Informe de Laboratorio 24

Tema: Proyecto final-lab24

Nota

Estudiante	Escuela	Asignatura
Ryan Fabian Valdivia Segovia, Dylan Antonio Zuñiga Huarca rvaldiviase@unsa.edu.pe, dzunigahu@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Fundamentos de la programación 2 Semestre: II Código: 1701213

Laboratorio	Tema	Duración
24	Proyecto final-lab24	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - B	Del 24 de Enero 2024	Al 29 de Enero 2024

1. Tarea

- Cree una versión del videojuego de estrategia usando componentes básicos GUI: Etiquetas, botones, cuadros de texto, JOptionPane, Color.
- Además, utilizar componentes avanzados GUI: Layouts, JPanel, áreas de texto, checkbox, botones de radio y combobox.
- Considerar nivel estratégico y táctico.
- Considerar hasta las unidades especiales de los reinos.
- Hacerlo iterativo.

2. Equipos, materiales y temas utilizados

- Sistema Operativo Windows 11 Home Single Language 64 bits 22621.2283
- VIM 9.0.
- Visual Studio Code 64 bits 1.82.2
- OpenJDK 64-Bits 21.0.2
- Git 2.41.0.windows.1
- IntelliJ IDEA 2023.3 Runtime version: 17.0.9+7-b1087.7 amd64
- Cuenta en GitHub con el correo institucional.

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/RyanValdivia/fp2-23b`
- URL para el laboratorio 24 en el Repositorio GitHub.
- `https://github.com/RyanValdivia/fp2-23b/tree/main/fase03/lab22`

4. URL del video de exposición

- `https://www.youtube.com/watch?v=9DbYmxoijf4`

5. Actividades

- Este laboratorio consistía en elaborar una interfaz gráfica de usuario, portando todo nuestro videojuego de laboratorios anteriores.
- Para esto, reutilicé el código de laboratorios anteriores, usando la misma lógica. Entonces reciclé las clases de laboratorios anteriores

Listing 1: Superclase Soldado

```
   package com.example.lab22;
import java.util.*;

public class Soldier {
    protected String name;
    protected String alias;
    protected int HP;
    protected int attack;
    protected int defense;
    protected int x;
    protected int y;
    protected boolean status;
    private String color;
    private int armyId;

    public Soldier(String name){
        this.name = name;
        this.status = true;
    }
    public Soldier(){
        this.name = "";
        this.alias = "";
        this.color = "#cdcdcd";
        this.status = false;
    }
   public void action(){};
    public static Soldier winner(Soldier s1, Soldier s2){
        Random random = new Random();
        double h1 = s1.getHP();
        double h2 = s2.getHP();
        double total = h1 + h2;
```

```
        h1 /= total;
        h2 /= total;
        double ans = random.nextDouble();
        if(h2 < h1){
            if(ans <= h2){
                return s2;
            }else{
                return s1;
            }
        }else{
            if(ans <= h1){
                return s1;
            }else{
                return s2;
            }
        }

    }
    @Override
    public String toString(){
        return name + ", " + alias + ", " + HP + ", " + attack + ", " + defense + ", " +
            armyId + ", " + x + ", " + y + ", " + status + ", " + color;
    }
    public static Soldier parseSoldier(String data){
        String[] parts = data.split(", ");
        String name = parts[0];
        String alias = parts[1];
        int hp = Integer.parseInt(parts[2]);
        int attack = Integer.parseInt(parts[3]);
        int defense = Integer.parseInt(parts[4]);
        int armyId = Integer.parseInt(parts[5]);
        int x = Integer.parseInt(parts[6]);
        int y = Integer.parseInt(parts[7]);
        boolean status = Boolean.parseBoolean(parts[8]);
        String color = parts[9];

        Soldier soldier = new Soldier(name);
        soldier.setX(x);
        soldier.setY(y);
        soldier.setAlias(alias);
        soldier.setHP(hp);
        soldier.setAttack(attack);
        soldier.setDefense(defense);
        soldier.setColor(color);
        soldier.setArmyId(armyId);
        soldier.setStatus(status);

        return soldier;

    }
    public void destroy(){
        this.name = "";
        this.alias = "";
        this.HP = 0;
        this.attack = 0;
        this.defense = 0;
```

```
        this.status = false;
        this.color = "#cdcdcd";
        this.armyId = 0;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getAlias() {
        return alias;
    }

    public void setAlias(String alias) {
        this.alias = alias;
    }

    public int getHP() {
        return HP;
    }

    public void setHP(int HP) {
        this.HP = HP;
    }

    public int getAttack() {
        return attack;
    }

    public void setAttack(int attack) {
        this.attack = attack;
    }

    public int getDefense() {
        return defense;
    }

    public void setDefense(int defense) {
        this.defense = defense;
    }

    public int getX() {
        return x;
    }

    public String getColor() {
        return color;
    }

    public void setColor(String color) {
        this.color = color;
    }
```

```
    public int getArmyId() {
        return armyId;
    }

    public void setArmyId(int armyId) {
        this.armyId = armyId;
    }

    public void setX(int x) {
        this.x = x;
    }

    public int getY() {
        return y;
    }

    public void setY(int y) {
        this.y = y;
    }

    public boolean getStatus() {
        return status;
    }

    public void setStatus(boolean status) {
        this.status = status;
    }
}
class Archer extends Soldier{
    private int arrows;
    Random random = new Random();

    public Archer(String name) {
        super(name);
        this.arrows = random.nextInt(10);
        this.setAttack(7);
        this.setDefense(3);
        this.setHP(random.nextInt(3) + 3);
    }

    public int getArrows() {
        return arrows;
    }

    public void setArrows(int arrows) {
        this.arrows = arrows;
    }

    //Shoots one arrow to an enemy
    @Override
    public void action(){
        this.arrows--;
    }
}
class Knight extends Soldier{
```

```
    private String weapon;
    private boolean isMounted;
    Random random = new Random();

    public Knight(String name) {
        super(name);
        this.setAttack(13);
        this.setDefense(7);
        this.setHP(random.nextInt(3) + 10);
    }

    public String getWeapon() {
        return weapon;
    }

    public void setWeapon(String weapon) {
        this.weapon = weapon;
    }

    public boolean isMounted() {
        return isMounted;
    }

    public void setMounted(boolean mounted) {
        isMounted = mounted;
    }

    //If it's unmounted, it will mount and its weapon will change to a Sword, if not,
    // it will unmount and its weapon will change to a Spear
    @Override
    public void action(){
        this.isMounted = !this.isMounted;

        if(this.isMounted){
            this.weapon = "Sword";
        }else{
            this.weapon = "Spear";
        }
    }
}
class Swordsman extends Soldier{
    private final int swordLong;
    Random random = new Random();

    public Swordsman(String name) {
        super(name);
        this.swordLong = random.nextInt(10);
        this.setAttack(10);
        this.setDefense(8);
        this.setHP(random.nextInt(3) + 8);
    }

    public int getSwordLong() {
        return swordLong;
    }
```

```
    //This will set a wall of shields, and increase its defense by 1
    @Override
    public void action(){
        this.setDefense(this.getDefense() + 1);
    }
}
class Spearman extends Soldier{
    private final int spearLong;
    Random random = new Random();

    public Spearman(String name) {
        super(name);
        this.spearLong = random.nextInt(10);
        this.setAttack(5);
        this.setDefense(8);
        this.setHP(random.nextInt(3) + 8);
    }

    public int getSpearLong() {
        return spearLong;
    }

    //This will make a schiltrom and will increase its defense by 1
    @Override
    public void action(){
        this.setDefense(this.getDefense() + 1);
    }
}
```

