Projet LU2IN002 - 2020-2021

Numéro du groupe de TD/TME :

Nom : Ramdani	*Nom :* Gazquez	*Nom :*
Prénom : Cyrena	*Prénom :* Miguel	*Prénom :*
N° étudiant : 3805942	*N° étudiant :* 28600492	*N° étudiant :*

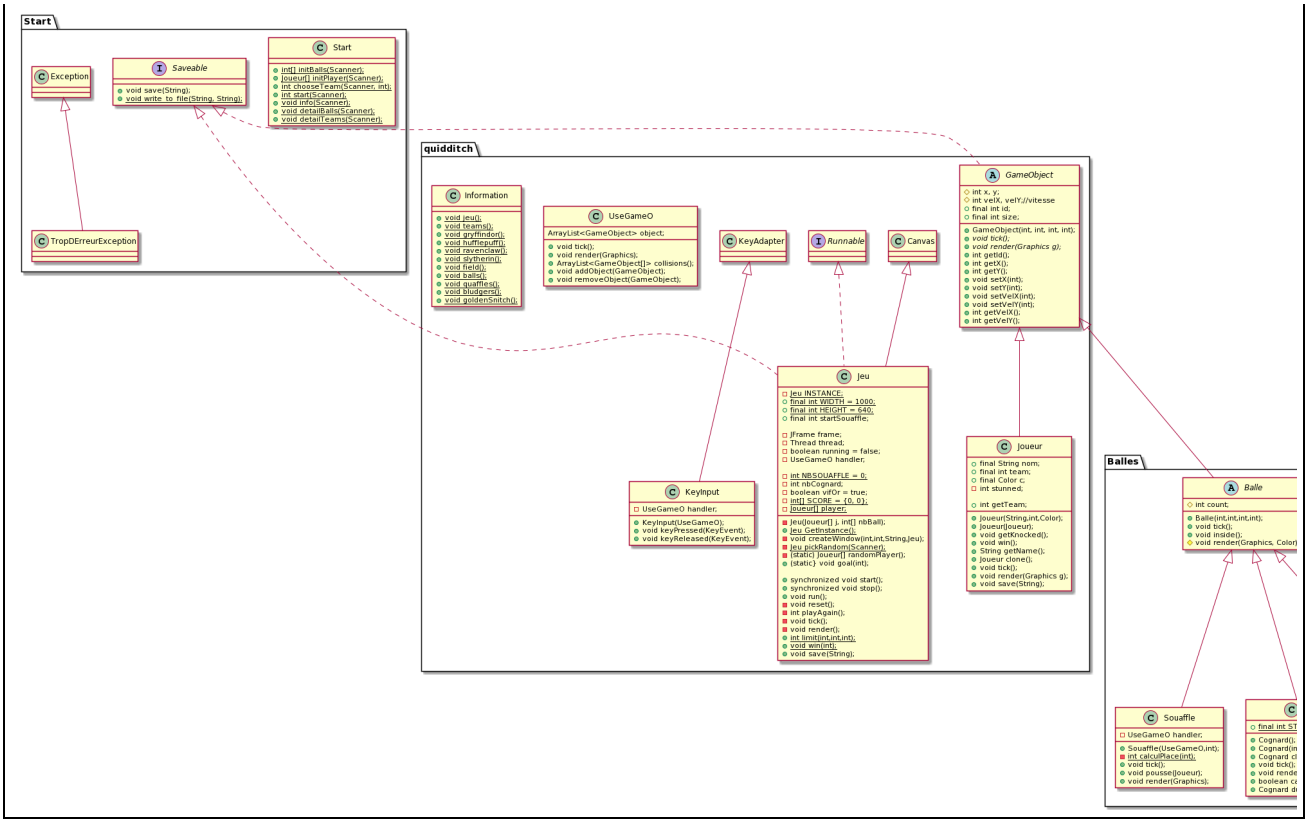
Thème choisi (en 2 lignes max.)

Le thème est inspiré du quidditch, un sport de balle fictif issu de la saga Harry Potter créé par J.K Rowling. Il s'agit d'un jeu multijoueur qui vous invite à chevaucher vos balais volant et défier vos amis.

Description des classes et de leur rôle dans le programme (2 lignes max par classe)

- **Information** : classe static printant sur la ligne de commande les informations sur le jeu, équipe, ou les balles.
- **Start** : Donne accès au menu d'accueil sur ligne de commande et démarre le Jeu en
- **TropDErreurException** : exception lance si l'utilisateur faire trop d'erreur en rentrant ses configurations avant de commencer le jeu.
- **Jeu** : Commence le fil de jeu, initialise la fenêtre de jeu, initialise les 2 joueurs, le nombre de balles)par sélection ou de façon aléatoire), ajoute les objets de jeu.
- **KeyInput** : permet de gérer l'utilisation du clavier et permet de déplacer les joueurs respectifs.
- **GameObject** : classe abstraite donnant les prérequis de chaque objet apparaissant sur le terrain (position, affichage, taille, identifiant, vitesse).
- **UseGameO** : Permet de gérer tous les GameObject en les mettant dans une ArrayList. Détecte les collisions entre objets et permet de les ajouter/supprimer/update.
- **Joueur** : chaque joueur a un nom, une équipe et une couleur. Il apparait sous la forme d'un carré. S'il entre en contact avec un Cognard il sera immobilisé pendant un délai.
- **Balle** : chaque balle rebondit sur les rebords. Elles apparaissent tous avec une forme circulaire mais on des tailles, couleurs, comportements différents.
- **Souaffle** : initialise au centre du terrain (magenta), est mobile seulement après un contact avec un joueur et perd de la vitesse progressivement. En contact avec l'un des goals il disparait et faire gagner un point à l'autre équipe.
- **Cognard** : avec cette option les Cognards apparaissent (en noir) aléatoirement et se déplacent toujours la même vitesse (avec parfois des changements de trajectoire)
- **VifOr** : avec cette option le Vif d'Or apparait (en jaune) aléatoirement à intervalle irrégulier en se téléportant. L'attraper fait gagner le match.
- **Saveable :** Un fichier est saveable si on peut en enregistrer les donner dans un fichier. On enregistre différentes choses en fonction de la classe dont il s'agit

