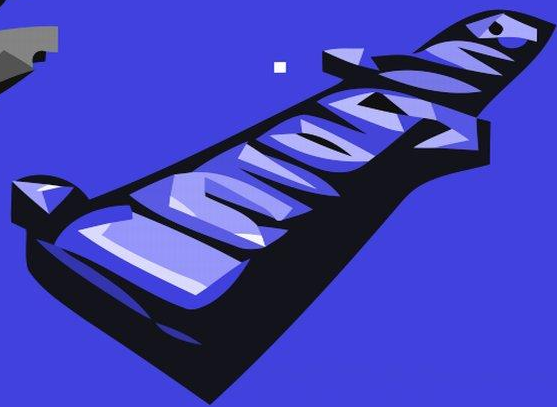


# WKNOP

**Por Bruno y Manuel**



# REALIZACIÓN ARKANOID



- Barra, movimiento y colisiones
- Pelota, movimiento y colisiones
- Ladrillo y colisiones
- Niveles



# BARRA

Primer paso, inicializar la barra en el constructor de la clase “Campo” dándole así unas propiedades como su X, Y, Ancho y Alto.

```
this.barra = new Barra(new Point2D(anchopixels / 2 - 33, altopixels - 29), 30, 14);
```

```
public void moverBarraIzquierda() {
    if (this.campo.getBarra() != null) {
        if (this.campo.getBarra().getPosicion().getX() > this.campo.getBorde()) {
            this.campo.getBarra().moverIzquierda();
            if (this.estado == EstadoJuego.PENDIENTEINICIAR) {
                this.colocarPelotaEnBarra();
            }
        }
    }
}

public void moverBarraDerecha() {
    if (this.campo.getBarra() != null) {
        if (this.campo.getBarra().getPosicion().getX() + this.campo.getBarra().getAncho()
            < Campo.getWidth() - this.campo.getBorde()) {
            this.campo.getBarra().moverDerecha();
            if (this.estado == EstadoJuego.PENDIENTEINICIAR) {
                this.colocarPelotaEnBarra();
            }
        }
    }
}
```

Segundo paso, movimiento y colisiones con Campo, tenemos que si la X de la barra es mayor que el borde, la barra puede moverse, así cuando llegue a el borde, se frenará en la izquierda y similar con la derecha, solo añadimos el ancho de Campo.

También hemos creado un estado para poner la pelota encima de la barra si este es igual a “PENDIENTEINICIAR”.



# BARRA

En la clase Juego, tenemos un método que sirve para detectar los estados del juego y realizar ciertas acciones. En este caso, al pulsar los botones adecuados, este ejecutará los métodos de mover la barra.

```
public EstadoCambiosJuego ciclo(boolean pulsados[]) {  
    EstadoCambiosJuego vuelta = EstadoCambiosJuego.NADA;  
  
    if (pulsados[0]) {  
        this.moverBarraIzquierda();  
    }  
    if (pulsados[1]) {  
        this.moverBarraDerecha();  
    }  
}
```

Botones en la clase “InterfazGrafica” para el movimiento.

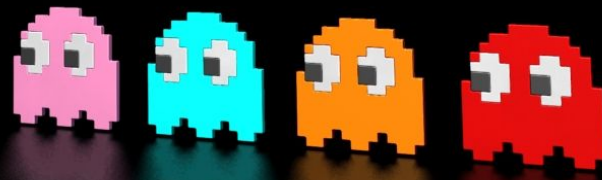
```
scene.setOnKeyReleased(e -> {  
    if (e.getCode() == KeyCode.LEFT) {  
        this.pulsados[0] = false;  
    }  
    if (e.getCode() == KeyCode.RIGHT) {  
        this.pulsados[1] = false;  
    }  
});
```



# PELOTA

Al igual que la barra, tenemos que inicializar la pelota en el campo y dándole sus propiedades también como la X, Y, y Radio.

```
this.pelota = new Pelota(  
    new Point2D(this.getBarra().getPosicion().getX()  
+ this.getBarra().getAncho() / 2,  
    this.getBarra().getPosicion().getY() - 15), 10);
```