Listing 2: Clase Board

```
284 package com.example.lab22;
285 import java.util.*;
286
287 public class Board {
288     private Soldier[][] table = new Soldier[10][10];
289
290     private String[] territories = new String[] {
291             "Bosque", "Campo Abierto", "Montana", "Desierto", "Playa" };
292     private String territory;
293     private Map<String, String> perks = new HashMap<>();
294     Random random = new Random();
295     public Board() {
296         this.territory = territories[random.nextInt(territories.length)];
297     }
298     public void initializeTable(){
299         for(int i = 0; i < table.length; i++){
300             for(int j = 0; j < table[i].length; j++){
301                 table[i][j] = new Soldier();
302             }
303         }
304     }
305     public void initializeArmy(Army a){
306         for(Soldier s: a.getSoldiers()){
307             int x = 0;
308             int y = 0;
```

```
309             do{
310                 x = random.nextInt(9);
311                 y = random.nextInt(9);
312             }while(table[y][x].getStatus());
313             s.setY(y);
314             s.setX(x);
315             table[y][x] = s;
316         }
317     }
318     public void deployArmy(Army a){
319         for(Soldier s: a.getSoldiers()){
320             int x = s.getX();
321             int y = s.getY();
322             table[y][x] = s;
323         }
324     }
325 
326     public Soldier[][] getTable() {
327         return table;
328     }
329 
330     public void setTable(Soldier[][] table) {
331         this.table = table;
332     }
333 
334     public String[] getTerritories() {
335         return territories;
336     }
337 
338     public void setTerritories(String[] territories) {
339         this.territories = territories;
340     }
341 
342     public String getTerritory() {
343         return territory;
344     }
345 
346     public void setTerritory(String territory) {
347         this.territory = territory;
348     }
349 
350     public Map<String, String> getPerks() {
351         return perks;
352     }
353 
354     public void setPerks(Map<String, String> perks) {
355         this.perks = perks;
356     }
357 }
```

- Esta clase implementa la interfaz gráfica en base a un fichero fxml, en el cual desarrollo lo que se va a mostrar al usuario.
- Para esta interfaz, decidí implementar 100 botones que representen un tablero de 10x10 usando el contenedor GridPane, para poder llevar una cuenta de las coordenadas de cada botón, además de una barra arriba del tablero para cambiar de turno, minimizar, cerrar o maximizar.

Listing 3: Archivo FXML

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<VBox fx:id="rootVBox" prefHeight="400.0" prefWidth="666.0" spacing="20.0" styleClass="vbox"
    stylesheets="@css/game.css" xmlns="http://javafx.com/javafx/17.0.2-ea"
    xmlns:fx="http://javafx.com/fxml/1"
    fx:controller="com.example.lab22.VideoGameController">
   <padding>
      <Insets bottom="20.0" left="20.0" right="20.0" top="20.0" />
   </padding>
  <children>
    <Pane prefHeight="200.0" prefWidth="200.0" styleClass="toolBar">
       <children>
          <Pane fx:id="closeButton" layoutX="580.0" layoutY="-12.0"
              onMouseClicked="#handleCloseButton" />
           <Pane fx:id="minimizeButton" layoutX="520.0" layoutY="-12.0"
               onMouseClicked="#handleMinimizeButton" />
           <Pane fx:id="expandButton" layoutX="550.0" layoutY="-12.0"
               onMouseClicked="#handleExpandButton" />
           <Pane fx:id="nextButton" layoutX="15.0" layoutY="-12.0"
               onMouseClicked="#changeTurn" />
           <Pane fx:id="saveButton" layoutX="45.0" layoutY="-12.0"
               onMouseClicked="#saveGame" />
           <Pane fx:id="loadButton" layoutX="75.0" layoutY="-12.0"
               onMouseClicked="#loadGame" />
           <Pane fx:id="infoButton" layoutX="105.0" layoutY="-12.0"
               onMouseClicked="#showRulesDialog" />
       </children>
    </Pane>


    <GridPane id="board" fx:id="gridPane" minWidth="0.0" prefHeight="330.0"
        prefWidth="352.0">
       <columnConstraints>
           <ColumnConstraints percentWidth="10" />
           <ColumnConstraints percentWidth="10" />
           <ColumnConstraints percentWidth="10" />
           <ColumnConstraints percentWidth="10" />
           <ColumnConstraints percentWidth="10" />
           <ColumnConstraints percentWidth="10" />
           <ColumnConstraints percentWidth="10" />
           <ColumnConstraints percentWidth="10" />
           <ColumnConstraints percentWidth="10" />
           <ColumnConstraints percentWidth="10" />
       </columnConstraints>
       <rowConstraints>
           <RowConstraints percentHeight="10" vgrow="SOMETIMES" />
           <RowConstraints percentHeight="10" vgrow="SOMETIMES" />
           <RowConstraints percentHeight="10" vgrow="SOMETIMES" />
           <RowConstraints percentHeight="10" vgrow="SOMETIMES" />
           <RowConstraints percentHeight="10" vgrow="SOMETIMES" />
           <RowConstraints percentHeight="10" vgrow="SOMETIMES" />
```

```
            <RowConstraints percentHeight="10" vgrow="SOMETIMES" />
            <RowConstraints percentHeight="10" vgrow="SOMETIMES" />
            <RowConstraints percentHeight="10" vgrow="SOMETIMES" />
            <RowConstraints percentHeight="10" vgrow="SOMETIMES" />
        </rowConstraints>
        <children>

            <!-- Row 0 -->
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="0" GridPane.rowIndex="0" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="1" GridPane.rowIndex="0" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="2" GridPane.rowIndex="0" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="3" GridPane.rowIndex="0" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="4" GridPane.rowIndex="0" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="5" GridPane.rowIndex="0" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="6" GridPane.rowIndex="0" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="7" GridPane.rowIndex="0" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="8" GridPane.rowIndex="0" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="9" GridPane.rowIndex="0" />

