Brian Gómez Martínez

Marco Rodríguez Moyses

PANG

1º DAW

ÍNDICE

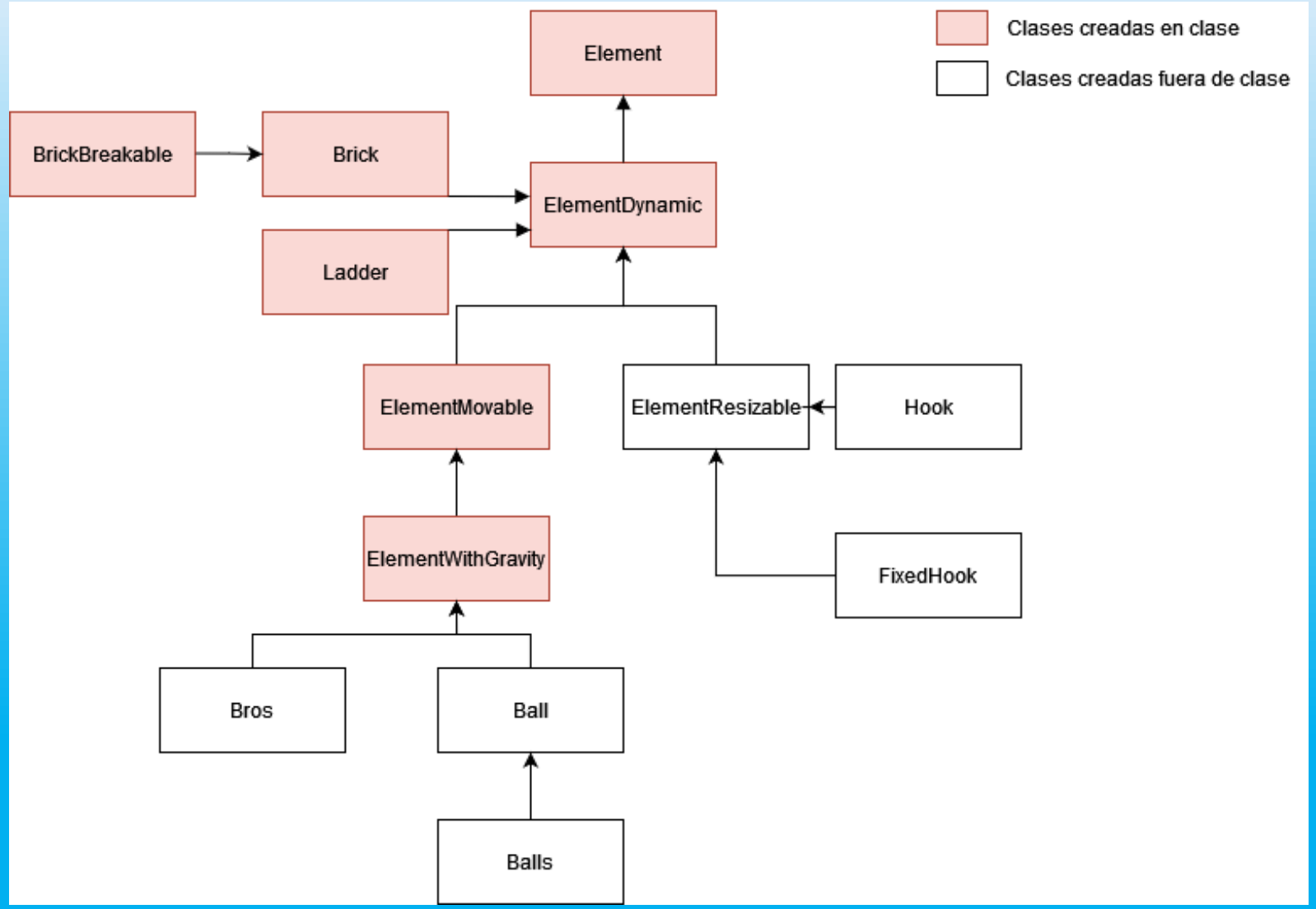
- **Introducción**
- **Clases creadas**
- **Detalles / Metodos importantes de cada clase**
- **Conclusiones**

INTRODUCCIÓN

En este proyecto hemos realizado en java una versión “Alpha” del juego Pang también llamado Buster Bros lanzado en 1989 desarrollado por Mitchell Corporation.

