# Penalty Kick!!!

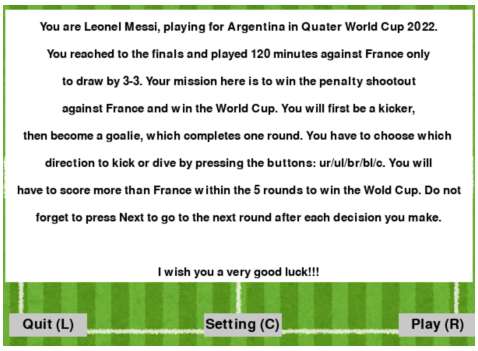
Game Design Document; Taishi Hiraishi

## Overview

"Penalty Kick!!!" will be a basic 2D arcade game made using pygame and simpleGE. The objective of this game is to simulate the finals of the Football World Cup 2022, and the player to undergo the penalty kick shootout as team Argentina against team France, which will be a computer. The player will have five trials each to be a kicker and goalkeeper. When the player is a kicker, players can choose to kick the top right corner, bottom right corner, center, top left corner, and bottom right corner. The computer will be a goalkeeper in this scene randomly picking the direction to dive within the five options. When the player kicks in a direction that the computer didn't choose (dive), the player will score one point. When the player kicks in a direction that the computer chose/dove, the computer saves the ball. After one try, now the player will become the goalkeeper and choose which direction to dive. This process will repeat 5 times, and players will win if they score more than the computer within the five tries, and if not, the computer wins.

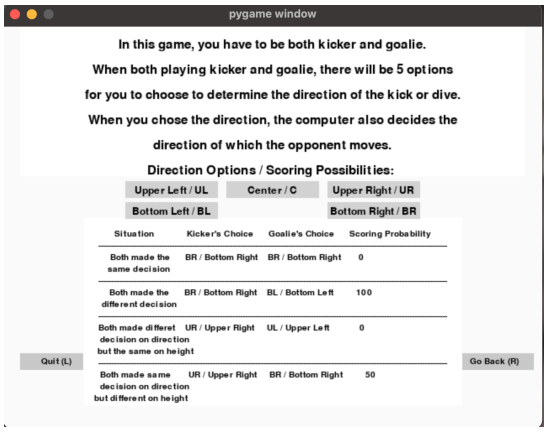
## HOW TO PLAY

1. Run the code
2. Instruction Screen gonna appear:




2.1 If you click quit, the game is going to end

2.2 If you click setting, the setting scene gonna appear → steps are shown in 3




2.3 if you click play, the kickerstate gonna appear → steps are shown in 4

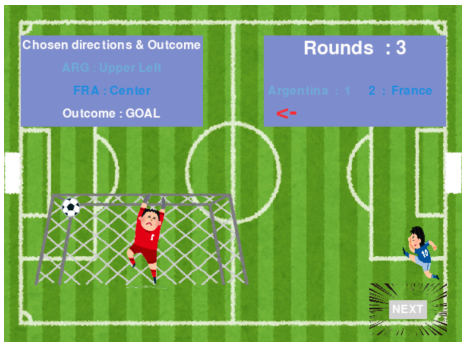
3. In setting scene, you can see the direction options, a direction in which you can make your kicker to kick or your goalie to dive. In addition, you would be able to see a scoring possibility, which shows all the possible outcomes that can happen in each situation.

3.1 If you click quit, the game is going to end

3.2 if you click Go back, the game will take you back to the instruction scene

4. In the kickerstate, you will be a kicker and must choose a direction in which the kicker will kick the ball. There will be five options for you to ch0ose: UR/BR/UL/BL/C

4.1 If you click UR, the ball will move to the goal's upper right corner, and simultaneously, the computer will randomly move the goalkeeper in one of the five directions.




4.2 If you click BR, the ball will move to the goal's bottom right corner, and simultaneously, the computer will randomly move the goalkeeper in one of the five directions.


4.3 If you click UL, the ball will move to the goal's upper left corner, and simultaneously, the computer will randomly move the goalkeeper in one of the five directions.

4.4 If you click BL, the ball will move to the goal's bottom left corner, and simultaneously, the computer will randomly move the goalkeeper in one of the five directions.

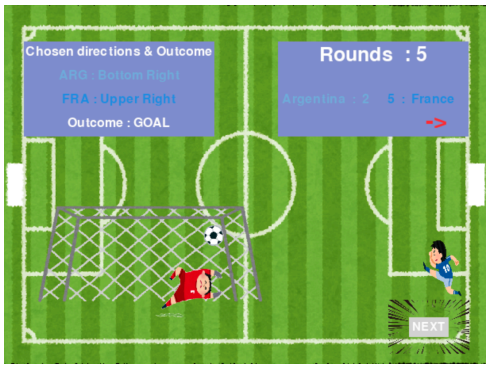

4.5 If you click C, the ball will move to the goal's center, and simultaneously, the computer will randomly move the goalkeeper in one of the five directions.


4.6 After clicking any of the five options, the next button gets activities, and you can click the next button. If you click the next button, the game will take you to goalkeeperState.