            <!-- Row 1 -->
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="0" GridPane.rowIndex="1" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="1" GridPane.rowIndex="1" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="2" GridPane.rowIndex="1" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="3" GridPane.rowIndex="1" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="4" GridPane.rowIndex="1" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="5" GridPane.rowIndex="1" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="6" GridPane.rowIndex="1" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="7" GridPane.rowIndex="1" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="8" GridPane.rowIndex="1" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="9" GridPane.rowIndex="1" />

            <!-- Row 2 -->
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="0" GridPane.rowIndex="2" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="1" GridPane.rowIndex="2" />
```

```
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="2" GridPane.rowIndex="2" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="3" GridPane.rowIndex="2" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="4" GridPane.rowIndex="2" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="5" GridPane.rowIndex="2" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="6" GridPane.rowIndex="2" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="7" GridPane.rowIndex="2" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="8" GridPane.rowIndex="2" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="9" GridPane.rowIndex="2" />

            <!-- Row 3 -->
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="0" GridPane.rowIndex="3" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="1" GridPane.rowIndex="3" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="2" GridPane.rowIndex="3" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="3" GridPane.rowIndex="3" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="4" GridPane.rowIndex="3" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="5" GridPane.rowIndex="3" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="6" GridPane.rowIndex="3" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="7" GridPane.rowIndex="3" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="8" GridPane.rowIndex="3" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="9" GridPane.rowIndex="3" />

            <!-- Row 4 -->
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="0" GridPane.rowIndex="4" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="1" GridPane.rowIndex="4" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="2" GridPane.rowIndex="4" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="3" GridPane.rowIndex="4" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="4" GridPane.rowIndex="4" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="5" GridPane.rowIndex="4" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="6" GridPane.rowIndex="4" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="7" GridPane.rowIndex="4" />
```

```
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="8" GridPane.rowIndex="4" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="9" GridPane.rowIndex="4" />

            <!-- Row 5 -->
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="0" GridPane.rowIndex="5" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="1" GridPane.rowIndex="5" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="2" GridPane.rowIndex="5" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="3" GridPane.rowIndex="5" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="4" GridPane.rowIndex="5" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="5" GridPane.rowIndex="5" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="6" GridPane.rowIndex="5" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="7" GridPane.rowIndex="5" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="8" GridPane.rowIndex="5" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="9" GridPane.rowIndex="5" />

            <!-- Row 6 -->
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="0" GridPane.rowIndex="6" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="1" GridPane.rowIndex="6" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="2" GridPane.rowIndex="6" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="3" GridPane.rowIndex="6" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="4" GridPane.rowIndex="6" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="5" GridPane.rowIndex="6" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="6" GridPane.rowIndex="6" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="7" GridPane.rowIndex="6" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="8" GridPane.rowIndex="6" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="9" GridPane.rowIndex="6" />

            <!-- Row 7 -->
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="0" GridPane.rowIndex="7" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="1" GridPane.rowIndex="7" />
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="2" GridPane.rowIndex="7" />
```

```
500             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="3" GridPane.rowIndex="7" />
501             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="4" GridPane.rowIndex="7" />
502             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="5" GridPane.rowIndex="7" />
503             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="6" GridPane.rowIndex="7" />
504             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="7" GridPane.rowIndex="7" />
505             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="8" GridPane.rowIndex="7" />
506             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="9" GridPane.rowIndex="7" />
507
508             <!-- Row 8 -->
509             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="0" GridPane.rowIndex="8" />
510             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="1" GridPane.rowIndex="8" />
511             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="2" GridPane.rowIndex="8" />
512             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="3" GridPane.rowIndex="8" />
513             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="4" GridPane.rowIndex="8" />
514             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="5" GridPane.rowIndex="8" />
515             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="6" GridPane.rowIndex="8" />
516             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="7" GridPane.rowIndex="8" />
517             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="8" GridPane.rowIndex="8" />
518             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="9" GridPane.rowIndex="8" />
519
520             <!-- Row 9 -->
521             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="0" GridPane.rowIndex="9" />
522             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="1" GridPane.rowIndex="9" />
523             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="2" GridPane.rowIndex="9" />
524             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="3" GridPane.rowIndex="9" />
525             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="4" GridPane.rowIndex="9" />
526             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="5" GridPane.rowIndex="9" />
527             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="6" GridPane.rowIndex="9" />
528             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="7" GridPane.rowIndex="9" />
529             <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                    styleClass="button" text="" GridPane.columnIndex="8" GridPane.rowIndex="9" />
```

```
            <Button mnemonicParsing="false" onMouseClicked="#onButtonClick"
                styleClass="button" text="" GridPane.columnIndex="9" GridPane.rowIndex="9" />


        </children>
      </GridPane>
   </children>
</VBox>
```

- Esto con el fin de asignar una función llamada onButtonClick, que implementaré mas adelante, a cada botón, para
- Por último, programé el resumen final, donde se dice cuál de los dos ejércitos es el ganador, basado en probabilidad según la cantidad de vida total por ejército. Eso es todo. Al final, solo me quedaba armar todo en la clase principal, sin embargo, se iba a ver algo feo, así que apliqué una hoja de estilos css para que se vea mucho más agradable a la vista.

Listing 4: Game.css

```
/* Estilos para el VBox (contenedor principal) */
.vbox {
    -fx-spacing: 20px;
    -fx-background-color: #000;
    -fx-alignment: center;
    -fx-halignment: center;
}
.toolBar{
    -fx-max-width: 626px;
}
/* Estilos para el GridPane */
#board {
    -fx-max-width: 626px;
    -fx-max-height: 624px;
    -fx-padding: 10px;
    -fx-border-color: #000;
    -fx-border-width: 2px;

}

/* Estilos para los botones */
.button {
    -fx-min-width: 60px;
    -fx-min-height: 60px;
    -fx-max-width: 60px;
    -fx-max-height: 60px;
    -fx-background-color: #fff; /* Color de fondo blanco por defecto */
    -fx-font-size: 14px;
    -fx-font-weight: bold;
    -fx-border-color: #000;
    -fx-border-width: 2px;
}
#closeButton{
    -fx-min-width: 30px;
    -fx-min-height: 30px;
    -fx-background-color: transparent;
```