Schéma UML des classes vision fournisseur (dessin "à la main" scanné ou photo acceptés)



Checklist des contraintes prises en compte:	*Nom(s) des classe(s) correspondante(s)*
Classe contenant un tableau ou une ArrayList	- UseGameO : ArrayList<GameObject> object - Jeu
Classe avec membres et méthodes statiques	- Information
Classe abstraite et méthode abstraite	- GameObject : tick(),render(Graphics g) - Balle
Interface	- Saveable
Classe avec un constructeur par copie ou clone()	- Cognard - Joueur
Définition de classe étendant Exception	- TropDErreurException
Gestion des exceptions	- Start - Jeu
Utilisation du pattern singleton	- Jeu

Présentation de votre projet (max. 2 pages) : texte libre expliquant en quoi consiste votre projet.

Notre projet est inspiré du **Quidditch**, un sport de balle fictif issu de la saga Harry Potter créé par J.K Rowling.

Voici les règles et particularité de ce sport :

Jeu

À chaque extrémité du terrain de Quidditch se trouvent trois anneaux à travers lesquels le Souafle doit passer pour marquer des points. Au centre du terrain se trouve un cercle, c'est de là que les balles sont lancées quand le match débute. Une fois les balles lancées, les joueurs se rassemblent au sol et s'envolent au coup de sifflet de l'arbitre.

Le Terrain

Le Quidditch se joue au-dessus du sol sur des balais. Il y a une zone de but à chaque extrémité du terrain. L'Aire de jeu est appelée le terrain de Quidditch.

Balles

On utilise trois types de balles différents :

Le Souafle sert à marquer des points.

Le Cognard est sans doute la balle la plus dangereuse :
il vole dans les airs et des joueurs appelés Batteurs sont chargés de le frapper. De graves blessures peuvent être causées par un Cognard et certains joueurs tombent de leur balai, déséquilibrés.

La troisième et la plus importante des balles est le **Vif d'or** :
C'est une petite balle pourvue d'ailes. Si l'attrapeur réussit à se saisir du Vif d'or, son équipe gagne 150 points d'un coup et fait remporter la plupart du temps la victoire à son équipe. Ces trois balles sont ensorcelés.

Pour ce qui est de notre projet :
Nous sommes parvenues à faire la version 2 proposé au début :

- GUI
- 1 joueur dans chaque équipe.
- 3-5 souaffles sont disponibles. le Cognard est disponible. Le Cognard se déplace de aléatoirement et s'il rentre en interaction avec un joueur, le joueur est ne peut pas bouger pendant le cooldown
- le Vif d'or est disponible et apparait de aléatoirement. Si un joueur, l'attrape il gagne la partie

Objectif est de déplacer le plus souaffle jusqu'au but adverse, celui qui en marque le plus gagne. Interface graphique représentant le terrain, avec un point de couleur représentant chaque joueur et la balle

Pour leur représentation :

- **joueur:** sont des carrés de couleur rouge, jaune, bleu ou vert en fonction de la maison choisi et apparaissent dans leurs camps respectifs il faut marquer dans le but adverse
- **balles:** sont des cerles les balles triées dans l'ordre de taille décroissante
  - **souaffle :** couleur magenta
  - **Cognard** : couleur noire
  - **Vif d'or** : couleur jaune

1. **Menu de jeu (en ligne de commande)**

Donne l'accès :
- Jeu
- Informations générales sur le jeu
- Terminer le programme

2. **Le jeu**

On peut choisir les paramètres :
- nom des joueurs et leur maison (couleur)
- le nombre de Souaffle mis en jeu (3 ou 5)
- le nombre de Cognard (entre 0 et 2)
- si on veut jouer avec le vif d'or ou non

Si le joueur se trompe de sélections trop de fois le choix se fait de façon aléatoire.

Ou on sélectionne directement des paramètres aléatoires.

3. **Fin de jeu**

La partie se termine s'il n'y a plus de Souaffles ou si un joueur touche le vif d'or
Le joueur peut alors décider de rejouer ou de terminer le programme

4. **Choses à implémenter**

- sauvegardes des scores/paramètres

Pas encore implémenté mais il suffit d'ajouter une option de sauvegarde au début du jeu.
On peut alors faire appel au fonction save de jeu qui sauvegarde les scores , ou save de GameObject qui sauvegarde les
- Menu en interface graphique
- Music player quand le jeu commence
- Amélioration d'interaction avec la balle et déplacement plus fluide
- Interaction entre Cognard et Souaffle
- IA pour jouer seul

Copier / coller vos classes et interfaces à partir d'ici :