5. In the goalkeeperState, you will be a goalkeeper and must choose a direction in which the goalkeeper will dive. There will be five options for you to chose: UR/BR/UL/BL/C

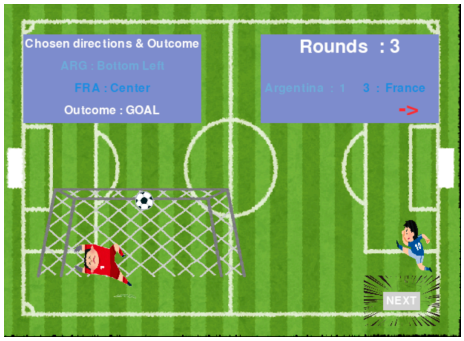
5.1 If you click UR, the goalkeeper will move to the goal's upper right corner, and simultaneously, the computer will randomly move the ball in one of the five directions.


5.2 If you click BR, the goalkeeper will move to the goal's bottom right corner, and simultaneously, the computer will randomly move the ball in one of the five directions.

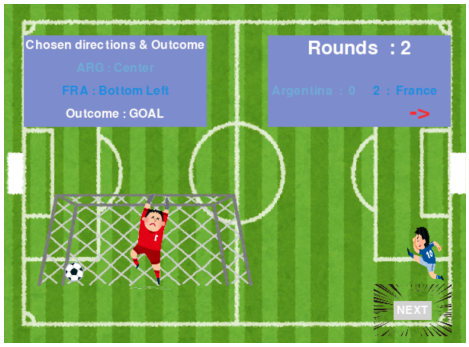



5.3 If you click UL, the goalkeeper will move to the goal's upper left corner, and simultaneously, the computer will randomly move the ball in one of the five directions.

5.4 If you click BL, the goalkeeper will move to the goal's bottom left corner, and simultaneously, the computer will randomly move the ball in one of the five directions.



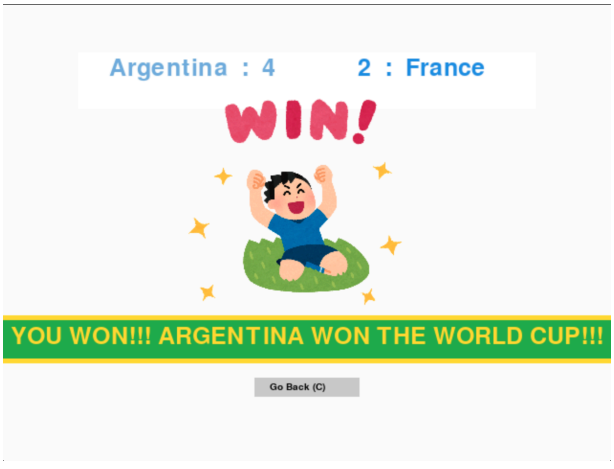

5.5 If you click C, the goalkeeper will move to the goal's center, and simultaneously, the computer will randomly move the ball in one of the five directions.




5.6 After clicking any of the five options, the next button gets activities, and you can click the next button. If you click the next button, the game will take you to kickerState again.


6. One cycle of 4~5 completes 1 round. You will repeat this cycle 5 times by pressing next every time you decide the direction.

6.1 if you score more than the computer after the first 5 rounds, you will be taken to the victory state.



6.1.1 if you press go back, the game will take you back to the instruction state.

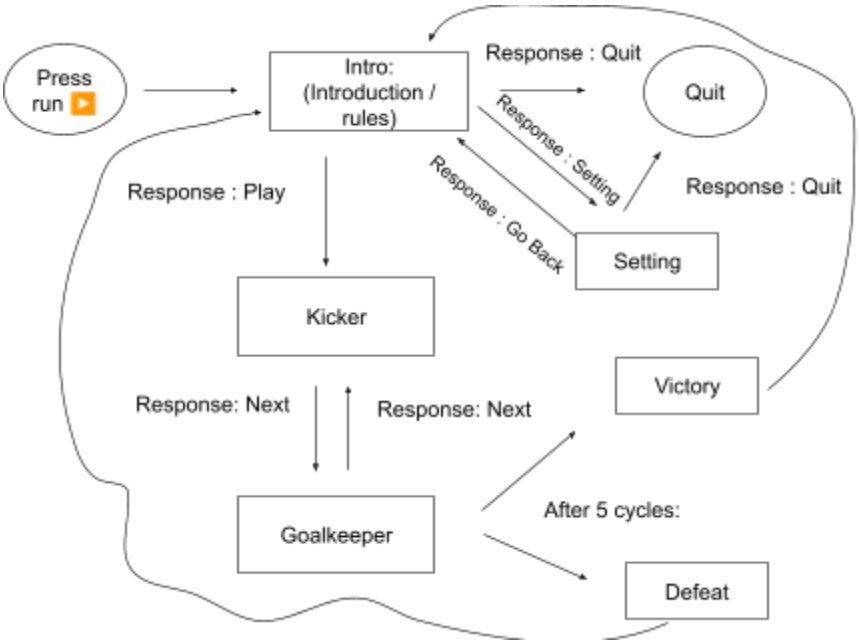
6.2 if you score less than the computer after the first 5 rounds, you will be taken to the defeat state.


