

## AP2\_LAB3\_TRAPTHEPACMAN

### 1. Especificación de requerimientos funcionales:

<b>Nombre</b>	<b>R.#1. Mostrar los pacman en la pantalla mediante la escogencia del nivel o (en caso de existir) abrir un archivo serializable con el estado guardado del anterior juego.</b>
<b>Resumen</b>	El juego será capaz de dibujar varios pacman según la especificación de la escogencia del nivel por parte del usuario, donde cada pacman tendrá características diferentes a los demás.
<b>Entradas</b>	
<b>Nivel.</b>	
<b>Resultados</b>	
El juego se ha iniciado.	

<b>Nombre</b>	<b>R.#2. Construir el pacman mediante las primitivas que proporciona el lenguaje.</b>
<b>Resumen</b>	El programa está capacitado de los métodos de la clase pacman para construir cada uno de manera primitiva especificando su radio, posición, ángulo de inicio, longitud y tipo de arco.
<b>Entradas</b>	
Radio, posición, ángulo de inicio, longitud y tipo de arco.	
<b>Resultados</b>	
Se ha creado un pacman mediante las primitivas del lenguaje.	

<b>Nombre</b>	<b>R.#3. Leer un archivo de texto para cargar el juego</b>
<b>Resumen</b>	El programa estará provisto de un lector de archivos que convertirá las líneas de texto en las especificaciones antes dichas para crear los pacman y asignarle un hilo a cada uno.
<b>Entradas</b>	
<b>Nivel.</b>	
<b>Resultados</b>	
Se ha iniciado el nivel.	

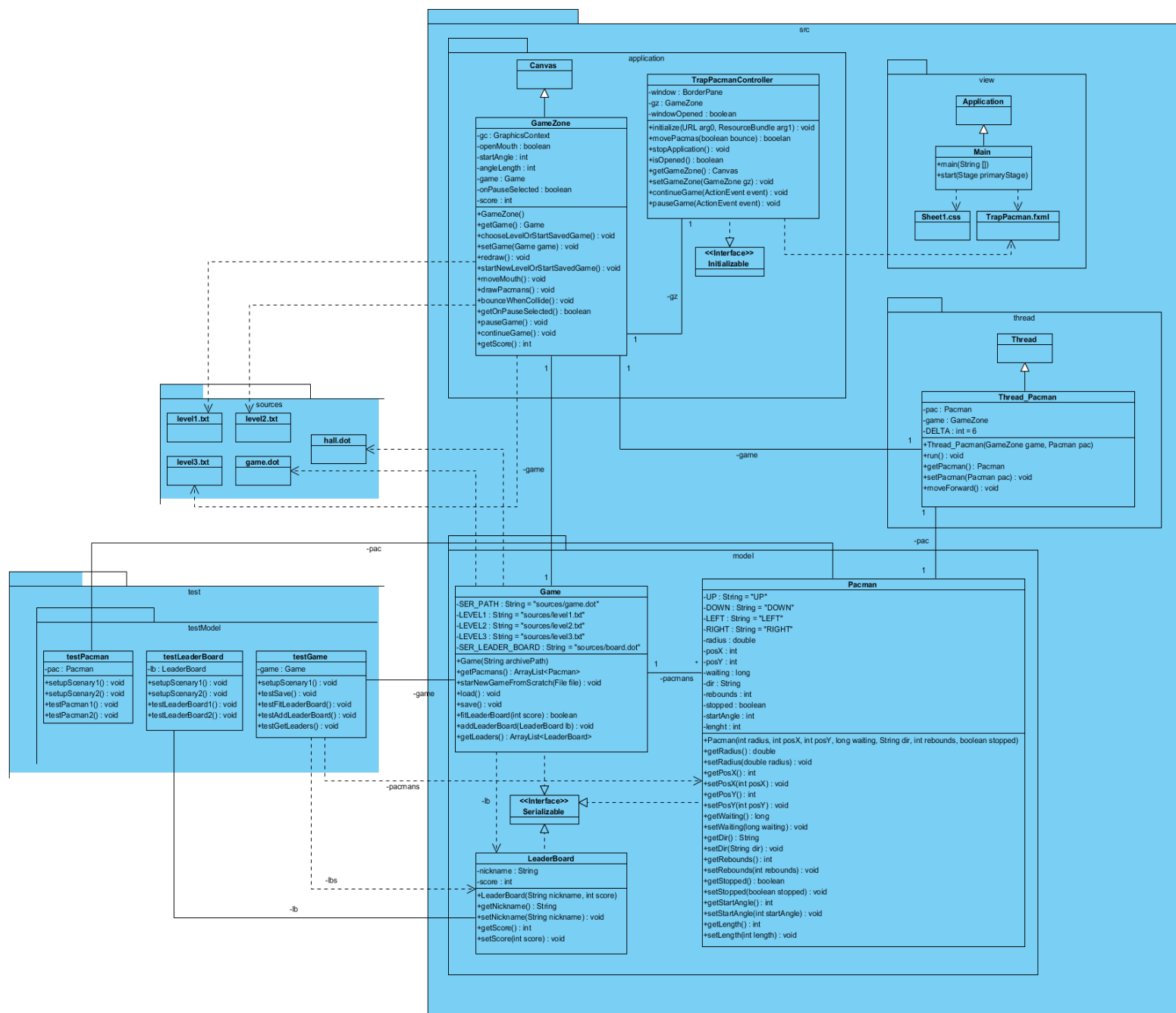
<b>Nombre</b>	<b>R.#4. Guardar el estado del juego cuando el usuario lo desee.</b>
<b>Resumen</b>	El programa será capaz de guardar el estado del juego en cualquier momento de cada nivel en un archivo serializable.
<b>Entradas</b>	
<b>Guardar.</b>	
<b>Resultados</b>	
Se ha guardado el estado actual del juego.	