```
574     -fx-background-image: url('../img/x-circle-regular-24.png');
575     -fx-background-repeat: no-repeat;
576     -fx-background-size: cover;
577 }
578 #minimizeButton{
579     -fx-min-width: 30px;
580     -fx-min-height: 30px;
581     -fx-background-color: transparent;
582     -fx-background-image: url('../img/minus-regular-24.png');
583     -fx-background-repeat: no-repeat;
584     -fx-background-size: cover;
585 }
586 #expandButton{
587     -fx-min-width: 30px;
588     -fx-min-height: 30px;
589     -fx-background-color: transparent;
590     -fx-background-image: url('../img/expand-alt-regular-24.png');
591     -fx-background-repeat: no-repeat;
592     -fx-background-size: cover;
593 }
594 #nextButton{
595     -fx-min-width: 30px;
596     -fx-min-height: 30px;
597     -fx-background-color: transparent;
598     -fx-background-image: url('../img/skip-next-circle-regular-24.png');
599     -fx-background-repeat: no-repeat;
600     -fx-background-size: cover;
601 }
602 #loadButton{
603     -fx-min-width: 30px;
604     -fx-min-height: 30px;
605     -fx-background-color: transparent;
606     -fx-background-image: url('../img/folder-open-regular-24.png');
607     -fx-background-repeat: no-repeat;
608     -fx-background-size: cover;
609 }
610 #saveButton{
611     -fx-min-width: 30px;
612     -fx-min-height: 30px;
613     -fx-background-color: transparent;
614     -fx-background-image: url('../img/save-regular-24.png');
615     -fx-background-repeat: no-repeat;
616     -fx-background-size: cover;
617 }
618 #infoButton{
619     -fx-min-width: 30px;
620     -fx-min-height: 30px;
621     -fx-background-color: transparent;
622     -fx-background-image: url('../img/info-circle-regular-24.png');
623     -fx-background-repeat: no-repeat;
624     -fx-background-size: cover;
625 }
```

- Con esto, solo me falta implementar la clase principal controladora para manejar el videojuego.

Listing 5: Clase controladora

```
629   package com.example.lab22;
630
631 import javafx.animation.KeyFrame;
632 import javafx.animation.Timeline;
633 import javafx.application.Platform;
634 import javafx.fxml.FXML;
635 import javafx.fxml.FXMLLoader;
636 import javafx.scene.Parent;
637 import javafx.scene.Scene;
638 import javafx.scene.control.*;
639 import javafx.scene.input.KeyCode;
640 import javafx.scene.input.MouseEvent;
641 import javafx.scene.layout.GridPane;
642 import javafx.scene.layout.VBox;
643 import javafx.stage.FileChooser;
644 import javafx.stage.Modality;
645 import javafx.stage.Stage;
646 import javafx.util.Duration;
647
648 import java.io.*;
649 import java.util.concurrent.atomic.AtomicBoolean;
650
651 public class VideoGameController {
652     @FXML
653     private GridPane gridPane = new GridPane();
654
655     private Army a1 = new Army(1);
656     private Army a2 = new Army(2);
657     private Board map = new Board();
658     private static int actualTurn = 1;
659     private int nEj1 = a1.getSoldiers().size();
660     private int nEj2 = a2.getSoldiers().size();
661     private boolean gameOver = false;
662
663
664     @FXML
665     private VBox rootVBox;
666
667
668     @FXML
669     public void initialize() {
670         initializeBoard();
671         runGame();
672     }
673
674     public void runGame() {
675         final boolean[] isAlertShown = {false};
676         Timeline timeline = new Timeline(new KeyFrame(Duration.seconds(1), event -> {
677             if (!gameOver) {
678                 gameOver();
679                 placeSoldiers();
680             } else if(!isAlertShown[0]){
681                 Platform.runLater(() -> {
682                     Alert alert = new Alert(Alert.AlertType.CONFIRMATION);
683                     alert.setTitle("Juego Terminado");
```

```
                alert.setHeaderText(null);
                alert.setContentText("El ganador es el Ejercito " + (nEj1 == 0 ? 2 : 1));
                alert.showAndWait();
                // Cerrar la aplicacion despues de mostrar el mensaje de juego terminado
                Platform.exit();
            });
            isAlertShown[0] = true;
        }
    }));
    timeline.setCycleCount(Timeline.INDEFINITE);
    timeline.play();

}
private void initializeBoard(){
    map.initializeTable();
    map.initializeArmy(a1);
    map.initializeArmy(a2);
    nEj1 = a1.getSoldiers().size();
    nEj2 = a2.getSoldiers().size();
}

private void placeSoldiers() {

    int rowCount = 1;
    int colCount = 1;

    for (javafx.scene.Node node : gridPane.getChildren()) {
        Soldier s = map.getTable()[rowCount - 1][colCount - 1];
        String color = s.getColor();
        String n = s.getAlias();
        if (node instanceof Button) {
            Button button = (Button) node;
            String buttonName = n;
            button.setText(buttonName);
            button.setStyle("-fx-background-color: " + color);
            colCount++;

            if (colCount > 10) {
                colCount = 1;
                rowCount++;
            }
        }
    }
}
public void onButtonClick(MouseEvent event){
    if (event.getSource() instanceof Button) {
        Button clickedButton = (Button) event.getSource();
        int columnIndex = GridPane.getColumnIndex(clickedButton);
        int rowIndex = GridPane.getRowIndex(clickedButton);

        Soldier s = map.getTable()[rowIndex][columnIndex];

        if(s.getStatus()){
            if(s.getArmyId() == actualTurn){
                getDirection(s);
            }else{
```

```
                    showAlert("Elige un soldado de tu propio Ejercito", "Error");
                }
            }

        }
    }
    private static void showAlert(String contentText, String title) {
        Alert alert = new Alert(Alert.AlertType.WARNING);
        alert.setTitle(title);
        alert.setHeaderText(null);
        alert.setContentText(contentText);
        alert.showAndWait();
    }
    private void battle(Soldier s1, Soldier s2){
        Soldier winner = Soldier.winner(s1, s2);


        if(winner.getArmyId() == 1){
            nEj2--;
        }else{
            nEj1--;
        }

        Alert alert = new Alert(Alert.AlertType.CONFIRMATION);
        alert.setTitle("Batalla");
        alert.setHeaderText(null);
        alert.setContentText("Los dos soldados batallaron, el ganador fue: " +
             winner.getName());

        alert.showAndWait();

        int x1 = s1.getX();
        int y1 = s1.getY();
        int x2 = s2.getX();
        int y2 = s2.getY();
        int x = winner.getX();
        int y = winner.getY();

        if(x1 == x && y1 == y){
            Soldier temp = map.getTable()[y2][x2];
            map.getTable()[y1][x1] = temp;
            map.getTable()[y2][x2] = winner;
            map.getTable()[y1][x1].destroy();
        }else{
            map.getTable()[y1][x1].destroy();
        }
    }
    private void move(Soldier s, int x, int y){
        Soldier s2 = map.getTable()[y][x];
        int oldX = s.getX();
        int oldY = s.getY();
        if(s2.getStatus()){
            if(s2.getArmyId() == s.getArmyId()){
                showAlert("No se permite el fuego aliado", "Error");
            }else{
                battle(s, s2);
```

