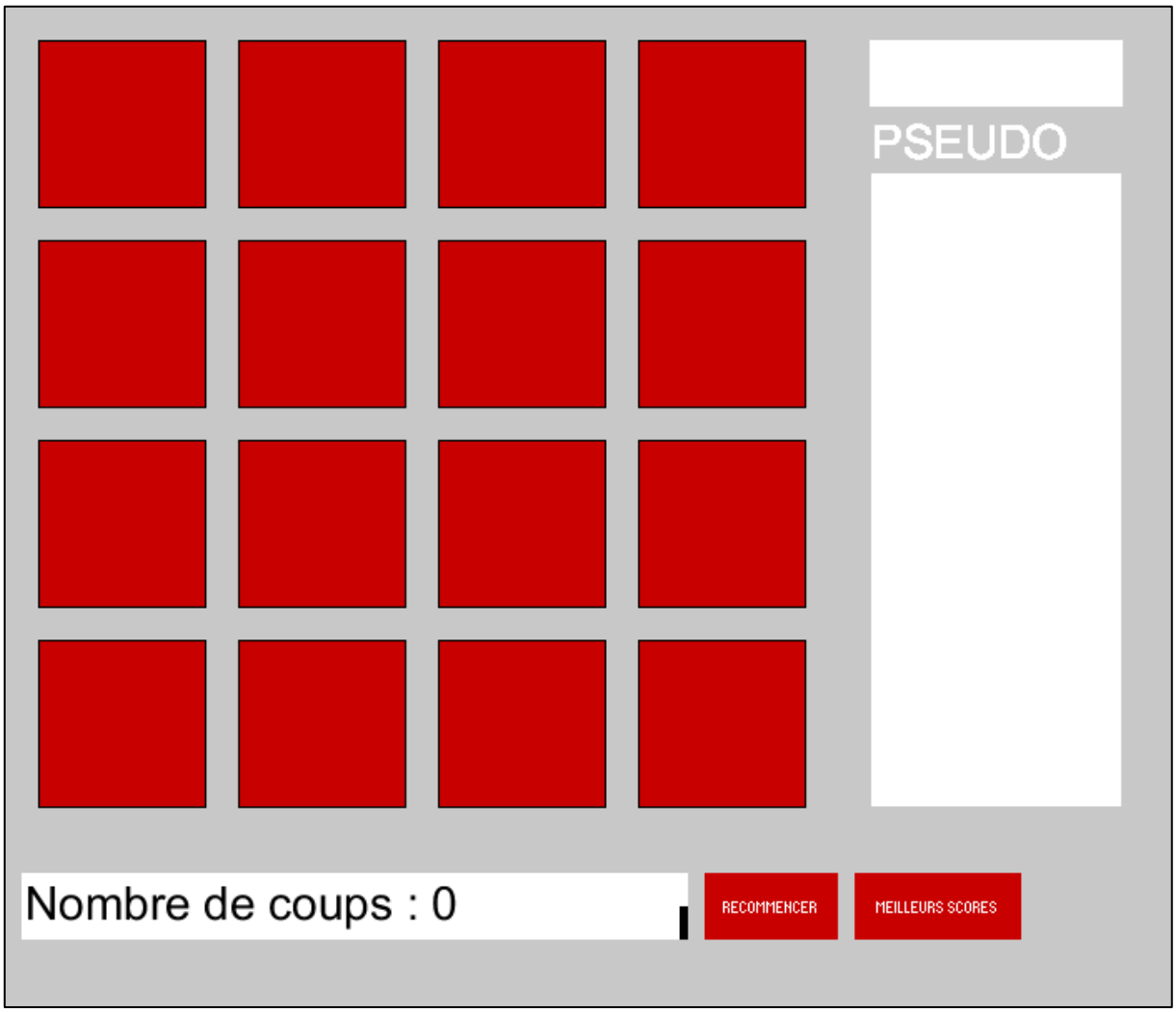
Morgane

KELLER

TS3

Dossier ISN : Jeu de memory



- Qu'est-ce qu'un jeu des memory ?