Package Start

```
package Start;

import java.awt.Color;
import java.util.*;


import quidditch.*;


/**
 * class Start permet d'afficher le menu d'accueil et avoir des informations sur le jeu et l'univers
 *
```

```
 * @author cyrena
 *
 */
public final class Start {

	private Start(){}

	public static void main(String[] args) {
		Scanner sc = new Scanner(System.in);
		if(start(sc) == 1)
			Jeu.GetInstance();
		else
			System.out.println("You chose to quit the game.\nSee you soon!");
	}

	/**
	* Cette fonction sert pour les menus : elle fait choisir l'utilisateur entre les différentes valeurs donné en argument.
	* En cas d'erreur, elle renvoie l'exception TropDErreurException.
	*/
	private static int choixVal(Scanner sc, int[] vals) throws TropDErreurException {
		int nbErreur = 0;
		int choice = 0;
		boolean ok = false;

      while (!ok) {
			choice = sc.nextInt();
			for (int val:vals) {
				if (choice == val) {
					ok = true;
					break;
				}
			}
			if (!ok) {
				nbErreur++;
					if (nbErreur > 4) throw new TropDErreurException("Nombre non valide");
				System.out.printf("Not valid number, try again.\n");
			}
		}
		return choice;
	}

	//Donne le choix au joueur du nombre de souaffle, Cognard et Vif d'Or
	public static int[] initBalls(Scanner sc) throws TropDErreurException {
		int[] nbBalls = new int[3];
		int choice;
		int nbErreur = 0;

		try{
			System.out.printf("How many Quaffles do you want?\n"+
				"\t 3 or 5 ? \n");
```

```
                int[] nb_souaffle = {3,5};
                nbBalls[0] = choixVal(sc,nb_souaffle);

                System.out.printf("How many Bludgers do you want?\n"+
                                        "\t 0, 1 or 2 ? \n");
                int[] nb_cognards = {0,1,2};
                nbBalls[1] = choixVal(sc,nb_cognards);

                System.out.printf("How many goldenSnitch do you want?\n"+
                                                "\t 0 or 1 ? \n");
                int[] nb_vif = {0,1};
                nbBalls[2] = choixVal(sc,nb_vif);

        } catch(Exception e){
                throw new TropDErreurException("You need to Enter a number");
        }
                 return nbBalls;
        }

        //Donne le choix au joueur de sa couleur(equipe) entre Gryffondor / Poufsouffle / Serdaigle /
Serpentard
        public static Joueur[] initPlayer(Scanner sc) throws TropDErreurException {
                Joueur[] player = new Joueur[2];
                int otherTeam = 0;
                String name;
                int team;
                int nbErreur = 0; //Au bout de 4 erreurs tout les paramètres seront choisis
aléatoirement
                try{
                        for (int i = 0 ; i < 2; i++) {
                                System.out.println("Player " + (i+1) + " : \nEnter your Name : \n");
                                name = sc.nextLine();
                                System.out.println("Player "+ (i+1) + " : " + name + "\n");
                                team = chooseTeam(sc, otherTeam);
                                sc.nextLine(); //this line consume the \n, otherwise the 2nd name can't
have an input
                                otherTeam = team;

                                if (team == 1)
                                        player[i] = new Joueur(name, i, Color.red);
                                else if (team == 2)
                                        player[i] = new Joueur(name, i, Color.blue);
                                else if (team == 3)
                                        player[i] = new Joueur(name, i, Color.yellow);
                                else if (team == 4)
                                        player[i] = new Joueur(name, i, Color.green);
                                else
                                        throw new TropDErreurException("Team not valid");
                        }

                } catch(Exception e){
                        throw new TropDErreurException("You need to Enter a string");
```

```
            }

                return player;
        }

        public static int chooseTeam(Scanner sc, int otherTeam) throws TropDErreurException{
                System.out.printf("Which team do you want to join? \n"+
                                "\t1. Gryffindor (red)\n"+
                        "\t2. Hufflepuff (blue)\n"+
                        "\t3. Ravenclaw (yellow)\n"+
                        "\t4. Slytherin (green)\n");

                int[] teams = {1,2,3,4};
                if (otherTeam != 0) {  //Si une équipe a déjà été choisie, on enlève du tableau.
                                                                        //Comme on utilise un tableau statique,
au lieu d'enlever on remplace par une autre valeur du tableau.
                        if (otherTeam == 1) {
                                teams[0] = teams[1];
                        } else {
                                teams[otherTeam-1] = teams[otherTeam-2];
                        }
                }

                return choixVal(sc,teams);
        }


        //menu de depart
        public static int start(Scanner sc){
    System.out.printf("What do you want to do ? \n"+
                "\t1. Start Quiddich Tournament !\n"+
                "\t2. Information\n"+
                "\t3. Quit\n");
    //try/catch affiche un message d'erreur si un int n'est pas rentré dans ligne de commande
    try{
        int choice = sc.nextInt();
        sc.nextLine();
        if (choice == 2){
            info(sc);
            start(new Scanner(System.in));
        }
        return choice;
    } catch(Exception e){
        throw new RuntimeException("Error.\nYou need to Enter a number");
    }
}

        //menu d'information
        public static void info(Scanner sc){
    System.out.printf("Which information do you want?\n"+
                "\t1. Game\n"+
                "\t2. Teams\n" +
```

```
                "\t3. Balls\n"+
                "\t4. Quit\n");
    //try/catch affiche un message d'erreur si un int n'est pas rentré dans ligne de commande
    try{
        int choice = sc.nextInt();
        sc.nextLine();
        if (choice == 1) {
            Information.jeu();
                Information.field();
        } else if (choice == 2)
                detailTeams(sc);
        else if (choice == 3)
            detailBalls(sc);
        else
                return;
        info(sc);
    } catch(Exception e){
        throw new RuntimeException("Error.\nYou need to Enter a number");
    }
}

    private static void detailBalls(Scanner sc) {
            Information.balls();
        System.out.printf("Which Ball information do you want?\n"+
                    "\t1. Quaffles\n"+
                    "\t2. Bludgers\n"+
                    "\t3. GoldenSnitch\n"+
                    "\t4. Quit\n");
          //try/catch affiche un message d'erreur si un int n'est pas rentré dans ligne de commande
        try{
            int choice = sc.nextInt();
            sc.nextLine();
            if (choice == 1)
                    Information.quaffles();
          else if (choice == 2)
            Information.bludgers();
          else if (choice == 3)
            Information.goldenSnitch();
          else
            return;
            detailBalls(sc);
        } catch(Exception e){
            throw new RuntimeException("Error.\nYou need to Enter a number");
        }
    }

    private static void detailTeams(Scanner sc) {
            Information.teams();
        System.out.printf("Which Team information do you want?\n"+
                    "\t1. Gryffindor\n"+
                    "\t2. Hufflepuff\n"+
                    "\t3. Ravenclaw\n"+
```

```
                        "\t4. Slytherin\n"+
                        "\t5. Quit\n");
                //try/catch affiche un message d'erreur si un int n'est pas rentré dans ligne de commande
            try{
                int choice = sc.nextInt();
                sc.nextLine();
                if (choice == 1)
                        Information.gryffindor();
                else if (choice == 2)
                  Information.hufflepuff();
                else if (choice == 3)
                   Information.ravenclaw();
                else if (choice == 4)
                   Information.slytherin();
                else
                   return;
                   detailTeams(sc);
                } catch(Exception e){
                   throw new RuntimeException("Error.\nYou need to Enter a number");
                }

        }
}


package Start;

public class TropDErreurException extends Exception {
        public TropDErreurException(String s) {
                super(s);
        }
}


package Start;

import java.io.*;



public interface Saveable {
        public void save(String file);

        public static void write_to_file(String fileName, String text) {
                try {
                        FileWriter file = new FileWriter(fileName);
                        file.write(text);
                        file.close();
                } catch(IOException e) {
                        System.out.println("Error in sabing youre data");
                }
        }
```

```
}
```

