COMP 132 – Spring 2021
Designing a Video Game using Swing
Space Pong

Introduction

In this project, you are expected to put your Java and Swing development skills into designing a video game. The goal of this game is to collect as many points as possible within the time limits, either by bouncing the ball with the paddle, or shooting the bonus items in the sky. The player will be given a time limit and 3 lives to play, which could be altered by the bonus items as well.

This document serves a guideline to describe some of the design choices, while the implementation will be mostly up to you. Keep an eye out for better software practices such as meaningful abstraction, code cleanliness, documentation etc.

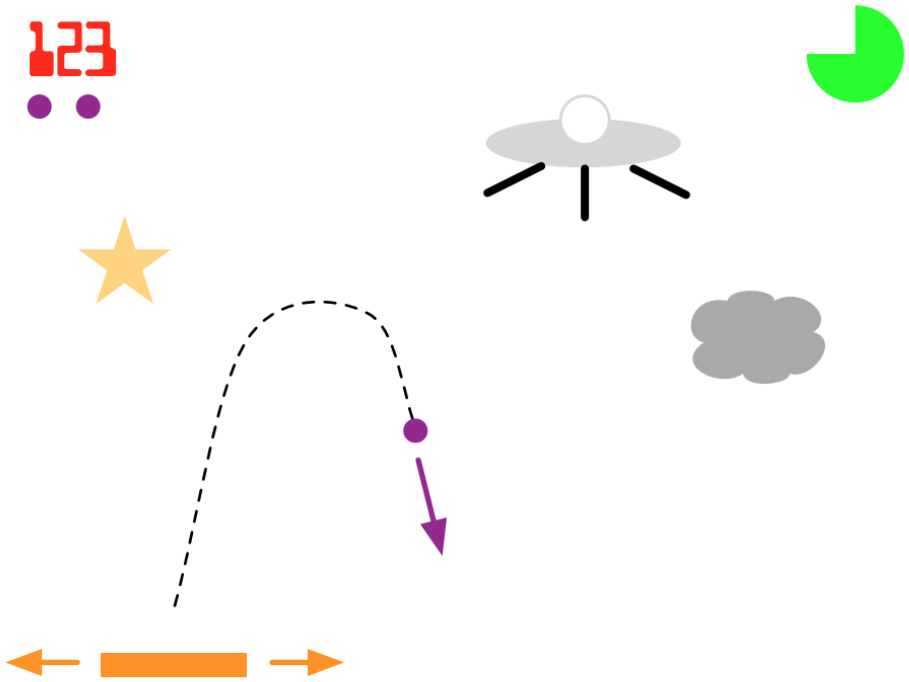
The project is different from the labs. In the labs, we specified exactly what to do. In this project, you will develop a program form scratch. You are expected to study swing compoents and their API, and use them. You have sufficient Java knowledge and practice to do it. We will have a dedicated discussion forum for your questions.

The project is an individual project. You are free to look at any resource (tutorials, youtube videos, books, etc). In fact we will provide such resources for you. You can discuss with your friends the concepts, but CANNOT share code. The code MUST BE belong to you. YOU ARE NOT ALLOWED TO COPY-PASTE ANY CODE FROM ANYWHERE. You need to type your code. Otherwise, plagiarism tools will find such similar code and disciplinary action will be taken.

Below, we describe the game, specify some constrains. You are expected to make a decision for anything not specified.

Game Dynamics

The player controls a paddle at the bottom of the game panel that moves horizontally by hitting left or right arrow keys on the keyboard. A ball will be launched into play, following a parabolic trajectory. The player's goal is to intercept the ball before it falls beneath the bottom of the screen. When the ball bounces back into the game area, the user gains one point. A rough sketch of the game might look like the following, but feel free to make your own design choices.