<b>Nombre</b>	<b>R.#5. Mostrar la tabla de mejores puntuaciones.</b>
<b>Resumen</b>	El juego será capaz de mostrarle la tabla de mejores puntuaciones cuando este especifique dicha opción en la pestaña "View" del juego.
<b>Entradas</b>	
<b>Mejores puntuaciones.</b>	
<b>Resultados</b>	
Tabla de mejores puntuaciones.	

<b>Nombre</b>	<b>R.#6. Permitir al usuario agregarse a la tabla de mejores puntuaciones cuando su puntaje sea apto.</b>
<b>Resumen</b>	El programa evaluará si el puntaje del jugador al acabar un nivel puede entrar en la tabla de mejores puntuaciones. Si es así, el juego le pedirá su nombre y lo agregará en la tabla en su posición correspondiente.
<b>Entradas</b>	
<b>Puntaje del jugador.</b>	
<b>Resultados</b>	
El jugador se/no ha agregado a la tabla de mejores puntuaciones.	

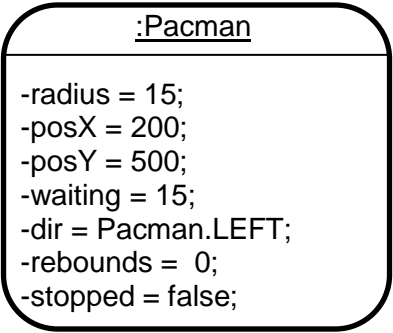
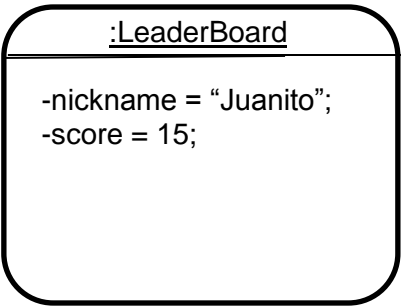
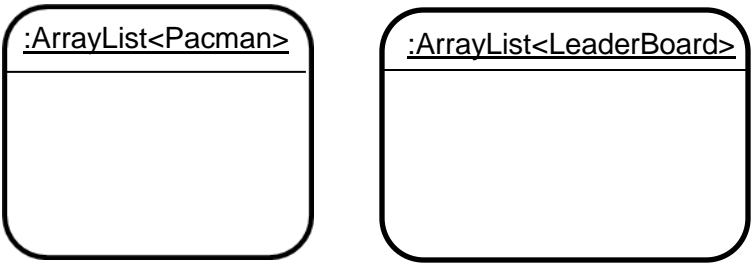
<b>Nombre</b>	<b>R.#7. Detener los pacman cuando son cliqueados</b>
<b>Resumen</b>	El programa tendrá la capacidad de un área de intersección entre el evento del mouse y el pacman para poder detenerlo.
<b>Entradas</b>	
<b>Evento del mouse.</b>	
<b>Resultados</b>	
El pacman se ha detenido.	