Package quidditch

```
package quidditch;

import java.awt.Graphics;
import Start.Saveable;


/**
 * classe abstraite dont heritent tous objets sur terrain visible avec une position,une taille et une vitesse
 * @author cyrena
 *
 */
public abstract class GameObject implements Saveable{
	protected int x, y;//position
	protected int velX, velY;//vitesse
	public final int id;
	public final int size;

	public GameObject(int x, int y, int id, int size) {
		this.x = x;
		this.y = y;
		this.id = id;
		this.size = size;
	}


	public abstract void tick();

	public abstract void render(Graphics g);

	public int getId() {
		return id;
	}

	public int getX() {
		return x;
	}

	public int getY() {
		return y;
	}

	public void setX(int newx) {
		x = newx;
	}

	public void setY(int newy) {
```

```
		y = newy;
	}

	public void setVelX(int vx) {
		velX = vx;
	}

	public void setVelY(int vy) {
		velY = vy;
	}

	public int getVelX() {
		return velX;
	}

	public int getVelY() {
		return velY;
	}

	public void save(String file) {
		Saveable.write_to_file(this.getClass() + "[x:"+x+", y:",+y+", velX:"+velX+",
velY:"+velY+"]\n");
	}
}

package quidditch;

/**
 * classe statique affichant toute sortes d'information sur le jeu
 * @author cyrena
 *
 */
public class Information {
	  private Information(){}

	   public static void jeu(){
	     System.out.println("Welcome in our game \"Quidditch Tournament\"! ");

		System.out.println("Tired of being a simple muggle? Take your chance and enter our
Quidditch Tournament.\n" +
				"It's your turn to ride your flying broomsticks and show your talent as the
best witch or wizard.\n" +
					"Be a part of your favorite House in Hogwarts and bring honour to
them.\n");

		System.out.println("A new way of playing this dangerous sport invented by author J.K.
Rowling for her fantasy book series Harry Potter.\n");

		System.out.println("\nHow to play :\n"+
				"1. Choose between solo and multiplayer mode.\n"+
				"2. Choose your house and name your wizard.\n"+
				"3. Pick how many Quaffles you want\n"+
```

```
                    "4. Choose to play with or without Bludgers and the Golden Snitch.\n" +
                    "5. Have fun !\n");
        }

        public static void teams(){
            System.out.println("\nHogwarts is divided into four houses, each bearing the last
name of its founder:\n"+
                               "Godric Gryffindor, Salazar Slytherin, Rowena Ravenclaw and
Helga Hufflepuff.\n"+
                               " Pick your values and color!\n");
        }

        public static void gryffindor(){
            System.out.println("Red: Gryffindor values courage, bravery, nerve, and chivalry.\n"
                               + " Gryffindor's mascot is the lion, and its colours are scarlet red and
gold\n");
        }

        public static void hufflepuff(){
            System.out.println("Yellow: Hufflepuff values hard work, patience, justice, and
loyalty.\n"
                               + " The house mascot is the badger, and canary yellow and black
\n");
        }

        public static void ravenclaw(){
            System.out.println( "Blue: Ravenclaw values intelligence, learning, wisdom and
wit.\n"
                               + " The house mascot is an eagle and the house colours are blue and
bronze.\n");
        }

        public static void slytherin(){
            System.out.println("Slytherin values ambition, cunning, leadership, and
resourcefulness.\n"
                              + "The house mascot of Slytherin is the serpent, and the house colours
are green and silver.\n");
        }

        public static void field(){
            System.out.println("\nThe field is pretty wide : "+ Jeu.WIDTH + " x " +
Jeu.HEIGHT + "sq feet.\n"+
                               "Both teams Hoops to score are highlighted.\n");
        }
        public static void balls(){
            System.out.println("\nA Quidditch can't start without balls. There are 3 kind of
ball.\n");
        }
        public static void quaffles(){
          System.out.println("\nYou can choose to play with 3 or 5 Quaffles\n"+
                             "Your goal is to bring all of them to the opponents' hoop it scores 10 points
and defend yours as well.\n"+
```

```
                        "The game finish once all the Quaffles got scored.\n");
        }
        public static void bludgers(){
          System.out.println("If you choose to play with Bludgers\n"+
                        "Both teams are attacked indiscriminately by the two Bludgers\n"+
                        "If you get hit by one of them your player while be shortly disoriented.\n");
        }

        public static void goldenSnitch(){
            System.out.println("If you choose to play with the Golden Snitch\n"+
                            "Catching the Snitch ends the game and scores 150 points.\n" +
                            "However catching it is pretty hard and it can get tricky if the other
team score all the Quaffles\n");
        }
}

package quidditch;

import Balles.*;
import Start.*;

import java.util.*; //need Random and Scanner and ArrayList
import java.awt.Canvas;
import java.awt.Color;
import java.awt.Graphics;
import java.awt.event.WindowEvent;
import java.awt.image.BufferStrategy;
import java.awt.Dimension;
import javax.swing.JFrame;
import java.io.*;


/**
 * class Jeu : suit le fil du jeu et affiche les logs
 * Il s'agit d'un singleton car une partie est unique
 * @author cyrena
 * Initialise :
 * - 2 Joueurs
 * - Les balles
 */

public class Jeu extends Canvas implements Runnable, Saveable{

        private static Jeu instance;