```
795                 actualTurn = (actualTurn == 1) ? 2 : 1;
796             }
797         }else{
798             s.setX(x);
799             s.setY(y);
800             s2.setX(oldX);
801             s2.setY(oldY);
802 
803             Soldier temp = map.getTable()[y][x];
804             map.getTable()[y][x] = s;
805             map.getTable()[oldY][oldX] = temp;
806             actualTurn = (actualTurn == 1) ? 2 : 1;
807 
808         }
809     }
810     public void gameOver(){
811             if (nEj1 == 0 || nEj2 == 0) {
812                 gameOver = true;
813             }
814     }
815     public void changeTurn(MouseEvent event) {
816         Alert alerta = new Alert(Alert.AlertType.CONFIRMATION);
817         alerta.setTitle("Confirmacion");
818         alerta.setHeaderText("Quieres terminar tu turno?");
819         alerta.setContentText("Por favor, elige una opcion.");
820 
821         // Personalizar los botones de la alerta
822         ButtonType botonSi = new ButtonType("Si");
823         ButtonType botonNo = new ButtonType("No");
824         alerta.getButtonTypes().setAll(botonSi, botonNo);
825 
826         // Mostrar la alerta y esperar a que el usuario elija una opcion
827         alerta.showAndWait().ifPresent(resultado -> {
828             if (resultado == botonSi) {
829                 actualTurn = (actualTurn == 1) ? 2 : 1;
830             }
831         });
832 
833 
834 
835     }
836 
837     // Metodo para cerrar la ventana
838     @FXML
839     private void handleCloseButton(MouseEvent event) {
840         Stage stage = (Stage) rootVBox.getScene().getWindow();
841         stage.close();
842     }
843 
844     // Metodo para minimizar la ventana
845     @FXML
846     private void handleMinimizeButton(MouseEvent event) {
847         Stage stage = (Stage) rootVBox.getScene().getWindow();
848         stage.setIconified(true);
849     }
850 
```

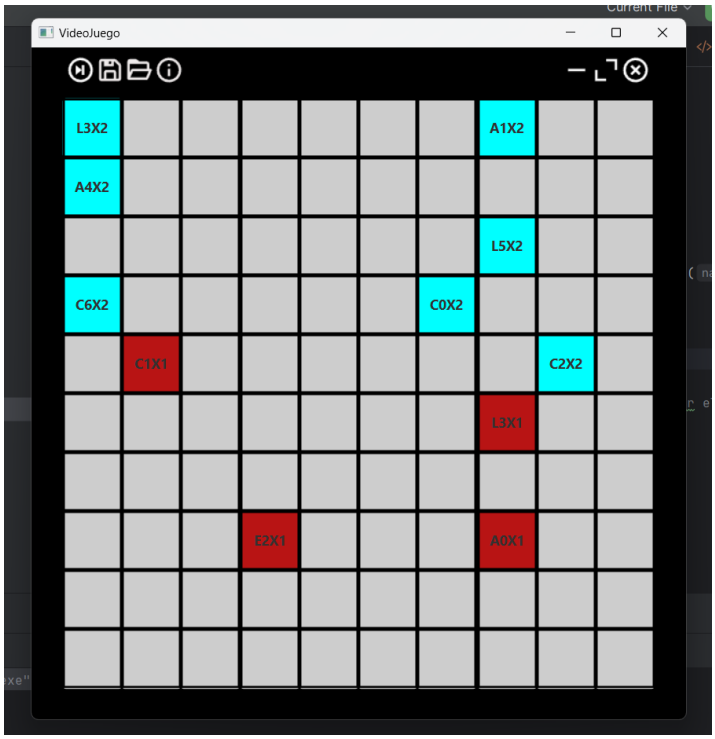
```
851     // Metodo para expandir/ restaurar la ventana (puedes personalizar segun tus necesidades)
852     @FXML
853     private void handleExpandButton(MouseEvent event) {
854         Stage stage = (Stage) rootVBox.getScene().getWindow();
855         if (stage.isMaximized()) {
856             stage.setMaximized(false);
857         } else {
858             stage.setMaximized(true);
859         }
860     }
861     @FXML
862     public void showRulesDialog(MouseEvent event) {
863         try {
864             FXMLLoader loader = new FXMLLoader(getClass().getResource("Rules.fxml"));
865             Parent rulesRoot = loader.load();
866 
867             Stage rulesStage = new Stage();
868             rulesStage.initModality(Modality.APPLICATION_MODAL);
869             rulesStage.setTitle("Reglas del Juego");
870 
871             Scene rulesScene = new Scene(rulesRoot, 400, 300);
872             rulesStage.setScene(rulesScene);
873             rulesStage.showAndWait();
874         } catch (IOException e) {
875             e.printStackTrace();
876         }
877     }
878 
879     private final AtomicBoolean isMoveAlertShown = new AtomicBoolean(false);
880     public void getDirection(Soldier s) {
881         final boolean[] result = {false};
882         String text = s.getName() + " " + s.getHP() + " HP";
883         Alert alerta = new Alert(Alert.AlertType.CONFIRMATION);
884         alerta.setTitle("Confirmacion");
885         alerta.setHeaderText(text + " seleccionado");
886         alerta.setContentText("Por favor, elige una opcion.");
887 
888         ButtonType botonSi = new ButtonType("Mover Soldado");
889         ButtonType botonNo = new ButtonType("Cancelar");
890         alerta.getButtonTypes().setAll(botonSi, botonNo);
891 
892         alerta.showAndWait().ifPresent(resultado -> {
893             if (resultado == botonSi) {
894                 Platform.runLater(() -> {
895                     Alert moveAlert = new Alert(Alert.AlertType.INFORMATION);
896                     moveAlert.setTitle("Mueve tu Soldado");
897                     moveAlert.setHeaderText("Usa las teclas para mover tu soldado.");
898                     moveAlert.setContentText("AWXD (vertical-horizontal), QEZC (diagonal) o S
                            (no moverse)");
899 
900                     Scene moveScene = moveAlert.getDialogPane().getScene();
901                     moveScene.setOnKeyPressed(event -> {
902                         KeyCode pressedKey = event.getCode();
903 
904                         if (!isMoveAlertShown.get()) {
905                             moveSoldado(pressedKey, s);
```

```
                        isMoveAlertShown.set(true);
                        moveAlert.close();
                    }
                });

                moveAlert.showAndWait();
                isMoveAlertShown.set(false);
            });
        }
    });