6.2.1 if you press go back, the game will take you back to the instruction state.

6.3 If your score is tied to the computer after the first 5 rounds, you will repeat the cycle of going through steps 4~5 until the results are determined and taken to either victory state or defeat state.

The game will have six scenes: **Introduction**, **Kicker, GoalKeeper, Victory, Defeat, Settings**

## State Transition Diagram



The player is initially sent to the Intro scene, which shows instructions & rules and three buttons that ask them if they want to play the game, go to a setting, or quit it. If the user chooses to quit, the game ends. If the user chooses to go to the setting, they are sent to the setting scene where they have two options: Go Back, or Quit if they choose to quit, the game ends, and if they choose to go back, they go back to the intro scene. If the user chooses to play the game, they are sent to the kicker scene.

The kicker scene will end after the user's input on the kicking direction, and the game will be sent to the goalkeeper scene. The goalkeeper scene will also end after the user's input, but this time in a diving direction, and the game will be sent to either a victory or defeat scene, depending on the situation.

In the victory/defeat section, two bottom will ask them to either quit or go back. If the user chooses to quit, the game exits. If the user chooses to go back, the game will be taken to the intro scene.

**Introduction:**

When it begins, the game will show an intro screen with instructions and three buttons. The play button will take the game into the kicker state. The quit button will exit the game.  The setting button will take the game into the setting state.


This scene has four visual elements:

- Instructions
- btnPlay
- btnQuit
- btnSetting

Other attributes:

- Response

The initializer will create all attributes and set up the sprite list

```
init():
  Set image to field.png
  Set response to "Play"
  Create instructions MultiLabel
  Add textLines containing instructions
  Set instructions center to (320, 240)
  Set instructions size to (500, 250)


  Create btnPlay
  Set text to "Play"
  Set center to (550, 400)

  Create btnQuit
  Set text to "Quit"
  Set center to (100, 400)


  Create btnSetting
  Set text to "Play"
  Set center to (300, 400)


  Add lblInstructions, btnSetting, btnQuit, and btnPlay to sprites
```

All event-handling will happen in the scene's process() method

```
process():
  If the quit button is pressed:
     Set response to "Quit"
     Stop the scene

  If the play button is pressed:
     Set the response to "Play"
     Stop the scene

  If the setting button is pressed:
     Set the response to "Setting"
     Stop the scene
```

**Kicker:**

In the kicker's state, the kicker will be at the bottom center of the screen looking towards the goalkeeper, who is located at the top center of the screen facing the kicker. The background will be an image of a soccer field. On the bottom right corner, there will be five buttons showing all the directions in which kickers can kick; top right corner, bottom right corner, center, top left corner, and bottom right corner. On the top right corner, there will be an scoreboard which informs with the current score. After the player's input, the kicker will kick the ball, which will move in the direction that the player chose, and the goalkeeper will dive in a direction that the computer randomly chose. After the animation, the game will automatically take the game in to goalkeeper state.

The Kicker class will have a number of visual attributes:

- **goalPost**
- **kicker(1&2)**
- **goalKeeper**(**UR,UL,BR,BL, CCatch&UR,UL,BR,BL, CGoal**)
- **ball**
- **lblScore**
- **lblRound**
- **btnCenter**
- **btnUpperLeft**
- **btnBottonLeft**
- **btnUpperRight**
- **btnLowerRight**

It will also contain some non-sprite assets:

- **Score**
- **sndBall**
- **sndGoal**
- **sndSave**

The initializer will create all the needed components:

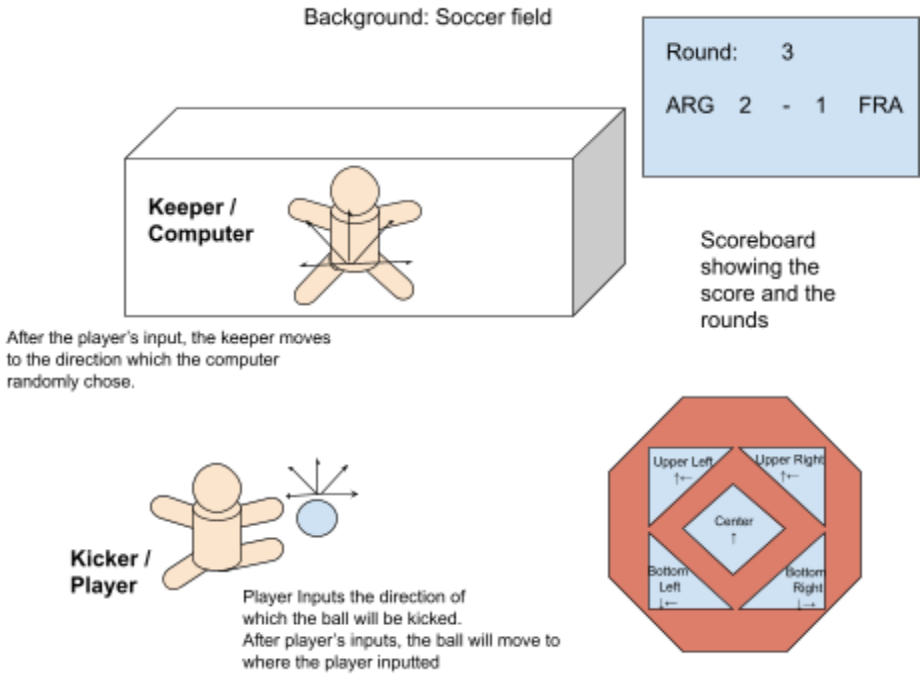
```
init:
  Set image to field.png
  Set score to zero
  Initialize sndBall to ball sound effect
  Initialize sndGoal to goal sound effect
  Initialize sndSave to save sound effect

  Create instance of Kicker1&2 -> kicker1&2
```

```
  Create instance of GoalKeeperUR,UL,BR,BL, CCatch&UR,UL,BR,BL, CGoal
-> goalKeeperUR,UL,BR,BL, CCatch&UR,UL,BR,BL, CGoal
  Create instance of Ball -> ball
  Create instance of GoalPost -> goalPost
  Create instance of LblScore
  Create instance of LblRound

  Create btnCenter
  Set text to "Center ↑"
  Set center to (550, 400)

  Create btnUpperLeft
  Set text to "Upper Left ↑←"
  Set center to (530, 380)


  Create btnUpperRight
  Set text to "Upper Right ↑→"
  Set center to (570, 380)

  Create btnBottomLeft
  Set text to "Bottom Left ↓←"
  Set center to (530, 420)

  Create btnBottomRight
  Set text to "Bottom Right ↓→"
  Set center to (570, 420)

  Add goalPost, kicker1, kicker2,
goalKeeperCatchUR,goalKeeperCatchBR, goalKeeperCatchBL,
goalKeeperCatchUL, goalKeeperCatchC,
goalKeeperGoalUR,goalKeeperGoalBR, goalKeeperGoalBL,
goalKeeperGoalUL, goalKeeperGoalC  ball, lblScore, btnCenter,
btnUpperLeft, btnBottonLeft, btnUpperRight, btnLowerRight, to sprites
```

All event-handling will occur in the scene's process() method:

```
Process:
  If the center button is pressed:
     Set response to "center"
     Play the sound of the ball getting kicked (sndKick)
     Make the ball move to the center
     Make the goalkeeper move randomly
  If the Upper Left button is pressed:
```

```
    Set the response to "upperLeft"
    Play the sound of the ball getting kicked (sndKick)
    Make the ball move to the upper left corner
    Make the goalkeeper move randomly
If the upper right button is pressed:
    Set the response to "upperRight"
    Play the sound of the ball getting kicked (sndKick)
    Make the ball move to the upper right corner
    Make the goalkeeper move randomly
If the bottom left button is pressed:
    Set the response to "bottomLeft”
    Play the sound of the ball getting kicked (sndKick)
    Make the ball move to the bottom left corner
    Make the goalkeeper move randomly
If the bottom right button is pressed:
    Set the response to "bottomRight"
    Play the sound of the ball getting kicked (sndKick)
    Make the ball move to the bottom right
    Make the goalkeeper move randomly
 If a ball collides with the goalKeeper:
    Play ball-goalKeeper collision sound (sndSave)
    Reset that ball
    Move to goalKeeper state
 If a ball does not collide with goalKeeper:
    Play the goal sound (sndGoal)
    Reset that ball
    Add one to the score
    Update lblScore to indicate the new score
    Move to goalKeeper state
```



**GoalKeeper:**

In the goalkeeper's state, the screen will look the same except for the position of the keeper and the goalkeeper's positions; it will get swapped. After the player's input, the kicker will kick the ball, which will move to the direction that the computer randomly chose, and the goalkeeper will dive in the direction that the player chose. After the animation, the game will automatically take the game into the kicker's state. However, after the 5th round, the game will be taken to a victory state or defeat state depending on the situation.

The GoalKeeper class will have the same visual elements, visual attributes, non-sprite assets, initializer, and the event handling process as the kicker class, but there will be some minor differences:

Goal class will have a number of visual attributes:

- **goalPost**
- **kicker(1&2)**
- **goalKeeper**(**UR,UL,BR,BL, CCatch&UR,UL,BR,BL, CGoal**)
- **ball**
- **lblScore**
- **Lbl round**
- **btnCenter**
- **btnUpperLeft**
- **btnBottonLeft**
- **btnUpperRight**
- **btnLowerRight**

It will also contain some non-sprite assets:

- **Score**
- **sndBall**
- **sndGoal**
- **sndSave**

The initializer will create all the needed components:

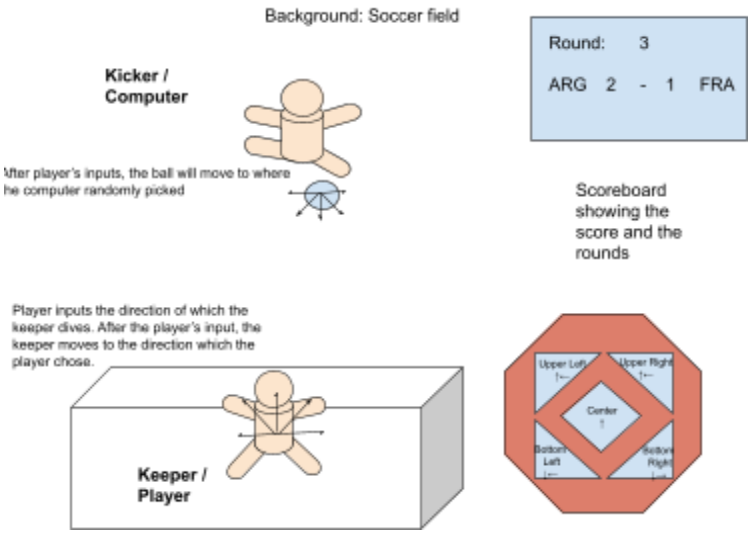
```
init:
  Set image to field.png
  Initialize sndBall to ball sound effect
  Initialize sndGoal to goal sound effect
  Initialize sndSave to save sound effect

  Create instance of Kicker1&2 -> kicker1&2
  Create instance of GoalKeeperUR,UL,BR,BL, CCatch&UR,UL,BR,BL, CGoal
-> goalKeeperUR,UL,BR,BL, CCatch&UR,UL,BR,BL, CGoal
```

```
  Create instance of Ball -> ball
  Create instance of GoalPost -> goalpost
  Create instance of LblScore
  Create instance of LblRound

  Create btnCenter
  Set text to "Center ↑"
  Set center to (550, 400)

  Create btnUpperLeft
  Set text to "Upper Left ↑←"
  Set center to (530, 380)


  Create btnUpperRight
  Set text to "Upper Right ↑→"
  Set center to (570, 380)

  Create btnBottomLeft
  Set text to "Bottom Left ↓←"
  Set center to (530, 420)

  Create btnBottomRight
  Set text to "Bottom Right ↓→"
  Set center to (570, 420)

    Add goalpost, kicker1, kicker2,
goalKeeperCatchUR,goalKeeperCatchBR, goalKeeperCatchBL,
goalKeeperCatchUL, goalKeeperCatchC,
goalKeeperGoalUR,goalKeeperGoalBR, goalKeeperGoalBL,
goalKeeperGoalUL, goalKeeperGoalC  ball, lblScore, btnCenter,
btnUpperLeft, btnBottonLeft, btnUpperRight, btnLowerRight, to sprites
```

All event-handling will occur in the scene's process() method:

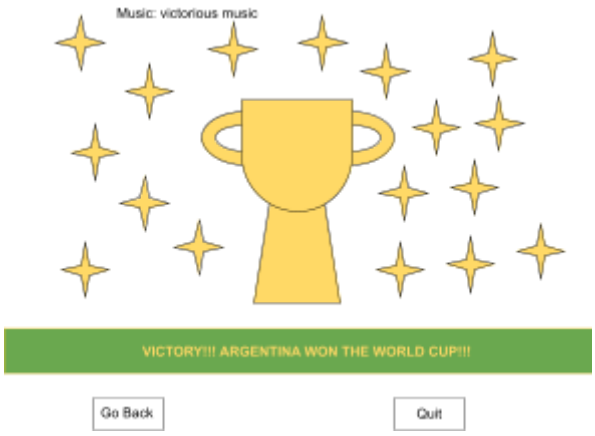
```
Process:
  If the center button is pressed:
     Set response to "center"
     Play the sound of the ball getting kicked (sndKick)
     Make the goalKeeper move to the center
     Make the ball move randomly
  If the Upper Left button is pressed:
     Set the response to "upperLeft"
     Play the sound of the ball getting kicked (sndKick)
```

```
   Make the goalKeeper move to the upper left corner
   Make the ball move randomly
If the upper right button is pressed:
   Set the response to "upperRight"
   Play the sound of the ball getting kicked (sndKick)
   Make the goalKeeper move to the upper right corner
   Make the ball move randomly
If the bottom left button is pressed:
   Set the response to "bottomLeft”
   Play the sound of the ball getting kicked (sndKick)
   Make the goalkeeper move to the bottom left corner
   Make the ball move randomly
If the bottom right button is pressed:
   Set the response to "bottomRight"
   Play the sound of the ball getting kicked (sndKick)
   Make the goalKeeper move to the bottom right
   Make the ball move randomly
 If a ball collides with the goalKeeper:
   Play ball-goalKeeper collision sound (sndSave)
   Reset that ball
   Add one to round
   Update lblround to indicate the new score
   Move to Kicker state

 If a ball does not collide with goalKeeper:
   Play the goal sound (sndGoal)
   Reset that ball
   Add one to the score
   Add one to round
   Update lblScore to indicate the new score
   Move to goalKeeper state
```




**Victory:**

In the victory state, the screen will have a label and a text “Congratulation!!! Argentina won the World Cup!!!” With the image of the trophy in the center and victorious music. The screen will have two buttons in the bottom right corner; Go Back, and Quit. Go Back button will take the game into the introduction state. The quit button will exit the game.