# PELOTA

Si la X de la Pelota es menor que el borde del campo, la dirección de la barra cambia en 180 grados, es decir, rebota. Y lo mismo pasa con la zona de la derecha, si la X de la pelota más su radio, es mayor que el Ancho del Campo menos el borde, la pelota rebotará.

```
// rebote izquierda del borde
if (this.campo.getPelota().getPosicion().getX() <= this.campo.borde) {
    this.campo.getPelota().setAngulo(180 - this.campo.getPelota().getAngulo());
}

// rebote derecha del borde
if (this.campo.getPelota().getPosicion().getX() + this.campo.getPelota().getRadio() >= Campo.getWidth() - this.campo.getBorde()) {
    this.campo.getPelota().setAngulo(180 - this.campo.getPelota().getAngulo());
}
```



# PELOTA

Al igual que en las paredes, necesitamos hacer que la pelota rebote arriba del campo. En este caso debemos mirar la Y de la pelota y compararla con el campo del borde.

Tendremos que mirar que la Y sea menor o igual para realizar en este caso el cambio de ángulo de la pelota, que sería 360 menos el ángulo actual de la pelota.

```
// rebote arriba del borde
if (this.campo.getPelota().getPosicion().getY() <= this.campo.getBorde()) {
    this.campo.getPelota().setAngulo(360 - this.campo.getPelota().getAngulo());
}
```





# PELOTA

Aquí delimitamos que la pelota solo rebote cuando da a la barra, debido a que si lo quitamos, la colisión se realizaría en el aire y no con ella.

```
// rebote de la pelota con la barra parte de la izquierda
if (this.campo.getPelota().getPosicion().getY() + this.campo.getPelota().getRadio() >= this.campo.getBarra().getPosicion().getY()) {
    if (this.campo.getPelota().getPosicion().getX() + this.campo.getPelota().getRadio() > this.campo.getBarra().getPosicion().getX()) {
        this.campo.getPelota().setAngulo(360 - this.campo.getPelota().getAngulo());
    }
}

// rebote de la pelota con la barra parte de la derecha
if (this.campo.getPelota().getPosicion().getY() + this.campo.getPelota().getRadio() >= this.campo.getBarra().getPosicion().getY()) {
    if (this.campo.getBarra().getPosicion().getX() + this.campo.getBarra().getAncho()
        <= this.campo.getPelota().getPosicion().getX() + this.campo.getPelota().getRadio()) {
        this.campo.getPelota().setAngulo(360 - this.campo.getPelota().getAngulo());
    }
}
```





# PELOTA

Aquí delimitamos que la pelota solo rebote cuando da a la barra, debido a que si lo quitamos, la colisión se realizaría en el aire y no con ella.

```
// rebote de la pelota con la barra parte de la izquierda
if (this.campo.getPelota().getPosicion().getY() + this.campo.getPelota().getRadio() >= this.campo.getBarra().getPosicion().getY()) {
    if (this.campo.getPelota().getPosicion().getX() + this.campo.getPelota().getRadio() > this.campo.getBarra().getPosicion().getX()) {
        this.campo.getPelota().setAngulo(360 - this.campo.getPelota().getAngulo());
    }
}

// rebote de la pelota con la barra parte de la derecha
if (this.campo.getPelota().getPosicion().getY() + this.campo.getPelota().getRadio() >= this.campo.getBarra().getPosicion().getY()) {
    if (this.campo.getBarra().getPosicion().getX() + this.campo.getBarra().getAncho()
        <= this.campo.getPelota().getPosicion().getX() + this.campo.getPelota().getRadio()) {
        this.campo.getPelota().setAngulo(360 - this.campo.getPelota().getAngulo());
    }
}
```