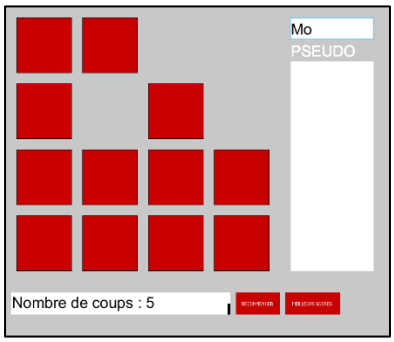
C'est un jeu de cartes constitué d'un certain nombre de paires (dans notre cas, 8 paires soit 16 cartes) disposé de façon à ce que le joueur voit l'arrière des cartes, identique pour chaque carte. Nous avons disposé les cartes virtuelles 4 x 4.

Le joueur entre son pseudo dans la case prévue à cet effet en haut à droite de l'écran.

Le joueur retourne 2 cartes. Si ces 2 cartes sont les mêmes, le joueur les retire de la disposition. Si elles sont différentes, le joueur retourne les cartes. Dans notre jeu, les cartes se retirent ou se retournent dos au joueur automatiquement. Le joueur peut essayer autant de fois qu'il le veut jusqu'à ce qu'il n'y ait plus aucune carte, le but étant de réussir à trouver toutes les paires en ayant retourné les cartes avec le moins de coups possible. Il peut visualiser le nombre de coup effectué dans la zone d'affichage en bas à gauche.

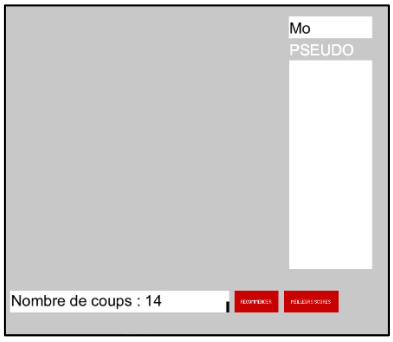
Une fois la partie finie, le joueur peut cliquer sur le bouton « recommencer » pour recommencer une nouvelle partie.

Le joueur peut voir quels sont les joueurs qui ont eu le meilleur score (qui ont retourné avec le moins de coup toutes les cartes) en cliquant sur le bouton « meilleurs scores ».



// Situation de jeu après 5 coups (10 clics)

- 2 paires ont été trouvées



// Situation où toutes les paires ont été trouvées après 14 coups (28 clics)

// Situation après avoir cliqué sur le bouton « recommencer »

- Toutes les cartes ont été reposées à des places différentes

- Le nombre de coups est remis à 0

- Le score a été enregistré dans scores.txt

- Avec quels moyens avons-nous créé ce jeu ?

Pour réaliser ce jeu, nous avons utilisé le logiciel Processing (j'ai personnellement préféré utiliser la version 2.2.1), et nous avons utilisé le Java. Pour les cartes, nous avons utilisé une image mis en ressource par notre professeur :

Chaque véhicule est présent sur 2 cartes puisque sous les cartes se trouvent des paires.

Nous avons également utilisé un fichier texte scores.txt avec des scores préenregistrés également mis en disposition par notre professeur :

Extrait de scores.txt :

«

```
Ted                                 Marc
16                                  28
Max                                 Sam
20                                  25
Marc                                Téo
18                                  30
Luc                                 MC
19                                  13
Tom                                 Mich
24                                  13
Bob                                 Nico
23                                  16
Jo                                  Sam
27                                  14
```

»

Après l'exemple de la page précédente, les lignes suivantes ont été ajoutées au fichier :

«

```
Mo
14
```

»

À quoi servent les grandes fonctions du programme ?