The Victory class will have a number of visual attributes:

- **Victory**
- **lblVictory**
- **btnGoBack**
- **btnQuit**

It will also contain some non-sprite assets:

- **sndVictory**

The initializer will create all attributes and set up the sprite list

```
init():
  Initialize sndVictory
  Create instance of Victory -> victory
  Create victory MultiLabel
  Add text lines containing praising statements
  Set lblvictory center to (320, 240)
  Set lblvictory size to (640, 100)

  Create btnGoBack
  Set text to "Go Back"
  Set center to (550, 400)
  Create btnQuit
  Set text to "Quit"
  Set center to (100, 400)


  Add sndVictoty, lblVictory, victory,  btnQuit, and btnGoback to
sprites
```

All event-handling will happen in the scene's process() method

```
process():
  If the quit button is pressed:
     Set response to "Quit"
     Stop the scene

  If the go back button is pressed:
     Set the response to "GoBack"
     Stop the scene
```

**Defeat:**

In the defeated state, the screen will have a label and the text “Oh no,,, France won the World cup,,” With the image of the sad face emoji in the center and depressing music. The screen will have two buttons in the bottom right corner; Go Back, and Quit. The go Back button will take the game into the introduction state. The quit button will exit the game.


The Defeat class will have a number of visual attributes:

- **Defeat**
- **lblDefeat**
- **btnGoBack**
- **btnQuit**

It will also contain some non-sprite assets:

- **sndDefeat**

The initializer will create all attributes and set up the sprite list

```
init():
  Initialize sndDefeat
  Create instance of Defeat -> defeat
  Create defeat MultiLabel
  Add text lines containing sad statements
  Set lbldefeat center to (320, 240)
  Set lbldefeat size to (640, 100)

  Create btnGoBack
  Set text to "Go Back"
  Set center to (550, 400)

  Create btnQuit
```

```
  Set text to "Quit"
  Set center to (100, 400)


  Add sndVictoty, lblDefeat, defeat,  btnQuit, and btnGoBack to
sprites
```

All event-handling will happen in the scene's process() method

```
process():
  If the quit button is pressed:
     Set response to "Quit"
     Stop the scene

  If the go back button is pressed:
     Set the response to "GoBack"
     Stop the scene
```

**Settings:**

In the setting state, the screen will have a label and the text of which it will explain the scoring probability of each situation that can happen in the game. It will look something like this:

| Situation | Kicker's choice | Goalie's choice | Scoring probability |
|---|---|---|---|
| Both made the same decision | **BR** (Bottom right corner) | **BR** (Bottom right corner) | 0% |
| Both made a different decision | **BR** (Bottom right corner) | **BL** (Bottom left corner) | 100% |
| Both made different decisions on direction but the same on height | **UR** (Top right corner) | **UL** (Top left corner) | 0% |
| Both made the same decision on direction but different on height | **UR** (Top right corner) | **BR** (Bottom right corner) | 50% |

Go Back

Quit

The Setting class will have a number of visual attributes:

- **Setting**
- **btnGoBack**
- **btnQuit**

The initializer will create all attributes and set up the sprite list

```
init():
  Create instance of Setting -> setting
  Create btnGoBack
  Set text to "Go Back"
  Set center to (550, 400)

  Create btnQuit
  Set text to "Quit"
  Set center to (100, 400)

  Add Setting,  btnQuit, and btnGoBack to sprites
```

All event-handling will happen in the scene's process() method

```
process():
  If the quit button is pressed:
     Set response to "Quit"
     Stop the scene

  If the go back button is pressed:
     Set the response to "GoBack"
     Stop the scene
```

# The main() function

The main function will manage the high-level state transition between intro and play states. It is a very standard main loop, containing four variables:

**instructions**
**setting**
**kicker**

**goalKeeper**
**victory**
**defeat**
**keepGoing**

Psuedocode for main

```
main():
  Import random
  Set keepGoing to true
  Set score (both argentina and France) to zero
  Set round to 0
  While keepGoing is true:
     Create an instance of Instructions -> instructions
     Start instructions
     When instructions ends,
     If instructions.response is "Play":
     Create an instance of kicker -> kicker
     Start kicker
          If round == 6:
               & score arg>fra
                    Go to Victory
               & score fra>arg
                    Go to Defeat
     If instructions.response is "Setting":
     Create an instance of Setting -> setting
     If instructions.response is "Quit":
     Set keepGoing to False, which will exit the game
```

# Componentes of each classes:

## Kicker class():

### Kicker1:

```
init:
  Set image to kicker1.png
  Set size to 50x70
  Set position to (500, 400)
  Set choice Image to kicker2.png
```

