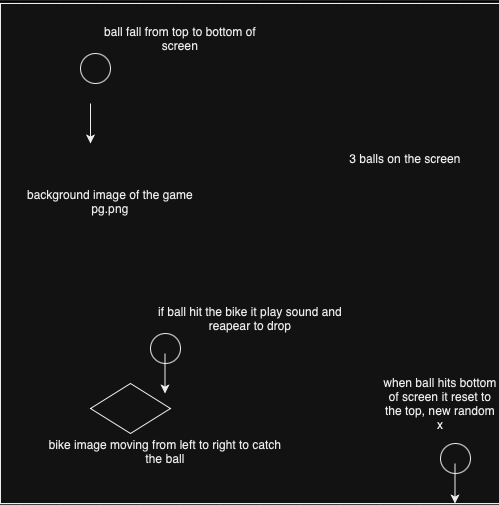
Catch the Ball Part II

Overview

“Catch the ball” is a basic 2d arcade game to demonstrate the overall flow of a game using python and simpleGE. Part II is a continuation of the original game with the addition of an introductory scene.

The Game premise: the game consists of player (player\_0.png), which appears near the bottom of the gameplay screen with a background image (storyboard.png). The user can move player sideways with the arrow keys. A series of balls (soccerBall.png) will be dropping from the top of the screen within the storyboard.png. Each ball will fall from a different x position, and at a different speed between 3 and 8 pixels per frame straight down. If ball touches the player, a positive sound effect is played, and the player score increased if a ball leave the bottom screen, or collide with the player, it reset to a new random position at the top of the screen



Updated Algorithm:

**Initialize the Game**

1. Set up the game background.
2. Load images and sounds for the player, balls, background, and effects.
3. Create the necessary game objects:

* Initialize the player object.
* Initialize balls with random speeds and positions.

**Introduction Scene**

1. Display instructions using multilabel.
2. Show the player’s last score.
3. Add buttons for starting and exiting the game.
4. Listen for button clicks or keyboard inputs to either begin the game or quit.

**Game Loop**

1. Move the player left or right with arrow keys.
2. Move each ball down the screen.
3. Check for collisions between the player and each ball.

* If a collision occurs:
  + 1. Play a sound.
    2. Increase the player’s score.
    3. Reset the ball to a new random position.

1. Display a countdown timer.

* End the game when the timer reaches zero.

Pseudocode:

Initialize pygame

Load “soccerBall>png”, “sound.mp3”, “player\_0.png”, “storyboard.png”

Class Ball:

Initialize (scene):

Set image to “soccerBall.png”

Set size to (25, 25)

Call reset()

Load sound “sound.mp3”

Reset ():

Position y at 10

Position x randomly across the width

Set downward movement speed randomly btw 3 and 8

Move():

Update y by downward movement speed if ball is below the screen: call reset()

Class player:

Initialize (scene):

Set image to “player\_0.png”

Set size to (60, 80)

Position at (300, 800)

Set moveSpeed

Move():

If left key pressed:

Move left by moveSpeed

If rright key pressed:

Move right by moveSpeed

Class Game:

Initialize()

Set background to “storyboard.png”

Create a player object

Create a list a Ball objects

GameLoop():

While game is running:

Handle input

Move player based on input

Move each ball

Check for collisions between player and balls

If collision, play sound, reset ball, and update score

Update and display timer

Redraw background, player, balls, and labels

Class Instruction:

Initialize(score):

Set background to "storyboard.png"

Create multilabel for instructions

Create label for last score

Create play and quit buttons

Process():

If play button clicked or up key pressed, start game

If quit button clicked or down key pressed, exit game

Function Main():

Initialize pygame

Set keepGoing to True

While keepGoing:

Create an Instruction object

Start Instruction scene

If play selected, create and start Game object

If quit selected, set keepGoing to False

Main()