VIDEOJUEGO CALDERO MODIFICADO

MARIO JIMÉNEZ MARSET

ÍNDICE

1. ENUNCIADO – OBJETIVOS

En esta práctica se pedía, a partir del código fuente facilitado, compilar y ejecutar el juego del caldero. Hecho esto y comprobado que este funciona, modificar cinco características visibles del programa. Para ello, se ha clonado el proyecto del videojuego para realizar estas modificaciones.

2. DESARROLLO – PROCEDIMIENTOS

Se muestra el código del juego con cinco modificaciones identificables a partir de los comentarios

Código Clase DesktopDrop:

```
package juego;
import com.badlogic.gdx.backends.lwjgl.LwjglApplication;
import com.badlogic.gdx.backends.lwjgl.LwjglApplicationConfiguration;

public class DesktopDrop {
	public static void main(String[] args) {
		LwjglApplicationConfiguration configuracion = new
LwjglApplicationConfiguration();
		//se ha modificado el título de la ventana
		configuracion.title = "Caldero Modificado";
		configuracion.width = 1024;
		configuracion.height = 768;
		new LwjglApplication(new Drop(), configuracion);
	}
}
```

Código Clase Drop:

```
package juego;
import com.badlogic.gdx.Game;
import com.badlogic.gdx.graphics.OrthographicCamera;
import com.badlogic.gdx.graphics.g2d.BitmapFont;
import com.badlogic.gdx.graphics.g2d.SpriteBatch;

public class Drop extends Game {
	OrthographicCamera camara;
	SpriteBatch spriteBatch;
	BitmapFont fuente;
	int gotasRecogidas;
	@Override
	public void create() {
		spriteBatch = new SpriteBatch();
		fuente = new BitmapFont();
		setScreen(new MainMenuScreen(this));
	}
	@Override
	public void render() {
		super.render();
	}
	@Override
	public void dispose() {
		spriteBatch.dispose();
```

```
                fuente.dispose();
        }
}
```

Código Clase GameOverScreen:

```
package juego;
import com.badlogic.gdx.Gdx;
import com.badlogic.gdx.Input.Keys;
import com.badlogic.gdx.Screen;
import com.badlogic.gdx.graphics.GL10;
import com.badlogic.gdx.graphics.OrthographicCamera;
import com.badlogic.gdx.scenes.scene2d.Stage;

public class GameOverScreen implements Screen {
        final Drop juego;
        Stage menu;
        OrthographicCamera camara;
        public GameOverScreen(Drop juego) {
                this.juego = juego;
                camara = new OrthographicCamera();
                camara.setToOrtho(false, 1024, 768);
        }
        @Override
        public void render(float delta) {
                //MODIFICACIÓN 1 --> cambiar el color de fondo de la pantalla final del
juego
                Gdx.gl.glClearColor(0,255,0,0.8f);

                Gdx.gl.glClear(GL10.GL_COLOR_BUFFER_BIT);
                camara.update();
                juego.spriteBatch.setProjectionMatrix(camara.combined);
                juego.spriteBatch.begin();
                juego.fuente.draw(juego.spriteBatch, "Fin del juego!!!!", 100, 150);
                juego.fuente.draw(juego.spriteBatch, "Tu puntuación: " +
juego.gotasRecogidas, 100, 130);
                juego.fuente.draw(juego.spriteBatch, "Si quieres jugar otra partida pulsa
la tecla 'C'", 100, 110);
                juego.fuente.draw(juego.spriteBatch, "Pulsa 'ESCAPE' para SALIR",
100, 90);
                juego.spriteBatch.end();
                //MODIFICACIÓN 2 --> se ha cambiado la tecla con la que empezar
una nueva partida
                if (Gdx.input.isKeyPressed(Keys.C)) {
                        juego.gotasRecogidas = 0;
                        juego.setScreen(new GameScreen(juego));
                }
                else if (Gdx.input.isKeyPressed(Keys.ESCAPE)) {
                        dispose();
                        System.exit(0);
                }

        }
        @Override
        public void resize(int width, int height) {
        }
        @Override
        public void show() {
```

```
		}

		@Override
		public void hide() {
		}
		@Override
		public void pause() {
		}
		@Override
		public void resume() {
		}
		@Override
		public void dispose() {
			juego.dispose();
		}
}
```