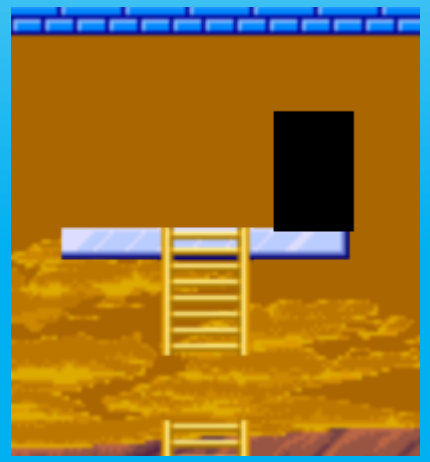
- **https://github.com/Brian-GM/Pang**

CLASES CREADAS



BOARD: COLISION ESCALERA Y BRICK

```
public Boolean collisionJugLadder(Bros b, Element e) {

    Boolean collision = false;

    for (int i = 0; i < this.elements.length; i++) {

        if (this.elements[i] instanceof Ladder && !collision && this.elements[i] instanceof Ladder && b.getRectangle().intersects(e.getRectangle())) {

            collision = true;

            System.out.println("Escalera");

            return collision;

        }

    }

   return collision;

  }
```



```
public Boolean collisionJugBrick(Bros b, Element a) {
        Boolean collisionbrick = false;
        for (int i = 0; i < this.elements.length; i++) {
            if (this.elements[i] instanceof Brick) {
                a = this.elements[i];
                if (b.getRectangle().intersects(a.getRectangle())) {
                    collisionbrick = true;
                    System.out.println("Brick");
                    return collisionbrick;
                }
            }
        }
        return collisionbrick;
    }
```

BOARD: WIN Y LOSE

WIN

```
private boolean win() {

    boolean vacio = true;

    for (int i = 0; i < balls.getSize(); i++) {

        if (this.balls.getBall(i) != null) {

            vacio = false;

        }

    }

    return vacio;

  }
```