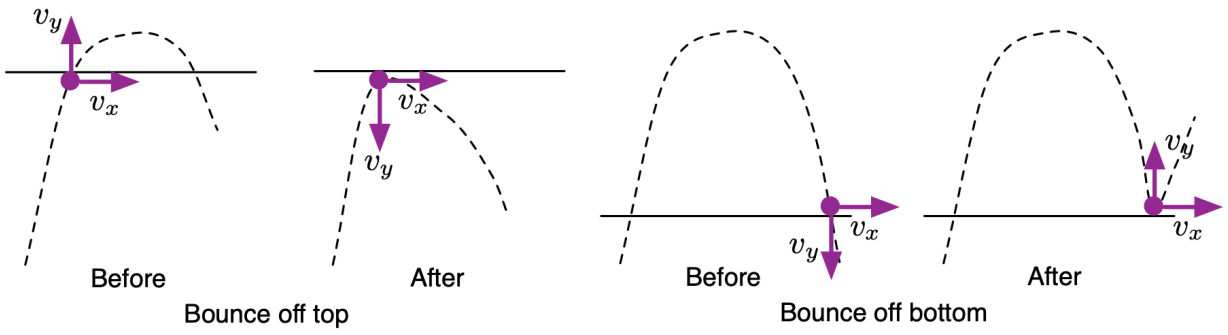
There will be certain physics dictating the motion of the ball, which you will have to implement in computing the next location to render the ball at. Remember your physics course and equations for free fall under gravitation (without friction):

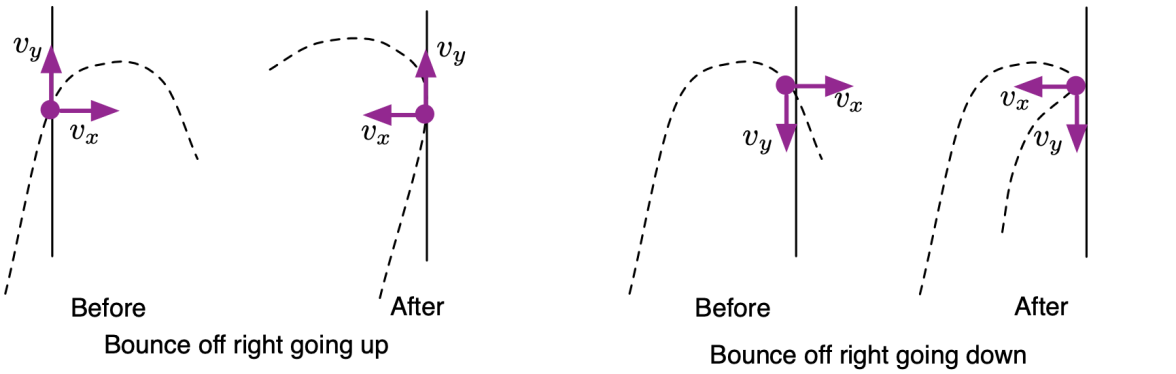
$$v_x(t) = v_0 cos(\theta)$$
$$v_y(t) = v_0 sin(\theta) - gt$$
$$x(t) = v_0 cos(\theta) . t$$
$$y(t) = v_0 sin(\theta) . t - \frac{1}{2} x\, gt^2$$

You will approximate it to animate a ball under gravity-like force field. The ball has an initial position x(0) and y(0), initial velocity v(0) (vx(0) and vy(0) are the x, y components). components) at time 0. We can compute the velocity and position of the ball at time t+1 as:

$v_x(t+1) = v_x(t)$ (since gravity has only y direction, vx does not change)
$v_y(t+1) = v_y(t) –9.8*ratio$. ratio is a constant that you need to determine for a smooth animation) and the position at t+1
$x(t+1) = x(t)+v_x(t+1)$
$y(t+1) = y(t)-v_y(t+1)$ (we negate vy, because the swing window has upper-left corner at 0,0 and y direction increases as we go down).

These equations are explicitly provided, for you need to implement the behavior that the ball not only bounces off of the paddle, but also top and sides of the game window. **Please note that the ball does not travel in straight lines, but rather in parabolic curves.** In terms of the dynamics, bouncing implies reflection of one of the velocity components, essentially flipping its sign.





You will need these equations and discrete timer steps to animate the ball. In the absence of a reflection, the speed can be updated using only the acceleration, and the next location can be computed from the speed. Careful implementation of the reflections is needed to avoid edge cases that introduce undesired behavior.