<b>Nombre</b>	<b>R.#8. Deserializar la información de la tabla de mejores puntuaciones cuando se vuelve a abrir el juego</b>
<b>Resumen</b>	El juego será capaz de deserializar el estado anterior de la tabla de puntuaciones del anterior juego para seguir compitiendo con los jugadores antecesores.
<b>Entradas</b>	
<b>&lt;Ninguna&gt;</b>	
<b>Resultados</b>	
Tabla de puntuaciones deserializada.	



### 3. Diseño de casos de prueba unitarias:

#### Configuración de los escenarios

Nombre	Clase	Escenario
setupScenary1	Pacman	<b>vacío</b>
setupScenary2	Pacman	 <pre> classDiagram     class Pacman {         -radius = 15;         -posX = 200;         -posY = 500;         -waiting = 15;         -dir = Pacman.LEFT;         -rebounds = 0;         -stopped = false;     }         </pre>
setupScenary1	LeaderBoard	<b>vacío</b>
setupScenary2	LeaderBoard	 <pre> classDiagram     class LeaderBoard {         -nickname = "Juanito";         -score = 15;     }         </pre>
setupScenary1	Game	 <pre> classDiagram     class ArrayListPacman {         &lt;&lt;array list&gt;&gt;     }     class ArrayListLeaderBoard {         &lt;&lt;array list&gt;&gt;     }         </pre>

**Objetivo de la prueba:** Verificar la correcta creación de una instancia de Pacman.

Clase	Método	Escenario	Valores de entrada	Resultado
Pacman	Pacman	setupScenario1	pacTest	Se ha creado una instancia de Pacman exitosamente.

**Objetivo de la prueba:** Verificar la correcta creación de una instancia de Pacman cuando ya hay creada una previa.

Clase	Método	Escenario	Valores de entrada	Resultado
Pacman	Pacman	setupScenario2	pacTest2	Se ha creado otra instancia de Pacman exitosamente.

**Objetivo de la prueba:** Verificar la correcta creación de una instancia de Pacman.

Clase	Método	Escenario	Valores de entrada	Resultado
LeaderBoard	LeaderBoard	setupScenario1	lbTest	Se ha creado una instancia de LeaderBoard exitosamente.

**Objetivo de la prueba:** Verificar la correcta creación de una instancia de LeaderBoard cuando ya hay creada una previa.

Clase	Método	Escenario	Valores de entrada	Resultado
LeaderBoard	LeaderBoard	setupScenario2	lbTest2	Se ha creado otra instancia de LeaderBoard exitosamente.

**Objetivo de la prueba:** Verificar la correcta creación de un archivo serializable que contiene el estado del juego en un momento predefinido.

Clase	Método	Escenario	Valores de entrada	Resultado
Game	save	setupScenario1	file	El juego ha sido guardado correctamente.

**Objetivo de la prueba:** Verificar la correcta verificación del método fitLeaderBoard con un puntaje dado.

Clase	Método	Escenario	Valores de entrada	Resultado
Game	fitLeaderBoard	setupScenario1	score = 27	El puntaje del jugador ha sido verificado.