All event-handling will be in process() method
Change to  kicker2.png after any key is pressed

```
process:
  If up key is pressed
     If right key is pressed (or btnUpperRight is pressed)
     Change to kicker2.png
  If up key is pressed
     If left key is pressed (or btnUpperLeft is pressed)
     Change to kicker2.png
  If bottom key is pressed
     If right key is pressed (or btnBottomRight is pressed)
     Change to kicker2.png
  If bottom key is pressed
     If left key is pressed (or btnBottomLeft is pressed)
     Change to kicker2.png
 If up key is pressed (or btnCenter is pressed)
     Change to kicker2.png
```

## Goal Keeper:

```
init:
  Set image to goalKeeper.png
  Set size to 70x50
  Set position to (150, 400)
```

All event-handling will be in process() method

Make the goalkeeper randomly choose which direction to dive after any key is pressed

```
Process:
  Create a variable called directions and give it a random
number between 1 to 100
direction = random.randint(1, 100)
  Create a variable called catchRate and give it a random
number between 1 to 100
catchRate= random.randint(1, 100)
  If up key is pressed
     If right key is pressed (or btnUPRight is pressed)
          If direction is between  1~20:
               Make the goal keeper stay in the center
(150,450)
               Change to goalKeeperGoalC.png
          elIf direction is between  21~40:
```

```
                Make the goalkeeper go to upper right (200,500)
                Change to goalKeeperCatchUR.png
           Elif direction is between 41~60:
                Make the goalkeeper move to bottom
right(200,450)
                If catchRate is between 1 to 50
                     Change to goalKeeperCatchBR.png
                Elif catchRate is between 51 to 100
                     Change to goalKeeperGoalBR.png
           Elif direction is between 61~80:
                Make the goalKeeper move to upper left
(100,500)
                Change to goalKeeperGoalUL.png
           Elif direction is between 81~100:
                Make the goalKeeper move to bottom left
(100,450)
                Change to goalKeeperGoalBL.png
     elif left key is pressed (or btnUpperLeft is pressed)
           If direction is between  1~20:
                Make the goal keeper stay in the center
(150,450)
                Change to goalKeeperGoalC.png
           elIf direction is between  21~40:
                Make the goalkeeper go to upper right (200,500)
                Change to goalKeeperGoalUR.png
           Elif direction is between 41~60:
                Make the goalkeeper move to bottom
right(200,450)
                Change to goalKeeperGoalBR
           Elif direction is between 61~80:
                Make the goalKeeper move to upper left
(100,500)
                Change to goalKeeperCatchUL.png
           Elif direction is between 81~100:
                Make the goalKeeper move to bottom left
(100,450)
                If catchRate is between 1 to 50
                     Change to goalKeeperCatchBL.png
                Elif catchRate is between 51 to 100
                     Change to goalKeeperGoalBL.png
     elif W key is pressed (or btnCenter is pressed)
           If direction is between  1~20:
                Make the goalkeeper stay in the center
(150,450)
                Change to goalKeeperCatchC.png
```

```
          elIf direction is between  21~40:
               Make the goalkeeper go to upper right (200,500)
               Change to goalKeeperGoalUR.png
          Elif direction is between 41~60:
               Make the goalkeeper move to bottom
right(200,450)
               Change to goalKeeperGoalBR
          Elif direction is between 61~80:
               Make the goalKeeper move to upper left
(100,500)
               Change to goalKeeperGoalUL.png
          Elif direction is between 81~100:
               Make the goalKeeper move to bottom left
(100,450)
               Change to goalKeeperGoalBL.png
  elif bottom key is pressed
     If right key is pressed (or btnBottomRight is pressed)
          If direction is between  1~20:
               Make the goal keeper stay in the center
(150,450)
               Change to goalKeeperGoalC.png
          elIf direction is between  21~40:
               Make the goalkeeper go to upper right (200,500)
               If catchRate is between 1 to 50
                    Change to goalKeeperCatchBR.png
               Elif catchRate is between 51 to 100
                    Change to goalKeeperGoalBR.png
          Elif direction is between 41~60:
               Make the goalkeeper move to bottom
right(200,450)
               Change to goalKeeperCatchBR.png
          Elif direction is between 61~80:
               Make the goalKeeper move to upper left
(100,500)
               Change to goalKeeperGoalUL.png
          Elif direction is between 81~100:
               Make the goalKeeper move to bottom left
(100,450)
               Change to goalKeeperGoalBL.png
     elIf left key is pressed (or btnBottomLeft is pressed)
          If direction is between  1~20:
               Make the goal keeper stay in the center
(150,450)
               Change to goalKeeperGoalC.png
          elIf direction is between  21~40:
```

```
                Make the goalkeeper go to upper right (200,500)
                Change to goalKeeperGoalUR.png
           Elif direction is between 41~60:
                Make the goalkeeper move to bottom
right(200,450)
                Change to goalKeeperGoalBR
           Elif direction is between 61~80:
                Make the goalKeeper move to upper left
(100,500)
                If catchRate is between 1 to 50
                     Change to goalKeeperCatchUL.png
                Elif catchRate is between 51 to 100
                     Change to goalKeeperGoalUL.png
           Elif direction is between 81~100:
                Make the goalKeeper move to bottom left
(100,450)
                Change to goalKeeperCatchBL.png
```

