The application is a Maze Runner 2.5D Game called *Labirinto Corridore*. It is implemented in Java using object oriented concepts, MVC architectural pattern and various design patterns listed below. It is developed using isometric images and animated sprites and has a save and load feature. The GUI is made using SWING in which we made use of its Graphics class which is the abstract super class for all graphics contexts which allow an application to draw onto components.

Game Description

This game is like the famous Pac-Man game, instead, here we will have a maze, and you will be the actor (maze runner), you should run through the maze to find your way out! But be careful not to hit a bomb, you can also destroy any obstacle you face by a weapon, the weapon will be fulfilled with limited ammo. While you are moving inside the maze, you should collect some gifts in your way. The end of the game is by reaching the gate of the maze, or if you died due to hitting multiple bombs. The maze runner is moved using the 4 arrow keys in the keyboard, and you should fire the weapon with the space bar.

Design Overview

The code's architecture follows MVC (Model View Controller) architectural pattern.
The Model is responsible for managing the game's entities (load images, sprites and animation), plus saving and loading the current state of the game.
The View is the user's interface, the game maze window and the score panel. It doesn't talk directly to the model but through the controller.
The controller commands the View's actions and command what to display. The controller takes command and information from the user, process this information and talks to the Model to retrieve or send data. It also takes all compiled information and command the View to how represent it to the user.

The code implements some important design patterns such as:

Singleton DP: used to create classes from which we can create only one object. The Runner and the Maze classes are used as singleton because only one player (runner) and one maze are supported.

Factory DP: used in constructing new elements that are added to the game window display (ex: tiles, gifts, bombs, prizes, etc…).

Observer DP: used in the info panel which observes the runner's health and score.

State DP: game start and stop states, hitting a bomb while wearing an armor or not, health state controls the movement speed.

Memento DP: used in the checkpoint functionality. The Memento class is the game's state at a specific point in time, it saves the current state of the game (runner position, number of bullets, score, health, etc...). CareTaker class contains an ArrayList: history, which contain the different game's states/Mementos at each checkpoint used in saving the game's state at a specific checkpoint as well as restoring the game's state.