LOSE

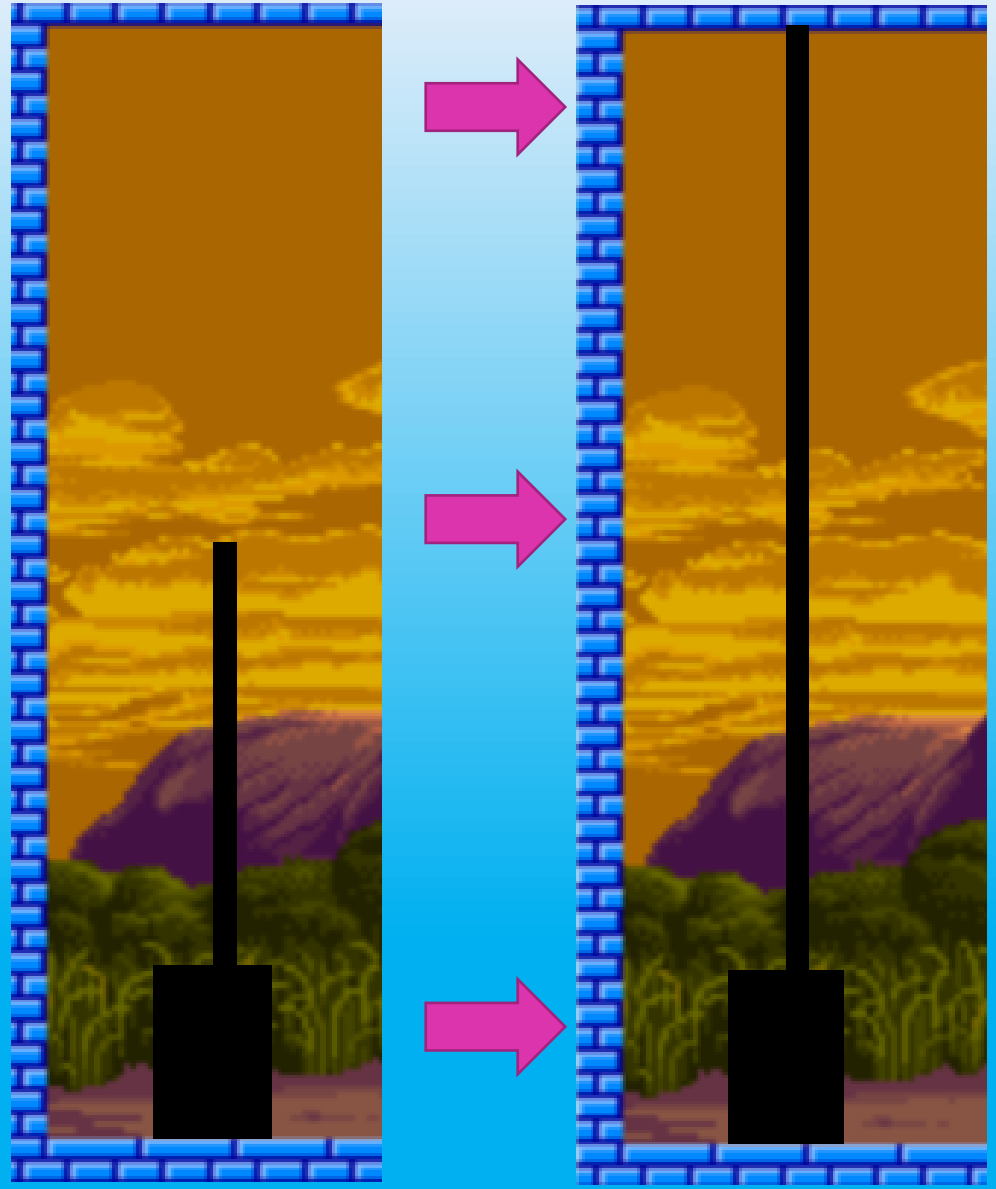
```
private void lose() {

    if (this.jugador != null && this.jugador.getLifes() == 0) {
        this.jugador.stop();
        this.jugador.setGy(0);

        for (int i = 0; i < balls.getSize(); i++) {
            if (this.balls.getBall(i) != null) {
                this.balls.getBall(i).stop();
                this.balls.getBall(i).setGy(0);
            }
        }
    }

  }
```

BOARD: HOOK Y FIXED HOOK

```
case S:

        gancho_fijo = new Hook(2, 0.1, 2, (int) jugador.getCenterX(), (int) jugador.getCenterY(), 4, 1, 1);

case A:

        gancho = new FixedHook(2, 0.1, 2, (int) jugador.getCenterX(), (int) jugador.getCenterY(), 4, 1, 1);

    }
```

```
private void resizeGanchoNormal(){
if (this.gancho != null) {
        if (this.gancho.collisionTop(game_zone)) {
            this.gancho = null;
        }
    }
    if (this.gancho != null) {
        this.gancho.resizeHeigth();
        this.gancho.paint(gc);
    }
}
```

```
private void resizeGanchoFijo(){

    if (this.gancho_fijo != null) {
        this.gancho_fijo.resizeHeigth();
        this.gancho_fijo.paint(gc);
    }

    if (this.gancho_fijo != null && this.gancho_fijo.collisionTop(game_zone)) {
        this.gancho_fijo.stop();
    }

}
```



