
Cette méthode permet de renvoyer le contenu du plateau sous la forme d'une chaine de caractères.

void setCases(Case[][] c)
Cet accesseur en écriture sert à attribuer un ensemble de cases à l'attribut 'm_cases'.

void setCase(int x, int y, Case c)
Cette méthode sert à attribuer une case à l'attribut 'm_cases' en précisant ses coordonnées.

void setLargeur(int l)
Cet accesseur en écriture permet de modifier la valeur de l'attribut 'm_largeur'.

void setHauteur(int h)
Cet accesseur en écriture de modifier la valeur de l'attribut 'm_hauteur'.

Case[][] getCases()
Cet accesseur en lecture permet d'obtenir la référence à l'ensemble des cases du plateau.

Case getCase(int x, int y)
Cet accesseur en lecture permet d'obtenir la référence à une des cases du plateau en précisant ses coordonnées.

int getLargeur()
Cet accesseur en lecture permet d'obtenir la valeur de la largeur du plateau.

int getHauteur()
Cet accesseur en lecture permet d'obtenir la valeur de la hauteur du plateau.