}
private void moveSoldado(KeyCode key, Soldier s) {
    int x = 0, y = 0;

    switch (key) {
        case W:
            y = -1;
            break;
        case A:
            x = -1;
            break;
        case D:
            x = 1;
            break;
        case S:
            // No moverse en la posicion actual
            break;
        case Q:
            x = -1;
            y = -1;
            break;
        case E:
            x = 1;
            y = -1;
            break;
        case Z:
            x = -1;
            y = 1;
            break;
        case C:
            x = 1;
            y = 1;
            break;
        case X:
            y = 1;
            break;
        default:
            break;
    }

    // Calcular la nueva posicion
    int newY = s.getY() + y;
    int newX = s.getX() + x;

```

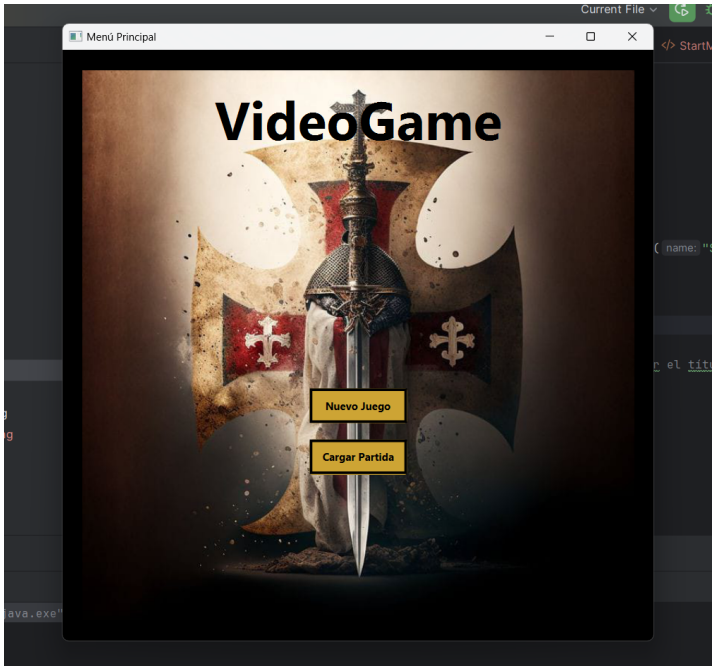
```
962         // Verificar si la nueva posicion es valida
963         if (isValidMove(newX, newY)) {
964             // Mover el soldado
965             move(s, newX, newY);
966             // Actualizar la posicion del soldado en la matriz map.getTable()
967 
968         } else {
969             showAlert("Casilla invalida (Solo puedes moverte una casilla en cada direccion)",
                    "Error");
970         }
971     }
972 
973     // Metodo para verificar si la posicion es valida
974     private boolean isValidMove(int x, int y) {
975         return x >= 0 && x < 10 && y >= 0 && y < 10;
976     }
977     @FXML
978     public void saveGame(MouseEvent event){
979         // Crear un objeto FileChooser
980         FileChooser fileChooser = new FileChooser();
981         fileChooser.setTitle("Guardar Archivo");
982 
983         // Agregar una extension al FileChooser (opcional)
984         FileChooser.ExtensionFilter extFilter = new FileChooser.ExtensionFilter("Archivos de
                Texto (*.txt)", "*.txt");
985         fileChooser.getExtensionFilters().add(extFilter);
986 
987         Stage stage = (Stage) rootVBox.getScene().getWindow();
988 
989         // Mostrar el cuadro de dialogo para guardar archivo
990         File file = fileChooser.showSaveDialog(stage);
991 
992         if (file != null) {
993             // El usuario ha seleccionado un archivo, ahora puedes escribir en el
994             writeToFile(file);
995         }
996     }
997     private void writeToFile(File file) {
998         try (BufferedWriter writer = new BufferedWriter(new FileWriter(file))) {
999             // Escribir contenido en el archivo
1000            writer.write("#" + a1.getId());
1001            writer.newLine();
1002            for(Soldier s: a1.getSoldiers()){
1003                writer.write(s.toString());
1004                writer.newLine();
1005            }
1006            writer.newLine();
1007            writer.write("#" + a2.getId());
1008            writer.newLine();
1009            for(Soldier s: a2.getSoldiers()){
1010                writer.write(s.toString());
1011                writer.newLine();
1012            }
1013 
1014        } catch (IOException e) {
1015            e.printStackTrace();
```

