Master Informatique **Approche Objet – 4TIN706U**
Université Bordeaux **2016 – 2017**

TD N^o1 Une simulation de mécanique

---

1 Simulation d'une balle en chute libre

Nous proposons le code suivant qui permet de simuler une expérience de physique au lycée. Nous rappelons les lois de cinématique suivantes :

— $x_f = x_i + v_i t + \frac{1}{2} a t^2$
— $v_f = v_i + a\Delta t$
— $a = F/m$

Avec x_i position initiale, x_f position finale, v_i vitesse initiale, v_f vitesse finale, a accélération, $\Delta k = k_f - k_i$

Le programme suivant, écrit en Java, permet de simuler la chute d'un ballon parfaitement élastique de 35cm de diamètre, de 1kg, lancé à 4m du sol avec une vitesse horizontale de $3{,}6km/h$:

Exercices

1. Ajouter à ce programme des propriétés d'absorption au sol de sorte qu'il paraisse plus réaliste (on fera une simulation en diminuant très la vitesse à chaque rebond).
2. Ajouter un second ballon.

   Remarquez que l'ajout de ce second ballon pose un problème de conception du programme : comment faire pour que les données du second ballon ne soient pas mélangées avec celles du premier ?

   Les dernières lignes du code doivent vous inviter à concevoir ce programme par l'approche objet. Motivez les raisons qui vous incitent à faire ce choix de programmation.
3. Créer une classe Ball dans laquelle l'ensemble des propriétés d'un ballon ou d'une balle seront regroupées (diamètre, masse, coefficient d'élasticité, position, vitesse, accélération).
4. Créer l'ensemble des méthodes propres à un ballon.

2 Simulation d'une balle attachée par un élastique (Jokari)

Nous rappelons qu'un ressort ou élastique idéal produit une force opposée proportionnelle au déplacement. L'élastique ne produit cette force qu'à partir d'une distance minimale.

Exercices

1. Créer une classe JokariBall qui implémente une balle d'un diamètre de 15cm d'une masse de 500g attachée à un élastique au sol et au centre.
2. Les balles de Jokari et les ballons ont des types de propriétés semblables. Ils ont des positions dans l'espace, une masse, un phénomène de rebond élastique, etc. Organiser le code en utilisant la hiérarchie de classe ou la composition d'objets pour exploiter au mieux ces types semblables.

3 Pour aller plus loin

Exercices

1. Ajouter une liste de balles de Jokari et une liste de ballons au jeu.
2. Ajouter un objet qui gère l'ensemble des collisions entre les ballonset les balles.

Code fourni [1]

Listing 1 – Main.java

```
1  import java.awt.Graphics;
   import java.awt.Graphics2D;
3  import java.awt.Rectangle;
   import java.awt.event.ActionEvent;
5  import java.awt.event.ActionListener;
   import java.awt.geom.Ellipse2D;
7  import javax.swing.JPanel;
   import javax.swing.Timer;
9
   @SuppressWarnings("serial")
11 public class Animation extends JPanel implements ActionListener {

13     // Quelques constantes
       final double G = 9.80665; // acceleration de la gravitation
15     final int STEP = 15; // duree de rafraichissement de l'ecran: 15ms
       final double MSSTEP = STEP / 1000.0;
17
       // Marges en pixels
19     final int MARGIN = 32;

21     // Echelle: 40 pixels = 1m
       final double SCALE = 40;
23
       // La balle
25     Ellipse2D.Double ellipse = new Ellipse2D.Double();

27     // coordonnees du plan
       double xmin, xmax, ymin, ymax;
29
       double t; // temps relatif
31     double x, y; // position
       double vx, vy; // vitesse
33     double ax, ay; // acceleration
       double x0, y0; // position initiale
35     double vx0, vy0; // vitesse initiale
       double fx, fy; // force
37
       // masse de la balle
39     final double M = 1; // 1 kg

41     // Taille de la balle
       final double W = 0.35; // 35 cm
43     final double H = 0.35;
       final double R = W/2;
45
       Timer timer;
47
       // initialise
49     public Animation(int width, int height) {
           super();
```