Código Clase GameScreen:

```
package juego;
import java.util.Iterator;
import com.badlogic.gdx.Gdx;
import com.badlogic.gdx.InputProcessor;
import com.badlogic.gdx.Screen;
import com.badlogic.gdx.Input.Keys;
import com.badlogic.gdx.audio.Music;
import com.badlogic.gdx.audio.Sound;
import com.badlogic.gdx.graphics.GL10;
import com.badlogic.gdx.graphics.OrthographicCamera;
import com.badlogic.gdx.graphics.Texture;
import com.badlogic.gdx.math.MathUtils;
import com.badlogic.gdx.math.Rectangle;
import com.badlogic.gdx.math.Vector3;
import com.badlogic.gdx.utils.Array;
import com.badlogic.gdx.utils.TimeUtils;
import com.badlogic.gdx.utils.Timer;
import com.badlogic.gdx.utils.Timer.Task;

public class GameScreen implements Screen, InputProcessor {
	final Drop juego;
	Texture spriteGota;
	Texture spriteCubo;
	Texture spriteRayo;
	Sound sonidoGota;
	Music musicaLluvia;
	Sound sonidoRayo;
	Rectangle cubo;
	Array<Rectangle> gotas;
	Array<Rectangle> rocas;
	long momentoUltimaGota;
	long momentoUltimaRoca;
	float tiempoJuego;
	boolean pausa = false;
	OrthographicCamera camara;
	public GameScreen(Drop juego) {
		this.juego = juego;
		tiempoJuego = 50;
```

```
                //MODIFICACIÓN 3 --> se cambia las rocas por rayos
                spriteGota = new Texture(Gdx.files.internal("droplet.png"));
                spriteCubo = new Texture(Gdx.files.internal("bucket.png"));
                spriteRayo = new Texture(Gdx.files.internal("rayo.png"));
                //MODIFICACIÓN 4 --> se cambia el sonido de la roca por el del rayo
                sonidoGota =
Gdx.audio.newSound(Gdx.files.internal("waterdrop.wav"));
                musicaLluvia =
Gdx.audio.newMusic(Gdx.files.internal("undertreeinrain.mp3"));
                sonidoRayo = Gdx.audio.newSound(Gdx.files.internal("rayo.mp3"));
                musicaLluvia.setLooping(true);
                cubo = new Rectangle();
                cubo.x = 1024 / 2 - 64 / 2;
                cubo.y = 20;
                cubo.width = 64;
                cubo.height = 64;
                gotas = new Array<Rectangle>();
                generarLluvia();
                rocas = new Array<Rectangle>();
                lanzarRoca();
                camara = new OrthographicCamera();
                camara.setToOrtho(false, 1024, 768);
                Gdx.input.setInputProcessor(this);
        }
        @Override
        public void render(float delta) {
                //MODIFICACIÓN 5 --> se cambia el color del fondo de pantalla
                Gdx.gl.glClearColor(255, 125, 0, 0.7f);
                Gdx.gl.glClear(GL10.GL_COLOR_BUFFER_BIT);
                camara.update();
                if (!pausa) {
                        comprobarInput();
                        actualizar();
                }
                dibujar();
        }
        private void comprobarInput() {
                if (Gdx.input.isTouched()) {
                        Vector3 posicion = new Vector3();
                        posicion.set(Gdx.input.getX(), Gdx.input.getY(), 0);
                        cubo.x = posicion.x - 64 /2;
                }
                if (Gdx.input.isKeyPressed(Keys.LEFT))
                        cubo.x -= 200 * Gdx.graphics.getDeltaTime();
                if (Gdx.input.isKeyPressed(Keys.RIGHT))
                        cubo.x += 200 * Gdx.graphics.getDeltaTime();
        }
        private void actualizar() {
                if (cubo.x < 0)
                        cubo.x = 0;
                if (cubo.x > 1024 - 64)
                        cubo.x = 1024 - 64;
                if (TimeUtils.nanoTime() - momentoUltimaGota > 100000000)
                        generarLluvia();
                if (TimeUtils.nanoTime() - momentoUltimaRoca > 1000000000)
                        lanzarRoca();
                Iterator<Rectangle> iter = gotas.iterator();
                while (iter.hasNext()) {
```

```
                        Rectangle gota = iter.next();
                        gota.y -= 200 * Gdx.graphics.getDeltaTime();
                        if (gota.y + 64 < 0)
                                iter.remove();
                        if (gota.overlaps(cubo)) {