        public static final int WIDTH = 1000;
        public static final int HEIGHT = 640;
        public final int startSouaffle;

        private JFrame frame;
        private Thread thread;
        private boolean running = false;
```

```
	private UseGameO handler;

	private static int nbSouaffle = 0;
	private int nbCognard;
	private boolean vifOr = true;
	private static int[] score = {0, 0};
	private static Joueur[] player;


	public static Jeu GetInstance(){
		Scanner sc = new Scanner(System.in);
		if (instance == null)
			instance = pickRandom(sc);
		return instance;
	}

	private  Jeu(Joueur[] j, int[] nbBall){
		handler  = new UseGameO();
		this.addKeyListener(new KeyInput(handler));
		createWindow(WIDTH, HEIGHT, "Quidditch Tournament!");
		this.requestFocusInWindow();

		startSouaffle = nbBall[0];
		nbCognard = nbBall[1];
		vifOr = nbBall[2] == 1;

		handler.addObject(j[0]);
		handler.addObject(j[1]);
		player = j;
//		player = new Joueur[2];
//		player[0] = j[0];
//		player[1] = j[1];

		for(int i = 0; i < startSouaffle; i++) {
			handler.addObject(new Souaffle(handler, startSouaffle));
			nbSouaffle++;
		}
		for(int i = 0; i < nbCognard; i++)
			handler.addObject(new Cognard());
		if (vifOr)	handler.addObject(new VifOr());

		this.start();


	}

	private void createWindow(int width, int height, String title) {
		frame = new JFrame(title);

		frame.setPreferredSize(new Dimension(width, height));
		frame.setMaximumSize(new Dimension(width, height));
		frame.setMinimumSize(new Dimension(width, height));
```

```
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.setResizable(false);
                frame.setLocationRelativeTo(null);
                frame.add(this);
                frame.setVisible(true);
        }

        //choose random (code pas tres elabore mais nous permet d'effectuer nos test plus
rapidement)
        private static Jeu pickRandom(Scanner sc){
      System.out.printf("What do you want to do ? \n"+
                "\t1. Choose your own settings!\n"+
                "\t2. Random settings\n");
      Random rd = new Random();
      //try/catch affiche un message d'erreur si un int n'est pas rentré dans ligne de commande
      int choice = sc.nextInt();
      sc.nextLine(); //this line consume the \n, otherwise the 2nd name can't have an input
      if (choice == 1){
        try {
                return new Jeu(Start.initPlayer(sc), Start.initBalls(sc));
        } catch(TropDErreurException e) {
                System.out.println("You have made too many mistakes, so your settings will be
random.\nDo better next time !");
                //Pas de return, on sélectionne donc au hasard
        }
      }
        int[] b =  {rd.nextBoolean() ? 3 : 5,rd.nextInt(2), rd.nextInt(1)};
        return new Jeu(randomPlayer(), b);
    }

        private static Joueur[] randomPlayer(){
                Joueur j[] = new Joueur[2];
                j[0] = new Joueur("Player 1", 0,Color.red);
                j[1] = new Joueur("Player 2", 1,Color.green);
                return j;
        }

        public int getNbStartSouaffle() {
                return startSouaffle;
        }

        public static int getNbSouaffle() {
                return nbSouaffle;
        }

        public static void goal(int team) {
                score[team]++;
                System.out.println("Current score : " + score[0] +"\t-\t"+ score[1] + "\n");
                if (score[0] == score[1])
                        System.out.println("The battle is fierce both player are ex aequo!\n");
                else
```

```
                        System.out.println(player[score[0] > score[1]? 0 : 1].getName() + " takes the
lead!\n");
                nbSouaffle--;
                if (nbSouaffle == 0)
                        System.out.println(player[score[0] > score[1]? 0 : 1].getName() + " won the
tournament!\n");
        }

        // start thread of game
        public synchronized void start() {
                thread = new Thread(this);
                thread.start();
                running = true;
        }

        private void reset() {
                instance = null;
                score[0] = 0;
                score[1] = 0;
                GetInstance();
        }

        // stop thread of game
        public synchronized void stop() {
                try {
//                      frame.dispatchEvent(new WindowEvent(frame,
WindowEvent.WINDOW_CLOSING));
                        thread.join();
                        running = false;
                } catch (Exception e) {
                        e.printStackTrace();
                }

        }

        //D'après un code trouvé sur internet puis modifié pour qu'il nous convienne.
        public void run() {
                long lastTime = System.nanoTime();
                double amountOfTicks = 60.0; //Nombre de tick pas seconde
                double ns = 1000000000/amountOfTicks;//nb de ns par tick
                double delta = 0;
                long timer = System.currentTimeMillis();
                int frames = 0;
                int ticks = 0;
                boolean new_tick = true;
                while (running) {
                        long now = System.nanoTime();
                        delta += (now - lastTime)/ ns; //Nombre de tick à effectuer
                        lastTime = now;
                        while (delta >= 1) {//si jeu en cours on update
                                tick();
                                ticks++;
```

```
                        delta--;
                        new_tick = true;
                    }
                    if (running && new_tick) {//si jeu en cours et qu'il y a eu un tick, on affiche