The player will control the paddle by left and right arrow keyboard keys. The player will be given some number of lives, a countdown from some number of seconds. Besides the game, you need to update and display the score.

You need to additionally include images of bonus items that appear in the sky and implement their behavior. These are, namely, an UFO ship, a meteorite, and a star. Feel free to include more elements if you wish to make your game more creative.

1. UFO Ship: The user gets “abducted”, so the user loses one of the lives.
2. Meteorite: The ball hits a meteorite, the ball should travel faster now.

3. Star: The user caught a star and he/she is starstruck! Add some more points to your score now!

UI Elements

This exercises serves as a comprehensive practice of the Swing fundamentals, so you are expected to demonstrate a rich use of the UI elements such as JFrame, JPanel, JButton, JTextField, some combination of JRadioButton, JComboBox, JProgressBar, etc. The project is doable with only JFrame. JPanel. JButttton, JLabel, and swing Timer. The rest is up to you if you want to provide a more complex/interesting GUI.

We expect a window (Jframe) with 3 panels (top, middle, and bottom) where top panel has score, level, lives, and time information. The middle part is the game area (ball, stellar objects, paddle), and the bottom panel has one button called Play/Pause.

The diagram below can serve you as an example of the hierarchy and layout for your own game development. This approach helps you isolate the implementation of rather exclusive parts in different panels and make each panel function for one task only. You can make your own design choices as you see fit, while we encourage you to make use of multiple JPanels and layout structures inside the JFrame. You can use default BorderLayout to organize panels in this format.

The description above can serve you as an example of the hierarchy and layout for your own game development. This approach helps you isolate the implementation of rather exclusive parts in different panels and make each panel function for one task only. You can make your own design choices as you see fit, while we encourage you to make use of multiple JPanels and layout structures inside the JFrame.

Game Rules:

Game Window: The game window size is 1024 x 768. It will contain three panels. The middle panel (largest one) is the game area (animation).

Game Time: 1 minutes. If the Game time is over and the player has “lives”, then the game level is incremented by 1, the ball initial speed is incremented by 50%, The game stops if the “lives” becomes zero.

Play/Pause button: The game is initially is in the pause mode. When the button is pressed, it switches to Play mode where the ball starts moving. The player can manipulate the paddle. If the button is pressed again it goes to Pause mode and so on/

Paddle: Paddle is rectangular area with a height 10, width 120. Initially is at the bottom/middle position. When the left key is pressed it moves left 30 pixels, when the right key is pressed it moves 30 pixels right. It's color is black.

Ball: When the game starts and in pause mode, the ball wil be reset to its initial position (and velocity). In the play mode, it will move. If the ball hits the panel it bounces, and the player's score is incremented. Otherwise, if it ball touches the bottom, the player loses one live, and the ball is reset its initial position.
Ball specs: Start position: 10,10, Initial velocity: 4, 1, Ball radius: 10. Ball color: red

Stellar Objects: They will be positioned randomly (without intersection) and they are fixed. We'll provide default images, you use other images (of similar size). If the ball intersects (collides) with the image box (height, and width of the image), the action of the stellar object will be performed. Note that, when the ball intersects the stellar object, the action will be done only once. While the ball still moving within the object, it will not be considered a collision. After the ball moves away from the object, it might collide again (triggering an action again).

Animation: Use the swing timer to repaint the GUI and to update the moving objects. The timer event will be generated at every 20 millliseconds.

Some resources
Ch26 and and Ch27 of the textbook.
See the bouncing ball animation example in the lecture notes.
Java Swing tutorial https://docs.oracle.com/javase/tutorial/uiswing/
A Youtube video for a similar game- Develop a Brick Breaker Game
Free Fall and the Acceleration of Gravity
https://www.physicsclassroom.com/class/1DKin/Lesson-5/How-Fast-and-How-Far

You can share any usefuk resource you find in the discussion forum. We repeat – You can access these resources to learn NOT TO COPY CODE. You need to type your own code.
Stay tuned for more information