# LADRILLO

Cada ladrillo tiene su posición, realizada con el Point2D y una dureza que determina los golpes que necesita un Ladrillo para romperse.

```
public class Ladrillo {  
  
    public static int ancho = 32;  
    public static int alto = 16;  
    private int dureza;  
    private boolean fijo;  
    private Point2D posicion;  
  
    public Ladrillo(){  
        this.posicion= new Point2D(0,0);  
        this.dureza = 2;  
        this.fijo = false;  
    }  
  
    public Ladrillo(Point2D posicion, int dureza){  
        this.posicion = posicion;  
        this.dureza = dureza;  
    }  
}
```



# LADRILLO

Aquí están las colisiones de la pelota con los ladrillos, según el ángulo en el que la pelota le de al ladrillo, este rebotará de una manera u otra...

```
for (int i = 0; i < this.nivelactual.getLadrillos().length; i++) {  
    for (int j = 0; j < this.nivelactual.getLadrillos()[i].length; j++) {  
        Ladrillo l;  
        l = this.nivelactual.getLadrillos()[i][j];  
        int lx = (int) l.getPosicion().getX();  
        int ly = (int) l.getPosicion().getY();  
        int ancho = (int) l.getAncho();  
        int alto = (int) l.getAlto();  
  
        if (pAngulo > 180) {  
            if (py == ly + alto && px + pr > lx && px < lx + ancho && l.getDureza() > 0) {  
                tipoangulo = 0;  
                l.setDureza(l.getDureza() - 1);  
                tocar = true;  
            }  
        } else if (pAngulo < 180) {  
            if (py + pr == ly && px + pr > lx && px < lx + ancho && l.getDureza() > 0) {  
                tipoangulo = 1;  
                l.setDureza(l.getDureza() - 1);  
                tocar = true;  
            }  
        } else if (pAngulo > 270) {  
            if (px + pr == lx && py < ly + alto && py + pr > ly && l.getDureza() > 0) {  
                tipoangulo = 2;  
                l.setDureza(l.getDureza() - 1);  
                tocar = true;  
            }  
        } else {  
            if (px == lx + ancho && py < ly + alto && py + pr > ly && l.getDureza() > 0) {  
                tipoangulo = 3;  
                l.setDureza(l.getDureza() - 1);  
                tocar = true;  
            }  
        }  
    }  
}
```

1. Primer caso: Que la pelota rebote por debajo del ladrillo.
2. Segundo caso: Que la pelota rebote por arriba.
3. Tercer caso: Que la pelota rebote por la parte izquierda del ladrillo.
4. Cuarto caso: Que la pelota rebote por la parte derecha del ladrillo.



# LADRILLO

Aquí están las colisiones de la pelota con los ladrillos, según el ángulo en el que la pelota le de al ladrillo, este rebotará de una manera u otra...

```
for (int i = 0; i < this.nivelactual.getLadrillos().length; i++) {  
    for (int j = 0; j < this.nivelactual.getLadrillos()[i].length; j++) {  
        Ladrillo l;  
        l = this.nivelactual.getLadrillos()[i][j];  
        int lx = (int) l.getPosicion().getX();  
        int ly = (int) l.getPosicion().getY();  
        int ancho = (int) l.getAncho();  
        int alto = (int) l.getAlto();  
  
        if (pAngulo > 180) {  
            if (py == ly + alto && px + pr > lx && px < lx + ancho && l.getDureza() > 0) {  
                tipoangulo = 0;  
                l.setDureza(l.getDureza() - 1);  
                tocar = true;  
            }  
        } else if (pAngulo < 180) {  
            if (py + pr == ly && px + pr > lx && px < lx + ancho && l.getDureza() > 0) {  
                tipoangulo = 1;  
                l.setDureza(l.getDureza() - 1);  
                tocar = true;  
            }  
        } else if (pAngulo > 270) {  
            if (px + pr == lx && py < ly + alto && py + pr > ly && l.getDureza() > 0) {  
                tipoangulo = 2;  
                l.setDureza(l.getDureza() - 1);  
                tocar = true;  
            }  
        } else {  
            if (px == lx + ancho && py < ly + alto && py + pr > ly && l.getDureza() > 0) {  
                tipoangulo = 3;  
                l.setDureza(l.getDureza() - 1);  
                tocar = true;  
            }  
        }  
    }  
}
```

1. Primer caso: Que la pelota rebote por debajo del ladrillo.
2. Segundo caso: Que la pelota rebote por arriba.
3. Tercer caso: Que la pelota rebote por la parte izquierda del ladrillo.
4. Cuarto caso: Que la pelota rebote por la parte derecha del ladrillo.