                                sonidoGota.play();
                                iter.remove();
                                juego.gotasRecogidas++;
                        }
                }
                Iterator<Rectangle> iterRoca = rocas.iterator();
                while (iterRoca.hasNext()) {
                        Rectangle roca = iterRoca.next();
                        roca.y -= 200 * Gdx.graphics.getDeltaTime();
                        if (roca.y + 64 < 0)
                                iterRoca.remove();
                        if (roca.overlaps(cubo)) {
                                sonidoRayo.play();
                                pausa = true;
                                Timer.schedule(new Task(){
                                   @Override
                                   public void run() {
                                      dispose();
                                                juego.setScreen(new
GameOverScreen(juego));
                                   }
                                }, 2);
                        }
                }
                tiempoJuego -= Gdx.graphics.getDeltaTime();
                if (tiempoJuego < 0) {
                        dispose();
                        juego.setScreen(new GameOverScreen(juego));
                }
        }
        private void dibujar() {
                juego.spriteBatch.begin();
                juego.spriteBatch.draw(spriteCubo, cubo.x, cubo.y);
                for (Rectangle gota : gotas)
                        juego.spriteBatch.draw(spriteGota, gota.x, gota.y);
                for (Rectangle roca : rocas)
                        juego.spriteBatch.draw(spriteRayo, roca.x, roca.y);
                juego.fuente.draw(juego.spriteBatch, "Puntos: " +
juego.gotasRecogidas, 1024 - 100, 768 - 50);
                juego.fuente.draw(juego.spriteBatch, "Tiempo: " + (int) (tiempoJuego),
1024 - 100, 768 - 80);
                juego.spriteBatch.end();
        }
        private void generarLluvia() {
                Rectangle gota = new Rectangle();
                gota.x = MathUtils.random(0, 1024 - 64);
                gota.y = 768;
                gota.width = 64;
                gota.height = 64;
                gotas.add(gota);
                momentoUltimaGota = TimeUtils.nanoTime();
        }
        private void lanzarRoca() {
```

```
            Rectangle roca = new Rectangle();
            roca.x = MathUtils.random(0, 1024 - 64);
            roca.y = 768;
            roca.width = 64;
            roca.height = 64;
            rocas.add(roca);
            momentoUltimaRoca = TimeUtils.nanoTime();
    }
    @Override
    public void show() {
            musicaLluvia.play();
    }
    @Override
    public void hide() {
            musicaLluvia.stop();
    }
    @Override
    public void dispose() {
            spriteGota.dispose();
            spriteCubo.dispose();
            spriteRayo.dispose();
            sonidoGota.dispose();
            musicaLluvia.dispose();
            sonidoRayo.dispose();

            gotas.clear();
            rocas.clear();
    }
    @Override
    public void resize(int width, int height) {
    }
    @Override
    public void pause() {
            pausa = true;
    }
    @Override
    public void resume() {
            pausa = false;
    }
    @Override
    public boolean keyDown(int keycode) {
            return false;
    }
    @Override
    public boolean keyUp(int keycode) {
            if (keycode == Keys.P)
                    pausa = !pausa;
            return false;
    }
    @Override
    public boolean keyTyped(char character) {
            return false;
    }
    @Override
    public boolean touchDown(int screenX, int screenY, int pointer, int button) {
            return false;
    }
    @Override
```

```
		public boolean touchUp(int screenX, int screenY, int pointer, int button) {
			return false;
		}
		@Override
		public boolean touchDragged(int screenX, int screenY, int pointer) {
			return false;
		}
		@Override
		public boolean mouseMoved(int screenX, int screenY) {
			return false;
		}
		@Override
		public boolean scrolled(int amount) {
			return false;
		}
}
```