**Objetivo de la prueba:** Verificar si un jugador es añadido a la tabla de puntuaciones que es serializable.

Clase	Método	Escenario	Valores de entrada	Resultado
Game	addLeaderBoard	setupScenario1	test	El jugador fue agregado correctamente a la tabla de puntuaciones.

**Objetivo de la prueba:** Verificar la correcta obtención de los jugadores que están en la tabla de posiciones al deserealizar.

Clase	Método	Escenario	Valores de entrada	Resultado
Game	getLeaders	setupScenario1	N/A	Se han recuperado los tres jugadores existentes.

#### 4. Tabla de trazabilidad de requerimientos vs métodos

Requerimiento funcional	Método que lo resuelve	Clase (s)
<b>R.#1. Mostrar los pacman en la pantalla mediante la escogencia del nivel o (en caso de existir) abrir un archivo serializable con el estado guardado del anterior juego.</b>	<pre> public void chooseLevelOrStartSavedGame() {     ChoiceDialog&lt;String&gt; cd = new ChoiceDialog&lt;String&gt;(Game.LEVEL1, Game.LEVEL2, Game.LEVEL3, Game.SER_PATH);     cd.showAndWait();     if(cd.getResult() != null)     {         try {             game = new Game(cd.getSelectedItem().toString());              for(Pacman pac:game.getPacmans()) {                  Thread_Pacman pt = new Thread_Pacman(this, pac); </pre>	<b>GameZone</b>

	<pre> pt.setDaemon(true);  pt.start();      }     } catch (ClassNotFoundException   IOException e) {      e.printStackTrace();     }  } </pre>	
<b>R.#2. Construir el pacman mediante las primitivas que proporciona el lenguaje.</b>	<pre> public void redraw() {     moveMouth();     drawPacmans();     bounceWhenCollide();     score = 0;     for(Pacman pac:game.getPacmans()) {         score += pac.getRebounds();     }     gc.setFill(Color.BLACK);     gc.fillText("Rebounds: " + score, 800, 595);     gc.setFill(Color.YELLOW); } </pre>	<b>GameZone</b>
<b>R.#3. Leer un archivo de texto para cargar el juego</b>	<pre> public void chooseLevelOrStartSavedGame() {     ChoiceDialog&lt;String&gt; cd = new ChoiceDialog&lt;String&gt;(Game.LEVEL1, Game.LEVEL2, Game.LEVEL3, Game.SER_PATH);     cd.showAndWait();     if(cd.getResult() != null)     {         try {             game = new Game(cd.getSelectedItem().toString());              for(Pacman pac:game.getPacmans()) {                  Thread_Pacman pt = new Thread_Pacman(this, pac);                  pt.setDaemon(true);                  pt.start();              }         } catch (ClassNotFoundException   IOException e) {              e.printStackTrace();         }     } } </pre>	<b>GameZone</b>



R.#4. Guardar el estado del juego cuando el usuario lo desee.	<pre>private void load() throws IOException, ClassNotFoundException {     File file = new File(SER_PATH);     FileInputStream in = new FileInputStream(file);     ObjectInputStream ins = new ObjectInputStream(in);     pacmans = (ArrayList)ins.readObject();     ins.close();     in.close(); }</pre>	Game
R.#5. Mostrar la tabla de mejores puntuaciones.	<pre>public void handle(ActionEvent event) case "Leader board":</pre>	TrapPacmanController, FileMenuItemSelected
R.#6. Permitir al usuario agregarse a la tabla de mejores puntuaciones cuando su puntaje sea apto.	<pre>public boolean fitLeaderBoard(int score)</pre>	Game
R.#7. Detener los pacman cuando son cliqueados	<pre>public void handle(ActionEvent event)</pre>	VerifyPacmanCaught, GameZone
R.#8. Deserializar la información de la tabla de mejores puntuaciones cuando se vuelve a abrir el juego	<pre>public void handle(ActionEvent event) case "Leader board":</pre>	TrapPacmanController, FileMenuItemSelected