                        render();
                        frames++;
                        new_tick = false;
                    }

                    if (System.currentTimeMillis() - timer > 1000) {
                        timer+= 1000;
//                      System.out.println("FPS :" + frames);//si on veut les fps
//                      System.out.println("TPS :" + ticks);//si on veut le nb de tick par
seconde
                        frames = 0;
                        ticks = 0;
                    }
                    if (nbSouaffle == 0) {

                        if (playAgain() == 1)
                            reset();
                        else {
                            save("save.quidditch");
                            frame.dispatchEvent(new WindowEvent(frame,
WindowEvent.WINDOW_CLOSING));
                            break;
                        }
                    }
                }
                stop();
        }

        private int playAgain() {
            Scanner sc = new Scanner(System.in);
            System.out.printf("Do you want to play again? \n"+
                    "\t1. Yes\n"+
                    "\t2. No\n");
          //try/catch affiche un message d'erreur si un int n'est pas rentré dans ligne de commande
          try{
            int choice = sc.nextInt();
            sc.nextLine(); //this line consume the \n, otherwise the 2nd name can't have an input
            return choice;
          } catch(Exception e){
            throw new RuntimeException("Error.\nYou need to Enter a number");
          }
        }

        //update all game object
        private void tick() {
            handler.tick();
        }
```

```
	private void render() {
		/**
		 * can create buffers at the launch of the game(bc null at beginning)
		 * fill the window with one color and the goals
		 */

		BufferStrategy bs = this.getBufferStrategy();
		if (bs == null) {
			this.createBufferStrategy(3);//nb of buffer recommended
			return;
		}

		Graphics g = bs.getDrawGraphics();


		g.setColor(Color.lightGray);
		g.fillRect(0, 0, WIDTH, HEIGHT-25);

		g.setColor(Color.darkGray);
		g.fillRect(0, 225, 10, 150);
		g.fillRect(WIDTH - 10, 225, 10, 150);

		handler.render(g);

		g.dispose();
		bs.show();
	}

	/**
	 * //empeche les joueurs de sortir du terrain
	 * @param x
	 * @param min
	 * @param max
	 * @return
	 */
	public static int limit(int x, int min, int max) {
		if (x >= max) return max;
		else if (x <= min) return min;
		else return x;
	}

	public static void win(int team) {
		nbSouaffle = 0;
	}

	@Override
	public void save(String file) {
		handler.save(file);
		String output = player[0].name + "(team " + player[0].team+") : " + score[0] +
"pts\n";
		output += player[1].name + "(team " + player[1].team+") : " + score[1] + "pts\n";
```

```
                Saveable.write_to_file(file,output);
        }

}

package quidditch;

import java.awt.Color;
import java.awt.Graphics;
import java.util.Random;

import Balles.Cognard;


/**
 *      Joueur :  A chaque partie il y a 2 joueurs = un dans chaque equipe
 *
 * @author cyrena
 *
 *      @summary
 *      -       Un joueur et caracterise par:
 *      -       Son nom
 *      -       Son equipe (0 commence a gauche et 1 a droite)
 *      -       Sa couleur (= la maison de son choix)
 *      -       positionne joueur a meme hauteur, au centre de leur cote du terrain
 *
 */

public class Joueur extends GameObject {
        public final String name;
        public final int team;
        public final Color c;
        private int stunned;
        //ajout barre vie -> si on se fait toucher trop de fois par cognard = mort ?

        public Joueur(String nom, int team, Color c) {
                super(team == 0 ? Jeu.WIDTH / 4 : Jeu.WIDTH * 3 / 4 , Jeu.HEIGHT / 2, team, 35);
                this.name = nom;
                this.team = team;
                this.c = c;
                this.stunned = 0;
        }

        public Joueur(Joueur player) {
                super(player.team == 0 ? Jeu.WIDTH / 4 : Jeu.WIDTH * 3 / 4 , Jeu.HEIGHT / 2,
player.team, 35);
                this.name = player.name;
                this.team = player.team;
                this.c = player.c;
                this.stunned = 0;
        }
```

```
	public void getKnocked() {
		stunned = Cognard.STUN_DURATION;
	}

	public void win() {
		System.out.println(name + " caught the Golden Snitch and won the
Tornament!\nIncredible!\n");
		Jeu.win(team);
	}

	public int getTeam() {
		return team;
	}

	public String getName() {
		return name;
	}

	public Joueur clone() {
		return new Joueur(this);
	}

	@Override
	public void tick() {
		if (stunned == 0) {//si joueur pas sonne se deplace
			x = Jeu.limit(x + velX, 0, Jeu.WIDTH-35);
			y = Jeu.limit(y + velY, 0, Jeu.HEIGHT-60);
		} else
			stunned--;
	}

	@Override
	public void render(Graphics g) {
		g.setColor(c);
		g.fillRect(x, y, size, size);
	}

}

package quidditch;

import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
/**
 * Gestion des touches de claviers utilise pour bouger les joueurs
 * Attention pour set pour clavier querty
 * @author cyrena
 *
 */
public class KeyInput extends KeyAdapter{
	private UseGameO handler;
```

```
	public  KeyInput(UseGameO handler) {
		this.handler = handler;
	}

	public void keyPressed(KeyEvent e) {
		int key = e.getKeyCode();

		for (int i = 0; i < handler.objects.size(); i++) {
			GameObject tmpObj = handler.objects.get(i);

			if(tmpObj.getId() == 0) {
				if(key ==  KeyEvent.VK_W )tmpObj.setVelY(-3);
				if(key ==  KeyEvent.VK_S)tmpObj.setVelY(3);