```
1016        }
1017    }
1018    @FXML
1019    public void loadGame(MouseEvent event){
1020        FileChooser fileChooser = new FileChooser();
1021        fileChooser.setTitle("Cargar Archivo");
1022
1023        // Agregar una extension al FileChooser (opcional)
1024        FileChooser.ExtensionFilter extFilter = new FileChooser.ExtensionFilter("Archivos de
                Texto (*.txt)", "*.txt");
1025        fileChooser.getExtensionFilters().add(extFilter);
1026
1027        Stage stage = (Stage) rootVBox.getScene().getWindow();
1028
1029        // Mostrar el cuadro de dialogo para cargar archivo
1030        File file = fileChooser.showOpenDialog(stage);
1031
1032        if (file != null) {
1033            // El usuario ha seleccionado un archivo, ahora puedes leer de el
1034            readFromFile(file);
1035        }
1036    }
1037    public void readFromFile(File file) {
1038        try (BufferedReader reader = new BufferedReader(new FileReader(file))) {
1039            // Limpiar ejercitos existentes antes de cargar nuevos datos
1040            a1.clearSoldiers();
1041            a2.clearSoldiers();
1042
1043            String line;
1044            while ((line = reader.readLine()) != null) {
1045                if (line.startsWith("#")) {
1046                    // Identificar el ejercito por el ID
1047                    int armyId = Integer.parseInt(line.substring(1));
1048                    Army currentArmy = (armyId == a1.getId()) ? a1 : a2;
1049
1050                    // Leer las siguientes lineas correspondientes a los soldados
1051                    while ((line = reader.readLine()) != null && (!line.startsWith("#") &&
                            !line.trim().isEmpty())) {
1052                        // Crear un nuevo soldado a partir de la linea y agregarlo al ejercito
1053                        Soldier soldier = Soldier.parseSoldier(line);
1054                        currentArmy.addSoldier(soldier);
1055                    }
1056                }
1057            }
1058            // Por ejemplo, volver a colocar los soldados en la interfaz grafica
1059            map.initializeTable();
1060            map.deployArmy(a1);
1061            map.deployArmy(a2);
1062
1063            placeSoldiers();
1064        } catch (IOException e) {
1065            e.printStackTrace();
1066        }
1067    }
1068}
```

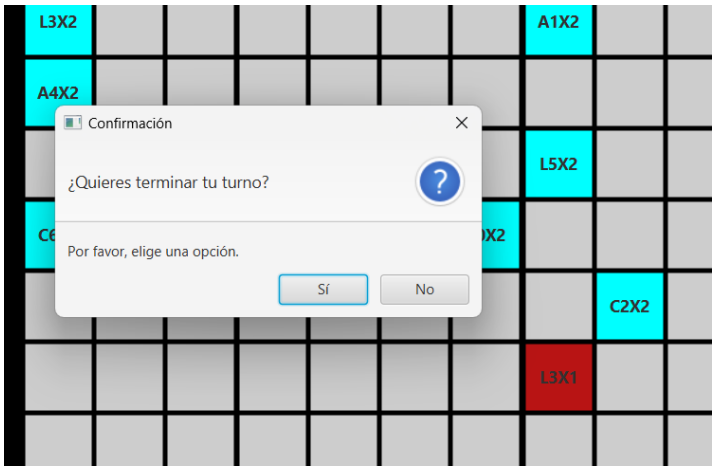
- En este código, tengo el método onButtonClick, que lo que hace es primero buscar si hay un soldado en esa posición, (usando de referencia el mapa) una vez que detecta un soldado, revisa si es del ejército respectivo al turno actual (si es turno del ejercito 1 o 2) para que solo puedas mover soldados cuando te toca a ti.
- Luego de eso, pregunto al usuario a que coordenadas lo quiere mover, revisando que sea una coordenada a 1 casilla de distancia, usando las teclas QWEADZXC.
- Luego de eso ve si hay un soldado en la casilla de destino y si lo hay, revisa que no sea del mismo ejército. Si no lo es realiza una batalla y coloca ahí al ganador, para terminar.
- Esa es la lógica principal, también está el método runGame que trabaja en un hilo, constantemente refrescando los gráficos. Y el resto de métodos son simplemente para que la interfaz funcione, como los métodos para minimizar o cerrar usando los botones de la barra de tareas.
- También está el método para determinar si el juego se ha terminado o no y se cierra la pestaña.
- Al final el tablero queda de esta forma al iniciar el juego.

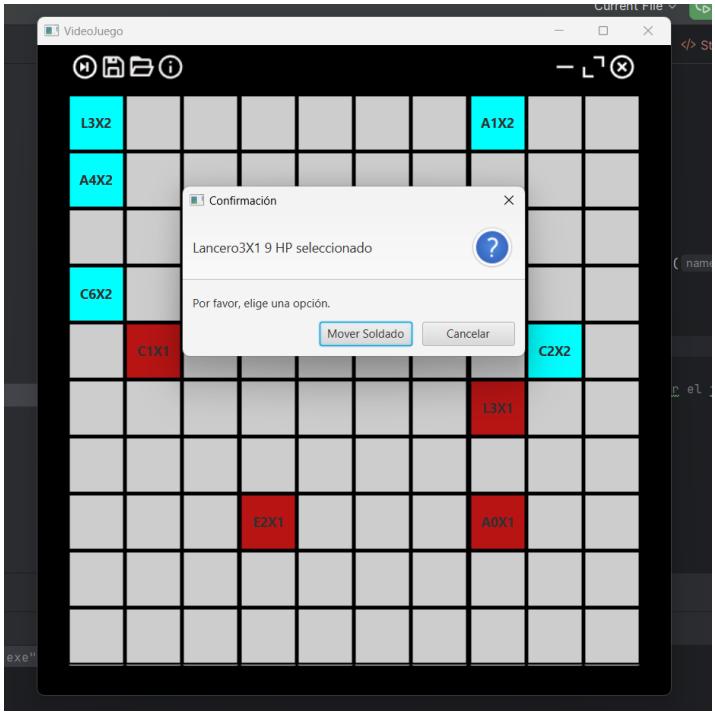


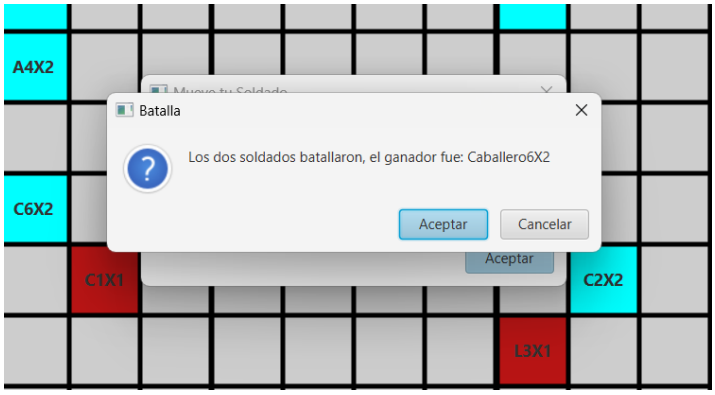
- Y con algunas capturas del funcionamiento del videojuego.

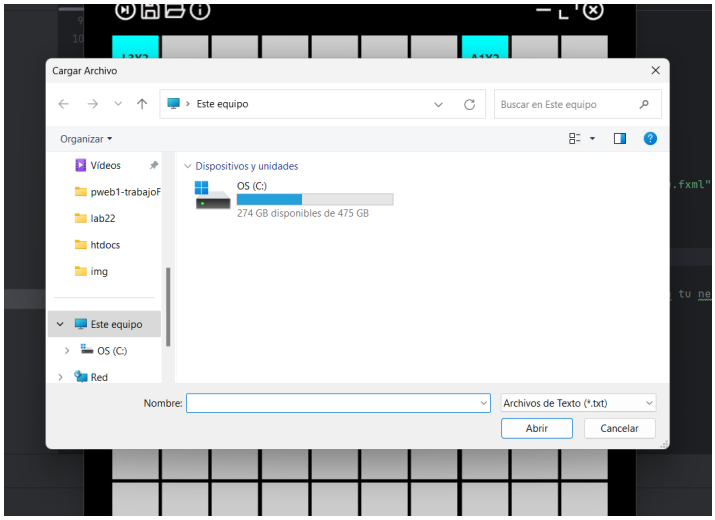

Menú Principal
VideoGame
Nuevo Juego
Cargar Partida


Error
Elige un soldado de tu propio Ejército
Aceptar
C1X1
C2X2


L3X2
A1X2
A4X2
Confirmación
¿Quieres terminar tu turno?
L5X2
Por favor, elige una opción.
Sí
No
C2X2
L3X1


VideoJuego
L3X2
A1X2
A4X2
Confirmación
Lancero3X1 9 HP seleccionado
Por favor, elige una opción.
Mover Soldado
Cancelar
C6X2
C1X1
C2X2
L3X1
E2X1
A0X1


A4X2
Batalla
Los dos soldados batallaron, el ganador fue: Caballero6X2
Aceptar
Cancelar
C6X2
C1X1
C2X2
L3X1



- Estos cambios fueron subidos al respositorio, a través de commits.

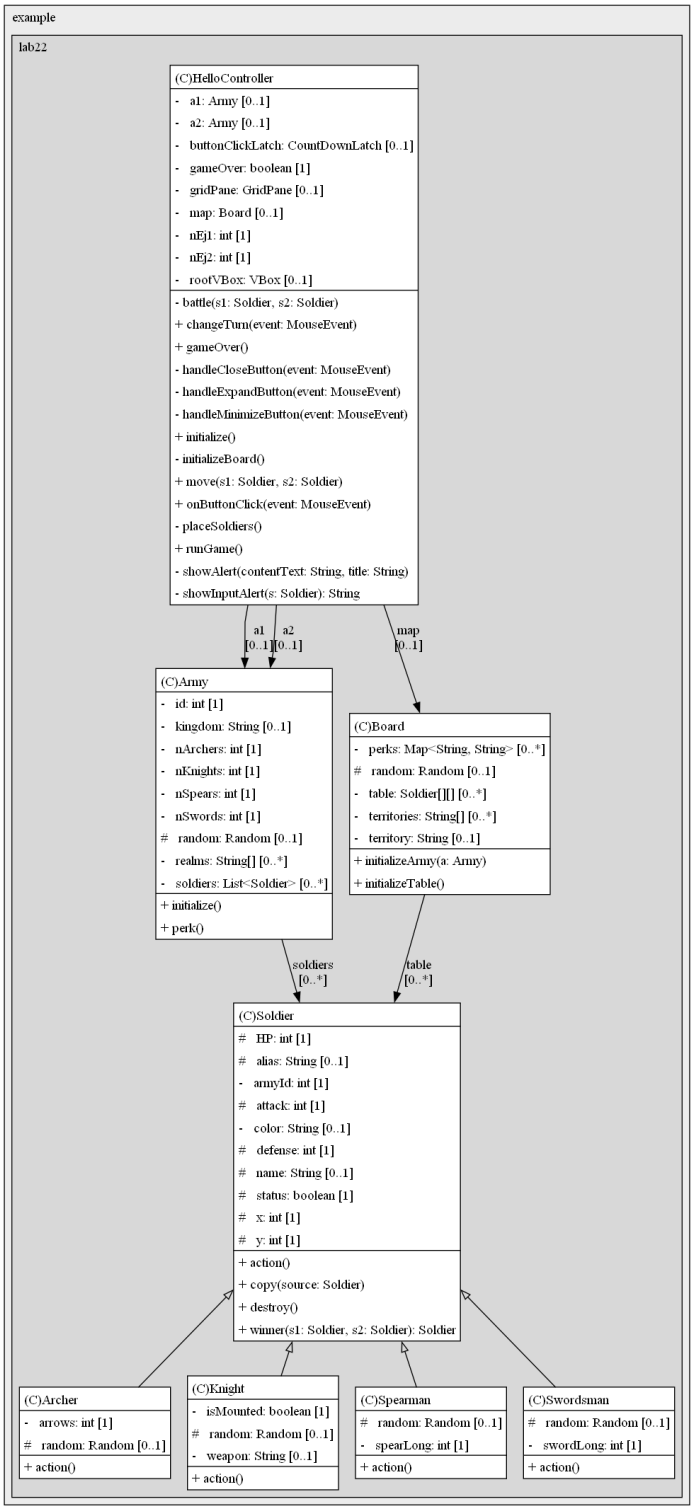
Listing 6: Commit