1. le code initial est disponible à cette URL : `www.labri.fr/perso/clement/enseignements/ao/`

```
51          setBounds(new Rectangle(0, width, 0, height));
            t = 0;
53          xmin = 1; // coordonnees du plan en metres
            ymin = 1;
55          xmax = 11;
            ymax = 11;
57          fx = 0;
            fy = M * G;
59          ax = fx / M; // acceleration initiale
            ay = fy / M;
61          vx0 = vx = 1; // vitesse initiale 3,6 km/h
            vy0 = vy = 0;
63          x0 = x = xmin + 5; // position initiale
            y0 = y = ymax - 4;
65          setXY(x, y);

67          // Timer permet de lancer actionPerformed a periodes regulieres
            timer = new Timer(STEP, this);
69          timer.start();
        }
71
        // repositionne la balle selon ses coordonnees
73      void setXY(double x, double y) {
            ellipse.setFrame(x * SCALE + MARGIN, y * SCALE - MARGIN,
75                           W * SCALE, H * SCALE);
        }
77
        // Procedure a chaque etape
79      void step() {
            x = x0 + vx0 * t + (ax * t*t) / 2;
81          y = y0 + vy0 * t + (ay * t*t) / 2;
            vx = vx0 + ax * t;
83          vy = vy0 + ay * t;
            if(x > xmax-W || x < xmin){
85              vx0 = -vx;
                vy0 = vy ;
87              x0 = x;
                y0 = y;
89              t = 0;
            }
91          if(y > ymax-H || y < ymin){
                vx0 = vx;
93              vy0 = -vy;
                x0 = x;
95              y0 = y;
                t = 0;
97          }
            t = t + MSSTEP;
99          setXY(x, y);
        }
101
        private final int ARR_SIZE = 4;
103
        // affichage
105     public void paint(Graphics g) {
```

```
        this.step();
        this.paintComponent(g);
        ((Graphics2D)g).drawLine((int)(xmin * SCALE) + MARGIN,
                                 (int)(ymax * SCALE) - MARGIN,
                                 (int)(xmax * SCALE) + MARGIN,
                                 (int)(ymax * SCALE) - MARGIN);
        for (double i=xmin ; i<=xmax ; i++){
            ((Graphics2D)g).drawLine((int)(i * SCALE) + MARGIN,
                                     (int)(ymax * SCALE) - MARGIN + 2,
                                     (int)(i * SCALE) + MARGIN,
                                     (int)(ymax * SCALE) - MARGIN - 2);
        }
        ((Graphics2D)g).drawLine((int)(xmin * SCALE) + MARGIN,
                                 (int)(ymin * SCALE) - MARGIN,
                                 (int)(xmax * SCALE) + MARGIN,
                                 (int)(ymin * SCALE) - MARGIN);
        ((Graphics2D)g).drawLine((int)(xmin * SCALE) + MARGIN,
                                 (int)(ymin * SCALE) - MARGIN,
                                 (int)(xmin * SCALE) + MARGIN,
                                 (int)(ymax * SCALE) - MARGIN);
        for (double j=ymin ; j<=ymax ; j++){
            ((Graphics2D)g).drawLine((int)(xmin * SCALE) + MARGIN - 2,
                                     (int)((ymax - j) * SCALE) - MARGIN,
                                     (int)(xmin * SCALE) + MARGIN + 2,
                                     (int)((ymax - j) * SCALE) - MARGIN);
        }
        ((Graphics2D)g).drawLine((int)(xmax * SCALE) + MARGIN,
                                 (int)(ymin * SCALE) - MARGIN,
                                 (int)(xmax * SCALE) + MARGIN,
                                 (int)(ymax * SCALE) - MARGIN);

        // Ballon
        ((Graphics2D)g).draw(ellipse);

    }

    public void actionPerformed(ActionEvent e) {
        this.repaint();
    }

}
```

Listing 2 – Main.java

```
import javax.swing.JFrame;


public class Main {

          public static void main(String[] args) {
                    JFrame frame = new JFrame("Animation");
                    Animation animation = new Animation(600, 500);
                    animation.setLayout(null);

                    frame.add(animation);
                    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
                frame.setSize(600, 500);
14              frame.setLocationRelativeTo(null);
                frame.setVisible(true);
16          }

18  }
```