				if(key ==  KeyEvent.VK_D )tmpObj.setVelX(3);
				if(key ==  KeyEvent.VK_A )tmpObj.setVelX(-3);
			}
			if(tmpObj.getId() == 1){
				if(key ==  KeyEvent.VK_UP )tmpObj.setVelY(-3);
				if(key ==  KeyEvent.VK_DOWN )tmpObj.setVelY(3);
				if(key ==  KeyEvent.VK_RIGHT )tmpObj.setVelX(3);
				if(key ==  KeyEvent.VK_LEFT)tmpObj.setVelX(-3);
			}
		}
	}

	public void keyReleased(KeyEvent e) {
		int key = e.getKeyCode();

		for (int i = 0; i < handler.objects.size(); i++) {
			GameObject tmpObj = handler.objects.get(i);

			if(tmpObj.getId() == 0) {
				if(key !=  KeyEvent.VK_A )tmpObj.setVelX(0);
				if(key !=  KeyEvent.VK_W )tmpObj.setVelY(0);
				if(key !=  KeyEvent.VK_D )tmpObj.setVelX(0);
				if(key !=  KeyEvent.VK_S)tmpObj.setVelY(0);

			}else if(tmpObj.getId() == 1){
				if(key !=  KeyEvent.VK_UP )tmpObj.setVelY(0);
				if(key !=  KeyEvent.VK_DOWN )tmpObj.setVelY(0);
				if(key !=  KeyEvent.VK_RIGHT )tmpObj.setVelX(0);
				if(key !=  KeyEvent.VK_LEFT)tmpObj.setVelX(0);
			}
		}
	}
}

package quidditch;

import java.awt.Graphics;
import java.util.ArrayList;
```

```
import Balles.*;
import Start.Saveable;
//import java.util.LinkedList;

/**
 *	UseGameO est une classe qui permet de gerer les gameObject
 *
 * @author cyrena
 *
 *	@summary
 *	Pour les ajouter , enlever ou juste update les GO
 *	garde aussi une liste des objets en collision a un instant t
 *
 */

public class UseGameO implements Saveable {
	ArrayList<GameObject> objects = new ArrayList<GameObject>();


	/**
	* Cette méthode est appelé régulièrement, elle est responsable de faire évoluer l'état du jeu.
	* Elle va donc appeler la méthode tick de chaque objet, et gérer les collisions.
	*/
	public void tick() {
		for (int i = 0; i < objects.size(); i++) {
			GameObject tmpObject = objects.get(i);
			tmpObject.tick();
		}

		for (GameObject[] coll: collisions()) {//pour chaque couple en collision
			GameObject o1 = coll[0];
			GameObject o2 = coll[1];
//			System.out.println(o1 + " " + o2);//log les 2 objets en collision

			if (o1 instanceof Joueur) {
				if (o2 instanceof Souaffle)
					((Souaffle) o2).pousse((Joueur)o1);
				else if (o2 instanceof Cognard)
					((Joueur)o1).getKnocked();
				else if (o2 instanceof VifOr)
					((Joueur)o1).win();
			}
			if (o1 instanceof Souaffle)
				if (o2 instanceof Joueur)
					((Souaffle) o1).pousse((Joueur) o2);

			if (o1 instanceof Cognard) {
				if (o2 instanceof Joueur)
					((Joueur) o2).getKnocked();
				if (o2 instanceof Cognard) {
					Cognard n = ((Cognard) o2).duplicate((Cognard) o1);
					if (n != null) {
```

```
                                        objects.add(n);
                                    }
                                }
                            }
                            if (o1 instanceof VifOr)
                                if (o2 instanceof Joueur)
                                    ((Joueur) o2).win();
                        }
                    }

                    /**
                    * Cette méthode va afficher tout les objets. Elle est appelé régulièrement pour raffraichir l'écran.
                    */
                    public void render(Graphics g) {
                        for (int i = 0; i < objects.size(); i++) {
                            GameObject tmpObject = objects.get(i);
                            tmpObject.render(g);
                        }
                    }

                    /**
                    * Retourne une liste de tout les couple d'objects qui sont en collision.
                    * (on considère que tout les objets sont des carrés pour plus de simplicité)
                    */
                    public ArrayList<GameObject[]> collisions() {
                        ArrayList<GameObject[]> lst = new ArrayList<GameObject[]>();
                        for (int i=0; i < objects.size();i++) {
                            GameObject object = objects.get(i);
                            for (int j=i+1; j < objects.size(); j++) { //On ne teste que les objets derrières pour ne pas avoir deux fois les couples.
                                GameObject other_object = objects.get(j);
                                int diff_x_1 = Integer.signum(object.getX() - other_object.getX() - other_object.size);
                                int diff_x_2 = Integer.signum(object.getX() + object.size - other_object.getX());
                                int diff_y_1 = Integer.signum(object.getY() - other_object.getY() - other_object.size);
                                int diff_y_2 = Integer.signum(object.getY() + object.size - other_object.getY());
                                if ((diff_x_1 != diff_x_2 || diff_x_1 == 0) && (diff_y_1 != diff_y_2 || diff_y_1 == 0)) {
                                    GameObject[] temp = {object,other_object};
                                    lst.add(temp); //On ne teste pas si object == other_object car la manière dont on fait la boucle garantie de ne pas avoir ce cas
                                }
                            }
                        }
                        return lst;
                    }
```

```
	public Joueur getPlayer(int p) { //get which player according to id
		int i = 0;
		assert(i == 0 || i == 1);//id 0 and 1 are the only player

		while (i < objects.size()) {
			if (objects.get(i).getId() == p)
				return (Joueur)objects.get(i);
			i++;
		}
		return null;
	}

	public void addObject(GameObject newObject) {
		objects.add(newObject);
	}

	public void removeObject(GameObject supObject) {
		objects.remove(supObject);
	}

	public void save(String file) {
		for (GameObject ob: objects) {
			ob.save(file);
		}
	}
}
```

