Code mis à disposition

Thomas Raimbault, janvier 2023

Un certain nombre de **fonctions sont mises à disposition** pour la *Coding Week*. Ces fonctions sont présentées sur des exemples d'utilisation.

Remarque importante : lors de l'utilisation des fonctions fournies dans Replit, elles sont soulignées en rouge dans le code comme s'il s'agissait d'une erreur... mais ne pas en tenir compte. Tenir uniquement compte des erreurs affichées en Console après avoir cliqué sur "Run" (par exemple une mauvaise écriture du nom d'une fonction).
Pour ce qui est de votre code / vos fonctions à vous, là c'est une autre histoire...

Table des matières

Console

> pour saisir au clavier

Ci-dessous des exemples d'utilisation des fonctions fournies :

```
int entier   = Console.saisirEntier();  // saisir un entier
double reel  = Console.saisirReel();    // saisir un réel
boolean bool = Console.saisirBooleen(); // saisi un booléen (true, false)
String texte = Console.saisirChaine();  // saisir une chaîne de caractères
```

> pour supprimer le contenu de la Console

```
Console.effacerConsole();
```

> pour marquer une pause (manuelle)

```
Console.appuyerSurEntrer();
```

Outils

> pour charger une carte

```
int[][] carte = Outils.chargerCarte("uneCarte.txt");
```

- en paramètre le nom du fichier de la carte à charger (type `String`)
- retourne une matrice d'entiers (type `int[][]`)

> pour enregistrer une carte

```
Outils.enregistrerCarte(carte, "uneCarte.txt");
```

- en paramètres la matrice d'entiers à enregistrer (type `int[][]`)
- et le nom du fichier de la carte où enregistrer (type `String`) si le fichier existe il est écrasé.

> pour marquer une pause (en durée)

```
Outils.attendre(1000);
```

- en paramètre la durée de pause en millisecondes (type `int`)
- sur cet exemple, pause de 1 seconde (1000 ms)

> pour générer des nombres aléatoires

```
double reel = Outils.reelAleatoire();
```

- retourne un réel aléatoire dans l'intervalle [0.0,1.0[

```
int entier  = Outils.entierAleatoire(10, 20);
```

- en paramètres les bornes min et max incluses (type `int`)
- retourne un entier aléatoire, dans les bornes (sur cet exemple, entre 10 et 20)

Fenêtre graphique

> pour associer une fenêtre à la matrice d'entiers

```
FenetreGraphique.initialiserFenetre(carte);
```

- en paramètre la matrice d'entiers manipulée durant le programme (type `int[ ][ ]`)

> pour rafraîchir graphiquement un élément

```
FenetreGraphique.rafraichirElement(indexLigne, indexColonne);
```

- à utiliser **systématiquement** après avoir modifié un élément dans la matrice d'entiers (pas de rafraîchissement automatique de la fenêtre lors de mises à jour de la matrice d'entiers)
- en paramètres les index de ligne puis de colonne de l'élément concerné (type `int`)

> pour modifier le message en bas de la fenêtre

```
FenetreGraphique.modifierMessageBas("le message");
```

- en paramètre le message à afficher (type String)

> pour modifier le message à droite de la fenêtre

```
FenetreGraphique.modifierMessageDroite("1ere ligne\n2e ligne\n3e ligne");
```

- en paramètre le message à afficher (type String)
- possibilité d'avoir un message sur plusieurs lignes (retour à la ligne via le caractère `"\n"`)

> pour afficher un message "pop-up"

```
FenetreGraphique.messagePopUp("le message");
```

- en paramètre le message à afficher (type String)

Quelques rappels Java

> pour afficher en Console

```
System.out.print(...);
System.out.println(...);  // avec retour à la ligne en fin d'affichage
```

> entiers, réels, division et modulo

Une division entre deux entiers est une division entière (même si le résultat est affecté à un réel). Par exemple :

```
int e    = 7 / 2;       // e vaudra 3
double r = 7 / 2;       // r vaudra 3.0
```

Une division dont au moins un des opérandes est un réel est une division réelle. Par exemple :

```
double r = 7 / 2.0;           // r vaudra 3.5
int e    = 7 / 2.0;           // erreur  (car risque de perte
                              // d'informations, ici la partie décimale)
```

Le reste d'une division est obtenu par l'opérateur modulo (symbole **%** en Java). Par exemple :

```
int reste = 7 % 2;            // reste vaut 1 (puisque 7 = 3*2 +1)
```

> allocation dynamique d'un tableau

```
int[] tab = new int int[10];        // création en mémoire d'un tableau 1D
                                    // de 10 éléments

int[][] tab2D =new int int[2][3];   // création en mémoire d'un tableau 2D
                                    // de 2 lignes et 3 colonnes
```

> taille d'un tableau (nombre d'éléments)

```
tab.length        // nombre d'éléments du tableau 1D (si non null)

tab2D.length      // nombre de lignes du tableau 2D (si non null)
tab2D[0].length   // nombre de colonnes de la 1ère ligne (d'index 0 ici)
```

> accès à un élément d'un tableau

```
tab[0]            // accès au 1er élément
tab[tab.length-1] // accès au dernier élément

tab[0][1]         // accès à l'élément en 1ère ligne (index 0 de ligne)
                  // et 2nde colonne (index 1 de colonne)
```