Código MainMenuScreen:

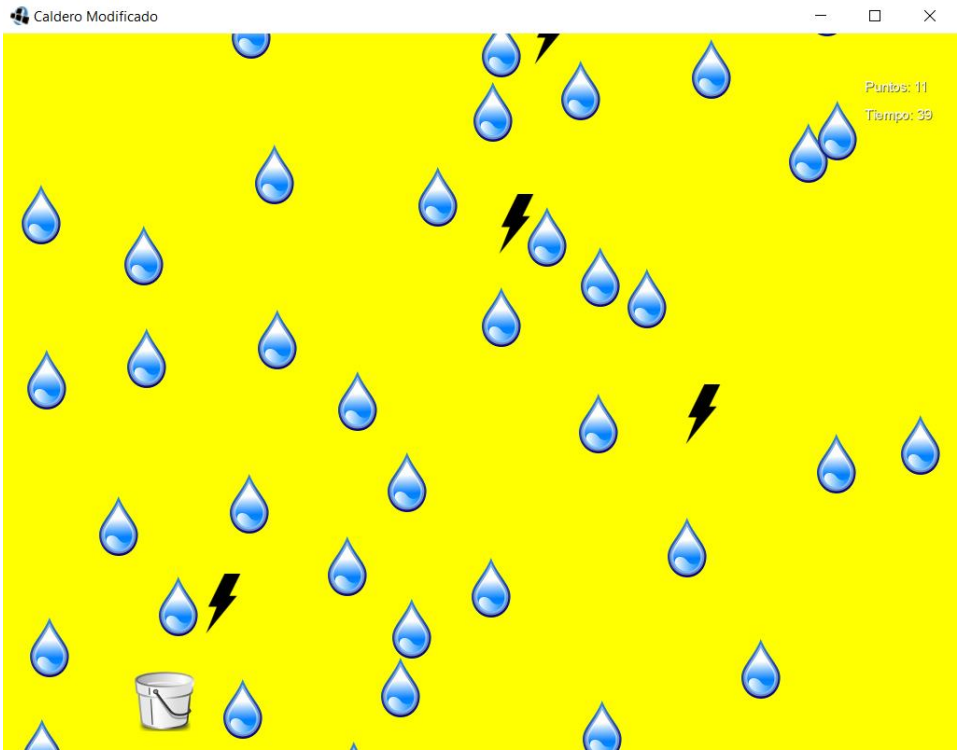
```
package juego;
import com.badlogic.gdx.Gdx;
import com.badlogic.gdx.Input.Keys;
import com.badlogic.gdx.Screen;
import com.badlogic.gdx.graphics.GL10;
import com.badlogic.gdx.graphics.OrthographicCamera;

public class MainMenuScreen implements Screen {
	final Drop juego;
	OrthographicCamera camara;
	public MainMenuScreen(Drop juego) {
		this.juego = juego;
		camara = new OrthographicCamera();
		camara.setToOrtho(false, 1024, 768);
	}
	@Override
	public void render(float delta) {
		Gdx.gl.glClearColor(0, 0, 0.2f, 1);
		Gdx.gl.glClear(GL10.GL_COLOR_BUFFER_BIT);
		camara.update();
		juego.spriteBatch.setProjectionMatrix(camara.combined);
		juego.spriteBatch.begin();
		juego.fuente.draw(juego.spriteBatch, "Bienvenido a Drop!!!!", 100, 150);
		juego.fuente.draw(juego.spriteBatch, "Pulsa para empezar", 100, 130);
		juego.fuente.draw(juego.spriteBatch, "Pulsa 'ESCAPE' para SALIR",
100, 110);
		juego.spriteBatch.end();
		if (Gdx.input.isTouched()) {
			juego.setScreen(new GameScreen(juego));
			dispose();
		}
		if (Gdx.input.isKeyPressed(Keys.ESCAPE)) {
			dispose();
			System.exit(0);
		}
	}
	@Override
	public void resize(int width, int height) {
	}
```

```
        @Override
        public void show() {
        }
        @Override
        public void hide() {
        }
        @Override
        public void pause() {
        }
        @Override
        public void resume() {
        }
        @Override
        public void dispose() {
        }
}
```

CAPTURAS RESULTADOS:

Se muestra la primera modificación (se ha cambiado el color de fondo de pantalla por el amarillo). También se muestra que se han cambiado las rocas por rayos. Además, a pesar de no ser visible por capturas, al tocar el cubo los rayos suena el sonido de un rayo, en vez del de una roca.



Se muestra además cómo se ha cambiado el color del menú final del juego a verde, además de la tecla N por la C, con la que iniciar otra partida. Esto hace un total de cinco modificaciones en el juego del caldero.