# LADRILLO

Clase LadrilloUI. Simplemente esta clase nos sirve para poder definir una imagen al ladrillo, darle las dimensiones a la imagen, su color según la dureza...

```
public class LadrilloUI {  
  
    private Ladrillo ladrillo[];  
    private int estado;  
    private Campo campo;  
    private static int NUMESTADOS = 6;  
    private Point2D dibujoladrillo[];  
    private Image imagen;  
    private static int colorLadrillos[][] = {  
        { //blanco  
            (0, 0)  
        },  
        { //naranja  
            (16, 0)  
        },  
        { //celeste  
            (32, 0)  
        },  
        { //verde  
            (48, 0)  
        },  
        { //rojo  
            (0, 8)  
        },  
        { //azul  
            (16, 8)  
        },  
        { //violeta  
            (32, 8)  
        },  
        { //amarillo  
            (48, 8)  
        },  
        { //ris  
            (0, 16)  
        },  
        { //dorado  
            (0, 24)  
        },  
    };  
}
```

```
public LadrilloUI(Image imagen, Ladrillo ladrillo[]) {  
  
    this.mapear();  
    this.imagen = imagen;  
    this.ladrillo = ladrillo;  
  
}  
  
private void mapear() {  
    this.dibujoladrillo = new Point2D[LadrilloUI.NUMESTADOS];  
    for (int i = 0; i < this.dibujoladrillo.length; i++) {  
        this.dibujoladrillo[i] = new Point2D(31, i * 8);  
    }  
}  
  
public void inc() {  
    this.estado++;  
    if (this.estado >= LadrilloUI.NUMESTADOS) {  
        this.estado = 0;  
    }  
}
```



# NIVELES

Esta sería la clase Nivel, que contendrá las propiedades del nivel que creamos. Por ejemplo, en esta clase almacenamos un vector de ladrillos, el inicio donde aparecerán los ladrillos y el fondo que se pintará.

```
public class Nivel {

    private Ladrillo[][] ladrillo;
    int nivel;
    private Point2D inicio;
    private Point2D fondo;

    public Nivel() {

    }

    public Nivel(Point2D inicio, int[][] matriz) {
        this.inicio = inicio;
        this.init(matriz);
    }

    public void init(int matriz[][]) {
        Point2D tempo;
        int inicioX = 0;
        int inicioY = 0;

        if (this.getInicio() != null) {
            inicioX = (int) this.getInicio().getX();
            inicioY = (int) this.getInicio().getY();
        }

        this.ladrillo = new Ladrillo[matriz.length][matriz[0].length];
        for (int i = 0; i < matriz.length; i++) {
            for (int j = 0; j < matriz[i].length; j++) {
                tempo = new Point2D(inicioX + j * Ladrillo.ancha, inicioY + i * Ladrillo.alto);
                this.ladrillo[i][j] = new Ladrillo(tempo, matriz[i][j]);
            }
        }
    }
}
```

```
public Ladrillo[][] getLadrillos() {
    return this.ladrillo;
}

public boolean haTerminado() {
    boolean terminado = true;

    for (int i = 0; i < this.getLadrillos().length; i++) {
        for (int j = 0; j < this.getLadrillos()[i].length; j++) {
            if (this.getLadrillos()[i][j].getDureza() > 0) {
                terminado = false;
            }
        }
    }

    return terminado;
}

/**
 * @return the fondo
 */
public Point2D getFondo() {
    return fondo;
}

/**
 * @param fondo the fondo to set
 */
public void setFondo(Point2D fondo) {
    this.fondo = fondo;
}

/**
 * @return the inicio
 */
public Point2D getInicio() {
    return inicio;
}

/**
 * @param inicio the inicio to set
 */
public void setInicio(Point2D inicio) {
    this.inicio = inicio;
}
}
```