```
   $commit feec5bf606898afbaaaadd1a80b1f0c0c356a83f (HEAD -> main, origin/main)
Author: RYAN VALDIVIA <rvaldiviase@unsa.edu.pe>
Date:   Sat Jan 20 11:48:31 2024 -0500

    Estableciendo las clases necesarias y realizando pruebas de JavaFx, usando IntelliJ Idea
```

Listing 7: Commit

```
   $commit 44f6eaa51af046b665943d901d3ac65b04040216 (HEAD -> main, origin/main)
Author: RYAN VALDIVIA <rvaldiviase@unsa.edu.pe>
Date:   Mon Jan 22 13:38:31 2024 -0500

    Trabajo terminado, el juego es 100% funcional
```

- Además, aquí está el diagrama UML de todo el proyecto.


example
lab22
(C)HelloController
- a1: Army [0..1]
- a2: Army [0..1]
- buttonClickLatch: CountDownLatch [0..1]
- gameOver: boolean [1]
- gridPane: GridPane [0..1]
- map: Board [0..1]
- nEj1: int [1]
- nEj2: int [1]
- rootVBox: VBox [0..1]
- battle(s1: Soldier, s2: Soldier)
+ changeTurn(event: MouseEvent)
+ gameOver()
- handleCloseButton(event: MouseEvent)
- handleExpandButton(event: MouseEvent)
- handleMinimizeButton(event: MouseEvent)
+ initialize()
- initializeBoard()
+ move(s1: Soldier, s2: Soldier)
+ onButtonClick(event: MouseEvent)
- placeSoldiers()
+ runGame()
- showAlert(contentText: String, title: String)
- showInputAlert(s: Soldier): String
a1 [0..1]
a2 [0..1]
map [0..1]
(C)Army
- id: int [1]
- kingdom: String [0..1]
- nArchers: int [1]
- nKnights: int [1]
- nSpears: int [1]
- nSwords: int [1]
random: Random [0..1]
- realms: String[] [0..*]
- soldiers: List<Soldier> [0..*]
+ initialize()
+ perk()
(C)Board
- perks: Map<String, String> [0..*]
random: Random [0..1]
- table: Soldier[][] [0..*]
- territories: String[] [0..*]
- territory: String [0..1]
+ initializeArmy(a: Army)
+ initializeTable()
soldiers [0..*]
table [0..*]
(C)Soldier
HP: int [1]
alias: String [0..1]
- armyId: int [1]
attack: int [1]
- color: String [0..1]
defense: int [1]
name: String [0..1]
status: boolean [1]
x: int [1]
y: int [1]
+ action()
+ copy(source: Soldier)
+ destroy()
+ winner(s1: Soldier, s2: Soldier): Soldier
(C)Archer
- arrows: int [1]
random: Random [0..1]
+ action()
(C)Knight
- isMounted: boolean [1]
random: Random [0..1]
- weapon: String [0..1]
+ action()
(C)Spearman
random: Random [0..1]
- spearLong: int [1]
+ action()
(C)Swordsman
random: Random [0..1]
- swordLong: int [1]
+ action()

6. Rúbricas

6.1. Entregable Informe

Tabla 1: Tipo de Informe

Informe	
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y facil de leer.

6.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumplio con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos lo items.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 2: Niveles de desempeño

	Nivel			
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Tabla 3: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	2	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	2	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
6. Fechas	Las fechas de modificación del código fuente estan dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	X	4	
Total		20		18	

7. Referencias

- Fundamentos de la programación 2 - Tópicos de la programación Orientada a Objetos (Marco Aedo)