À quoi sert la fonction setup() pour notre jeu :

J'y ai défini :

- Une fenêtre de taille 700 x 600

- Un fond de couleur grise

- Des carrés (les cartes) de couleur rouge, de dimensions 100 x 100, derrière lesquels se trouvent une des 8 images de véhicule de façon à ce qu'ils apparaissent 2 fois chacun

- La zone d'affichage indiquant le nombre de coups de dimension 400 x 40, la zone d'affichage indiquant les meilleurs scores de dimension 150 x 380

- La zone de saisie pour entrer son pseudo de dimension 150 x 40

- Le bouton « recommencer » lié à la fonction void recommencer() de dimension 80 x 40, le bouton « Meilleurs scores » lié à la fonction void meilleursScores() de dimension 100 x 40

À quoi sert la fonction draw() pour notre jeu :

- À l'aide de fonction if j'ai fait en sorte que si le joueur à cliquer sur 2 cartes différentes et que ce sont les même, elles disparaissent ou qu'elles se retournent si elles sont différentes

- J'ai ajouté un délai de 0,5s entre le moment où le joueur à cliquer sur les cartes et le moment où s'effectue la fonction afin qu'il puisse visualiser les cartes

À quoi sert la fonction mousséPressed() pour notre jeu :

Lorsque l'on clique sur une des 16 cartes :

- On incrément le nombre de clics ainsi que le nombre de coups

- On fait apparaitre l'image caché derrière la carte

- On fait modifie le nombre de clics dans la zoneTexte

À quoi sert la fonction recommencer() (liée au bouton « recommencer ») pour notre jeu :

Si on a trouvé les 8 paires :

- On replace des cartes avec les images à des places différentes

- On remet à 0 le nombre de cartes trouvées, le nombre de coups et le nombre de clics

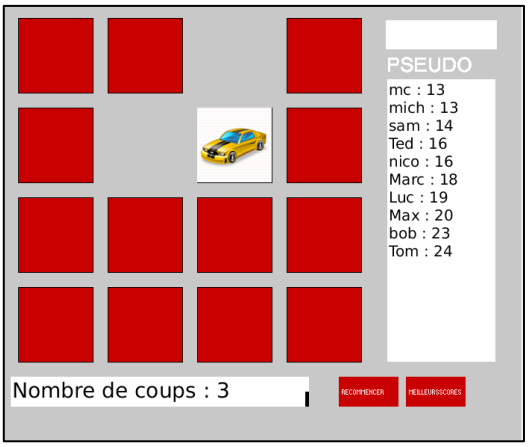
- On enregistre le pseudo et le score dans le fichier scores.txt

<u>À quoi sert la fonction meilleursScores() (liée au bouton « Meilleurs Scores ») :</u>

- À afficher dans la zoneScores les 10 meilleurs scores accompagnés des pseudos correspondants

<u>À quoi sert la fonction tri() :</u>

- Tri bulles permettant de trier les scores par ordre croissant



// Situation après avoir cliqué sur le bouton « Meilleurs scores »