Package Balles

```
package Balles;

import java.awt.Color;
import java.awt.Graphics;
import java.util.Random;
import quidditch.*;

/**
 * Toutes les balles rebondissent sur les parois de la map
 * et ont une forme ronde avec une couleur differente et des actions differentes
 * @author cyrena
 *
 */
public abstract class Balle extends GameObject {
	protected int count;


	public Balle(int x, int y , int id, int size) {
		super(x, y, id, size);
		count = 0;
	}

	public void tick() {
```

```
		inside();
	}

	public void inside() {
		if(x <= 0 || x >= Jeu.WIDTH - this.size)
			velX *= -1;
		if(y <= 0 || y >= Jeu.HEIGHT - this.size - 25)
			velY *= -1;
	}

	protected void render(Graphics g, Color c) {
			g.setColor(c);
			g.fillOval(x, y, size, size);
	}
}

package Balles;

import java.awt.Color;
import java.awt.Graphics;
import java.util.Random;
import quidditch.*;

/**
 * /**
 * Les cognards sont toujours mobile et apparaissent aleatoirement sur le terrain au debut
 * leur vitesse est constante
 * On peut avoir 0,1 ou 2 Cognards au depart.
 * Ils bougent si un joueur entre en collision avec eux il sera immobilise pendant un certain temps
 * @author cyrena
 *
 */
public class Cognard extends Balle{
	public static final int STUN_DURATION = 60;
	public int timer;

	public Cognard() {
		super((int)(Math.random()*(Jeu.WIDTH - 50) + 30),
(int)(Math.random()*(Jeu.HEIGHT-50)) + 30, 3, 25);
		velX = 2;
		velY = 2;
		timer = 0;
	}


	public Cognard(int x, int y) {
		super(x, y, 3, 25);
		velX = 2;
		velY = 2;
	}

	public Cognard clone() {
```

```
		return new Cognard(x, y);
	}

	public Cognard duplicate(Cognard c) {
		Cognard new_cognard = null;
		if (this.canDuplicate() && c.canDuplicate()) {
			new_cognard = clone();
			new_cognard.velX = - velX;
			new_cognard.velY = - velY;
			this.timer = 60;
			c.timer = 60;
			//Quand deux cognard se cognent, un d'entre eux se dédouble
		}
		return new_cognard;


	}

	@Override
	public void tick() {
		if (timer != 0) timer--;
		if (Math.random() < 0.008) { //0.8% de chance que le cognard change de direction
			if (Math.random() > 0.5)
				velX = 2;
			else
				velX = -2;
			if (Math.random() > 0.5)
				velY = 2;
			else
				velY = -2;
		}
		x += velX;
		y += velY;
		super.tick();//keep balls inside
	}

	public void render(Graphics g) {
		super.render(g, Color.black);
	}

	public boolean canDuplicate() {
		return timer == 0;
	}

}

package Balles;

import java.awt.Color;
import java.awt.Graphics;
import java.util.Random;
import quidditch.*;
```

```
/**
 * Les souaffles sont immobile au depart et apparaissent sur la ligne centrales.
 * Leur ordonne de depart depend de si on commencent avec 3 ou 5 souaffles.
 * Ils bougent si un joueur entre en collision avec eux et
 * perdent de la vitesse avec le temps si aucune nouvelle collision n'est faite
 * Si un souaffle entre dans un goal il donne un point a l'equipe adverse
 * @author cyrena
 *
 */
public class Souaffle extends Balle{
	private UseGameO handler;

	public Souaffle(UseGameO handler, int nbStart) {
		super((int)(Jeu.WIDTH/2), calculPlace(nbStart), 2, 35);
		this.handler = handler;
	}

	private static int calculPlace(int nbStart) {
		int implement = nbStart == 3 ? 200 : 125;
		return nbStart == 3 ? 100 + Jeu.getNbSouaffle() * implement : 50 +
Jeu.getNbSouaffle() * implement;
	}

	//interraction avec joueur
	public void tick() {
		//   une vitesse decroissante, aka force de frottement (en fonction des ticks)
		if (count == 70){
			if (velX > 0) velX--;
			if (velX < 0) velX++;
			if (velY > 0) velY--;
			if (velY < 0) velY++;
			count = 0;
		} else
			count++;

//On verifie si le souaffle n'entre pas dans une des cages
		if ( y >= 180 && y <= 360) {
			if (x <= 5) {//entre dans goal gauche
				System.out.println("Team 2 Scored!\n"); // ajouter le score a l'equipe 1
				handler.removeObject(this);
				Jeu.goal(1);
			}
			else if ( x >= Jeu.WIDTH - 35){//entre dans goal gauche
				System.out.println("Team 1 Scored!\n");// ajouter le score a l'equipe 0
				handler.removeObject(this);
				Jeu.goal(0);
			}
		}
//empeche la balle de sortir
		x = Jeu.limit(x + velX, 0, Jeu.WIDTH-35);
		y = Jeu.limit(y + velY, 0, Jeu.HEIGHT-60);
```

```
		super.tick();//keep balls inside
	}

//gagne la vitesse du joueur avec lequel le souaffle entre en collision
	public void pousse(Joueur j) {
		velX += j.getVelX();
		velY += j.getVelY();
	}

	public void render(Graphics g) {
		super.render(g, Color.magenta);//couleur magenta
	}
}
```

```
package Balles;

import java.awt.Color;
import java.awt.Graphics;
import java.util.Random;
import quidditch.*;
/**
 * Le vif d'or apparait de maniere aleatoire pendant des temps aleatoire sur le terrain
 * pour simplifier les choses on le positionne juste en dehors du terrain lorsqu'il est invisible
 * si le joueur touche le vif d'or il gagne automatiquement la partie
 * @author cyrena
 *
 */
public class VifOr extends Balle{
	private boolean visible;

	public VifOr() {
		super((int)(Math.random()*Jeu.WIDTH), (int)(Math.random()*Jeu.HEIGHT), 4,
15);
		visible = true;
	}

	@Override
	public void tick() {
		if ( 0 > count) {
			x = visible ? (int)(Math.random() * Jeu.WIDTH) : -20;
			y = visible ? (int)(Math.random() * Jeu.HEIGHT) : -20;
			count = (int)(Math.random()* 100) + 50;
			visible = !visible;
		} else
			count--;
	}

	public void render(Graphics g) {
		super.render(g, Color.yellow);
	}

}
```