# NIVELES

Para ir inicializando los niveles, tenemos estos atributos en la clase Juego. Almacenamos las vidas, una matriz con niveles, el objeto que es Nivel actual y un contador para llevar la cuenta y ir sumando en el cada vez que un nivel termine.

```
public class Juego {  
  
    public int vidas = 3;  
    public Nivel niveles[];  
    public Nivel nivelactual;  
    public int nivel = 0; // nivel actual
```

```
public void init() {  
    this.nivel = 0;  
    niveles = new Nivel[4];  
  
    Point2D inicio = new Point2D(110, 50);  
  
    int[][] nivel1 = {  
        {1,0,1,0,1,0,1},  
    };  
  
    int[][] nivel2 = {  
        {2,0,1,0,1,0,2},  
        {0,2,0,1,0,2,0},  
    };  
  
    int[][] nivel3 = {  
        {2,0,1,0,1,0,2},  
        {0,2,0,1,0,2,0},  
        {1,0,1,0,1,0,1},  
    };  
  
    niveles[0] = new Nivel(inicio, nivel1);  
    niveles[0].setFondo(new Point2D(0, 257));  
  
    niveles[1] = new Nivel(inicio, nivel2);  
    niveles[1].setFondo(new Point2D(929, 0));  
  
    niveles[2] = new Nivel(inicio, nivel3);  
    niveles[2].setFondo(new Point2D(233, 257));  
  
    this.nivelactual = niveles[nivel];  
}
```

Tenemos un método llamado init en la clase Juego donde iniciamos todos los niveles con sus matrices y dentro las durezas. Aparte, usamos los métodos setFondo para establecer con ellos un fondo.





# NIVELES

En JuegoUI disponemos de un método llamado paintBackground el cual se encarga de pintar todo el fondo y los ladrillos, ya que según su dureza, los va a pintar de un color o otro.

```
public void paintBackground(GraphicsContext gc) {  
    // Pinta los ladrillos que detecte en el fondo y el fondo  
    int borde = 10;  
  
    gc.drawImage(this.imagen_fondo, this.juego.nivelactual.getFondo().getX(), this.juego.nivelactual.getFondo().getY(), 224, 240, 0, 0, 448, 480);  
    Ladrillo l;  
    if (this.juego != null && this.juego.nivelactual.getLadrillos() != null) {  
        //pintar el ladrillo  
        for (int i = 0; i < this.juego.nivelactual.getLadrillos().length; i++) {  
            ///System.out.println(this.juego.getCampo().getLadrillo());  
            for (int j = 0; j < this.juego.nivelactual.getLadrillos()[i].length; j++) {  
                l = this.juego.nivelactual.getLadrillos()[i][j];  
                if (l.getDureza() > 0 && l != null) {  
  
                    gc.drawImage(this.imagen_bloques, this.colorladrillos[l.getDureza()][0],  
                                this.colorladrillos[l.getDureza()][1],  
                                16, 8,  
                                l.getPosicion().getX(),  
                                l.getPosicion().getY(),  
                                32, 16);  
                }  
            }  
        }  
    }  
  
    if (this.juego.getVidas() > 0) {  
        for (int i = 0; i < this.juego.getVidas(); i++) {  
            gc.drawImage(this.imagen_fondo, 8/*posicionXimagen*/,  
                        240/*posicionYImagen*/, 16/*tamañoXObjeto*/,  
                        7/*tamañoYObjeto*/,  
                        20 + (50 * i)/*posicionXDondeSePinta*/,  
                        Juego.ALTO - 20/*posicionYDondeSePinta*/,  
                        32, 16/*proporcionAPintar*/);  
        }  
    }  
}
```