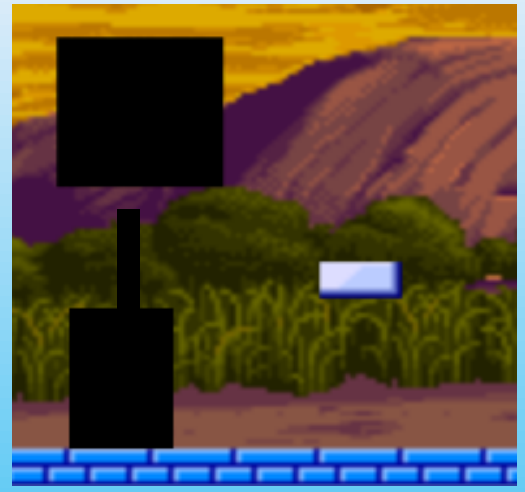
BOARD:COLISION GANCHOS CON BOLAS

```
public void explosion(Ball b, ElementResizable g) {

    if (this.collisionBolaGancho(b, g)== true) {

        Ball tempo = this.balls.getFirst();

        Ball[] bs = tempo.explotar();

        this.balls.removeBall(tempo);

        if (bs != null) {

            this.balls.addBall(bs[0]);

            this.balls.addBall(bs[1]);

        }

    }

}
```

```
//Comprueba si hay colision entre las bolas y el gancho normal
    for (int i = 0; i < this.balls.getSize(); i++) {
        if (this.balls.getBall(i) != null && this.gancho != null) {
            this.explosion(this.balls.getBall(i), gancho);

        }
    }
```

```
//Comprueba si hay colision entre las bolas y el gancho fijo
    for (int i = 0; i < this.balls.getSize(); i++) {
        if (this.balls.getBall(i) != null && this.gancho_fijo != null) {
            this.explosion(this.balls.getBall(i), gancho_fijo);
        }
    }
```

BALL: EXPLOTAR



```
public Ball[] explotar() {

    Ball[] childrens = new Ball[2];

    BallType next = this.getNext();


    if (next != null) {

        childrens[0] = new Ball(next, this.getColor(), this.getGx(), this.getGy(), true, true,
this.getOriginal_vx(), this.getOriginal_vy(), this.getCenterX(), this.getCenterY(),
this.getWidth(), this.getHeight());

        childrens[1] = new Ball(next, this.getColor(), this.getGx(), this.getGy(), true, true,
-this.getOriginal_vx(), -this.getOriginal_vy(), this.getCenterX(), this.getCenterY(),
this.getWidth(), this.getHeight());

        return childrens;


    }

    return null;

}
```

BALL: REBOTAR CON BRICK

```
public Optional<Collision> Collision(Element e) {

        Optional<Collision> c = super.Collision(e);

        if (c.isPresent()) {

            System.out.println("pedro.ieslaencanta.com.busterbros.balls.Ball.collision()");

            double dx = this.evalCollisionX(e);

            double dy = this.evalCollisionY(e);

            c.get().setSeparator(new Point2D(dy, dx));

        }

        return c;

    }
```

```
//Detecta la colision en el eje x
    private double evalCollisionX(Element e) {
        Rectangle2D x = new Rectangle2D(this.rectangle.getMinX(), this.rectangle.getMinY() - this.getVy(),
this.rectangle.getWidth(), this.rectangle.getHeight());
        if (e.getRectangle().intersects(x)) {
            if (this.getCenterX() < e.getCenterX()) {
                return Math.abs(e.getCenterX() - this.getCenterX() - (this.getWidth() / 2 + e.getWidth() / 2));
            } else {
                return Math.abs((this.getCenterX() - e.getCenterX()) - (this.getWidth() / 2 + e.getWidth() / 2));
            }
        } else {
            return 0;
        }
    }
```

```
//Detecta la colision en el eje y
private double evalCollisionY(Element e) {
        Rectangle2D y = new Rectangle2D(this.rectangle.getMinX(), this.rectangle.getMinY() - this.getVy(),
this.rectangle.getWidth(), this.rectangle.getHeight());
        if (e.getRectangle().intersects(y)) {
            if (this.getCenterY() < e.getCenterY()) {
                return Math.abs(e.getCenterY() - this.getCenterY() - (this.getHeight() / 2 + e.getHeight() / 2));
            } else {
                return Math.abs((this.getCenterY() - e.getCenterY()) - (this.getHeight() / 2 + e.getHeight() / 2));
            }
        } else {
            return 0;
        }
    }
```

FIN

- Gracias por su tiempo y su atención.