Annexe

```
//------------------------------ DECLARATIONS -------------------------------
int l = 100;  // Largeur
int h = 100;  // Hauteur
int x = 20;  // Abscisse du coin supérieur gauche
int y = 20;  // Ordonnée du coin supérieur gauche
int clic = 0;  // Variable pour compter le nombre de clics
PImage img1;
PImage img2;
int abs = 0;
int tab1[][] = new int [4][4];
int tab2[] = {
  0, 0, 0, 0, 0, 0, 0, 0
};
int nbClic = 0;
int i1 = -1;
int i2 = -1;
int j1 = -1;
int j2 = -1;
int n1 = -1;
int n2 = -1;
import controlP5.*; // On importe la bibliothèque controlP5
ControlP5 cp5; // On déclare l'objet cp5
Textarea zoneTexte;
int nbCoups = 0;
Button bouton1;
Textfield zonePseudo;
int nbTrouve = 0;
String lignes[];// On déclare le tableau qui contiendra les lignes du
fichier
PrintWriter fichier;// On déclare l'objet fichier pour pouvoir écrire
dedans
int nbLignes;// Variable entière pour le nombre de lignes du fichier
scores.txt
Button bouton2;
Textarea zoneScores;
//--------------------------- FONCTION SETUP() ----------------------------
void setup()
{
  background(200);  // Couleur de fond de la fenêtre principale en RVB
  size(700, 600);  // Taille de la fenêtre principale
  rect(x, y, l, h);  // On trace le rectangle
  img1 = loadImage("Image1.png"); // On charge l'image en mémoire :
Image1.png dans la variable img1
  fill(200, 0, 0); //Couleur d'intérieur
  for ( int i=0; i < 4; i++) // Boucle for pour afficher 4 lignes de carré
  {
    for ( int j=0; j < 4; j++)// Boucle for pour afficher 4 colonnes de
carré
    {
      rect(20+120*i, 20+120*j, 100, 100); // On affiche les carrés de la
matrice
      int n = int(random(8));
      while (tab2[n] == 2)
      {
        n = int(random(8));
      }
      tab1[i][j]=n;
```

```
      tab2[n]++;
    }
  }
  cp5 = new ControlP5(this);
  zoneTexte = cp5.addTextarea("zone")
    .setPosition(10, 520) // Abscisse et ordonnées du coin supérieur gauche
      .setSize(400, 40) // Largeur et hauteur de la zone d'affichage
        .setFont(createFont("arial", 28)) // type et taille de la police
          .setLineHeight(30) // hauteur de chaque ligne dans la zone
d'affichage
            .setColor(color(0)) // Couleur de la police
              .setColorBackground(color(255)) // Couleur du fond
                .setColorForeground(color(255)) // Couleur de premier plan
                  .setText("Nombre de coups : 0")
                    ;
  nbClic=0;
  cp5 = new ControlP5(this);
  bouton1=cp5.addButton("Recommencer")
    .setPosition(420, 520)// Abscisse et ordonnées du coin supérieur gauche
      .setSize(80, 40)// Largeur et hauteur de la zone d'affichage
        .setColorBackground(color(200, 0, 0))// Couleur du fond
          ;
  zonePseudo = cp5.addTextfield("Pseudo")
    .setPosition(520, 20) // Abscisse et ordonnées du coin supérieur gauche
      .setSize(150, 40) // Largeur et hauteur de la zone d'affichage
        .setFont(createFont("arial", 28)) // type et taille de la police
          .setColor(color(0)) // Couleur de la police
            .setColorBackground(color(255)) // Couleur du fond
              .setColorForeground(color(255)) // Couleur de premier plan
                ;
  bouton2=cp5.addButton("Meilleurs scores")
    .setPosition(510, 520)// Abscisse et ordonnées du coin supérieur gauche
      .setSize(100, 40)// Largeur et hauteur de la zone d'affichage
        .setColorBackground(color(200, 0, 0))// Couleur du fond
          ;
  zoneScores = cp5.addTextarea("zone")
    .setPosition(520, 100) // Abscisse et ordonnées du coin supérieur
gauche
      .setSize(150, 380) // Largeur et hauteur de la zone d'affichage
        .setFont(createFont("arial", 28)) // type et taille de la police
          .setLineHeight(30) // hauteur de chaque ligne dans la zone
d'affichage
            .setColor(color(0)) // Couleur de la police
              .setColorBackground(color(255)) // Couleur du fond
                .setColorForeground(color(255)) // Couleur de premier plan
                  ;
}
//--------------------------- FONCTION DRAW() -----------------------------
void draw()  // Cette fonction même vide doit être présente
{
  if (nbClic==2)
  {
    if (n1!=n2)
    {
      fill(200, 0, 0); //Couleur d'intérieur
      rect(20+120*i1, 20+120*j1, 100, 100);
      rect(20+120*i2, 20+120*j2, 100, 100);
    } else
    {
      fill(200); //Couleur de fond pour "enlever" la carte
      stroke (200);
```

```
      rect(20+120*i1, 20+120*j1, 100, 100);
      rect(20+120*i2, 20+120*j2, 100, 100);
      tab1[i1][j1]=-1;
      tab1[i2][j2]=-1;
      nbTrouve++;
    }
    delay(500); // On fait une pause d'une demi-seconde soit 500
millisecondes.
    nbClic=0;
  }
}
//-------------------------- FONCTION MOUSEPRESSED() ----------------------
void mousePressed()  // Fonction qui s exécute quand on clique avec la
souris
{
  for ( int i=0; i < 4; i++ )
  {
    for ( int j=0; j < 4; j++ )
    {
      if ( mouseX>20+i*120 && mouseX<120+i*120 && mouseY>20+j*120 &&
mouseY<120+j*120 && nbClic < 2 && tab1[i][j]!=-1)
      {
        nbClic++;
        abs = tab1[i][j];
        img2 = img1.get(abs*100, 0, 100, 100);
        image(img2, 20+i*120, 20+j*120);
        if (nbClic == 1)
        {
          i1 = i;
          j1 = j;
          n1 = tab1[i][j];
        } else
        {
          if (i==i1 && j==j1)
          {
            nbClic--;
          } else
          {
            i2=-1;
            j2=-1;
            n2=-1;
          }
          i2=i;
          j2 =j;
          n2=tab1[i][j];
          nbCoups++;
          zoneTexte.setText("Nombre de coups : " + nbCoups);
        }
      }
    }
  }
}
//-------------------------- FONCTION RECOMMENCER() ----------------------
void recommencer()
{
  String pseudo = zonePseudo.getText();
  lignes = loadStrings("scores.txt");// On charge les lignes du fichier
scores.txt dans le tableau
  nbLignes = lignes.length; // On récupère le nombre de lignes
  if (nbTrouve==8 &&!pseudo.equals(""))
  {
```

```
    fichier = createWriter("scores.txt");
    for (int i=0; i<nbLignes; i++)
    {
      fichier.println(lignes[i]);// On écrit chaque ligne dans le fichier
    }
    fichier.println(pseudo);// On rajoute une ligne en fin de fichier avec
un pseudo
    fichier.println(nbCoups);// On rajoute une 2ème ligne en fin de fichier
avec un score
    fichier.flush();// On actualise le fichier
    fichier.close();// On ferme le fichier
  }
  nbTrouve=0;
  nbCoups=0;
  nbClic=0;
  zoneTexte.setText("Nombre de coups : 0");
  for ( int i=0; i < 8; i++)
  {
    tab2[i]=0;
  }
  img1 = loadImage("Image1.png"); // On charge l'image en mémoire :
Image1.png dans la variable img1

  for ( int i=0; i < 4; i++)
  {
    for ( int j=0; j < 4; j++)
    {
      fill(200, 0, 0);//Couleur d'intérieur
      rect(20+120*i, 20+120*j, 100, 100);
      int n = int(random(8));
      while (tab2[n] == 2)
      {
        n = int(random(8));
      }
      tab1[i][j]=n;
      tab2[n]++;
    }
  }
}
//------------------------Fonction MEILLEURSSSCORES()----------------------
void meilleursScores()
{
  int tabScores;
  String tabPseudo;
  for ( int i = 0; i<20; i+=2 )
  {
    tabScores[i] = lignes [i/2] ;  // Si i est pair
    tabPseudo[i] = lignes[i/2+1] ;  // Si i est impair
  }
}
//--------------------------Fonction TRI (tri bulles)----------------------
void tri() // Procédure de tri bulles
{
  boolean echange=true; // Variable pour savoir si il y a eu au moins un
échange
  int temp; // Variable intermédiaire
  while (echange) // Tant qu'un échange est effectué
  {
    echange=false; // Pour l'instant aucun échange n'a été réalisé
    for (int i=0; i<nbLignes-2; i+=2)
    {
```

```
      if ( lignes[i]>lignes[i+2] ) // Si l'ordre croissant n'est pas
respecté
      {
        temp = lignes[i]; // On échange les deux valeurs du tableau
        lignes[i] = lignes[i+2];
        lignes[i+2] = temp;
        echange = true; // Un échange a été effectué
      } // Fin du if
    } // Fin du for
  } // Fin du While
  String reponse=""; // Chaîne initialisée à vide pour afficher le tableau
trié
  // dans la console
  for ( int i=0; i<nbLignes; i++ ) // Boucle pour afficher les valeurs du
tableau
  {
    reponse = reponse + str(lignes[i]) + "\n" ; // On ajoute les valeurs
par concaténation
  }
  println(reponse); // On affiche le tableau trié dans la console
}

//---------------------------------- FIN ---------------------------------
```