## Ball:

```
init:
  Set image to ball.png
  Set size to 20x20
  Set position to (420, 420)
```

All event-handling will be in process() method

Make the goalkeeper randomly choose which direction to dive after any key is pressed

```
Process:
   If up key is pressed
     If right key is pressed (or btnUPRight is pressed)
          Make the ball go to upper right (200,500)
     elif left key is pressed (or btnUpperLeft is pressed)
          Make the ball move to upper left (100,500)
     elif W key is pressed (or btnCenter is pressed)
          Make the ball go to the center (150,450)
  elif bottom key is pressed
     If right key is pressed (or btnBottomRight is pressed)
          Make the ball move to bottom right(200,450)
     elIf left key is pressed (or btnBottomLeft is pressed)
          Make the ball move to bottom left (100,450)
```

## Goal Post:

```
init:

  Set image to GoalPost.png
  Set size to 100x40
  Set position to (100, 400)
```

## GoalKeeper class():

## Kicker1:

```
init:
  Set image to kicker1.png
  Set size to 45x70
  Set position to (500, 400)
  Set choice Image to kicker2.png
```

All event-handling will be in process() method
Change to kicker2.png after any key is pressed

```
process:
  If up key is pressed
      If right key is pressed (or btnUpperRight is pressed)
      Change to kicker2.png
  If up key is pressed
      If left key is pressed (or btnUpperLeft is pressed)
      Change to kicker2.png
  If bottom key is pressed
      If right key is pressed (or btnBottomRight is pressed)
      Change to kicker2.png
  If bottom key is pressed
      If left key is pressed (or btnBottomLeft is pressed)
      Change to kicker2.png
 If up key is pressed (or btnCenter is pressed)
      Change to kicker2.png
```

## Goal Keeper:

```
init:
  Set image to goalKeeper.png
  Set size to 50x70
  Set position to (150, 400)
```

All event-handling will be in process() method

Make the goalkeeper move dive in a direction which user choses:

```
Process:
   If up key is pressed
      If right key is pressed (or btnUPRight is pressed)
           Make the goalKeeper go to upper right (200,500)
      elif left key is pressed (or btnUpperLeft is pressed)
           Make the goalKeeper move to upper left (100,500)
      elif W key is pressed (or btnCenter is pressed)
           Make the goalKeeper go to the center (150,450)
  elif bottom key is pressed
      If right key is pressed (or btnBottomRight is pressed)
           Make the goalKeeper move to bottom right(200,450)
      elIf left key is pressed (or btnBottomLeft is pressed)
           Make the goalKeeper move to bottom left (100,450)
```

## Ball:

```
init:
  Set image to ball.png
  Set size to 20x20
  Set position to (420, 420)
```

All event-handling will be in process() method

Make the ball move randomly

```
Process:

  Create a variable called directions and give it a random
number between 1 to 100
direction = random.randint(1, 100)
  Create a variable called catchRate and give it a random
number between 1 to 100
catchRate= random.randint(1, 100)
  If up key is pressed
      If right key is pressed (or btnUPRight is pressed)
           If direction is between  1~20:
                Make the ball go to the center (150,450)
                Change to goalKeeperGoalC.png
           elIf direction is between  21~40:
```

```
            Make the ball go to upper right (200,500)
            Change to goalKeeperCatchUR.png
      Elif direction is between 41~60:
            Make the ball move to bottom right(200,450)
            If catchRate is between 1 to 50
                  Change to goalKeeperCatchBR.png
            Elif catchRate is between 51 to 100
                  Change to goalKeeperGoalBR.png
      Elif direction is between 61~80:
            Make the ball move to upper left (100,500)
            Change to goalKeeperGoalUL.png
      Elif direction is between 81~100:
            Make the ball move to bottom left (100,450)
            Change to goalKeeperGoalBL.png
elif left key is pressed (or btnUpperLeft is pressed)
      If direction is between  1~20:
            Make the ball go to the center (150,450)
            Change to goalKeeperGoalC.png
      elIf direction is between  21~40:
            Make the ball go to upper right (200,500)
            Change to goalKeeperGoalUR.png
      Elif direction is between 41~60:
            Make the ball move to bottom right(200,450)
            Change to goalKeeperGoalBR
      Elif direction is between 61~80:
            Make the ball move to upper left (100,500)
            Change to goalKeeperCatchUL.png
      Elif direction is between 81~100:
            Make the ball move to bottom left (100,450)
            If catchRate is between 1 to 50
                  Change to goalKeeperCatchBL.png
            Elif catchRate is between 51 to 100
                  Change to goalKeeperGoalBL.png
elif W key is pressed (or btnCenter is pressed)
      If direction is between  1~20:
            Make the ball move to the center (150,450)
            Change to goalKeeperCatchC.png
      elIf direction is between  21~40:
            Make the ball move to upper right (200,500)
            Change to goalKeeperGoalUR.png
      Elif direction is between 41~60:
            Make the ball move to bottom right(200,450)
            Change to goalKeeperGoalBR
      Elif direction is between 61~80:
            Make the ball move to upper left (100,500)
```

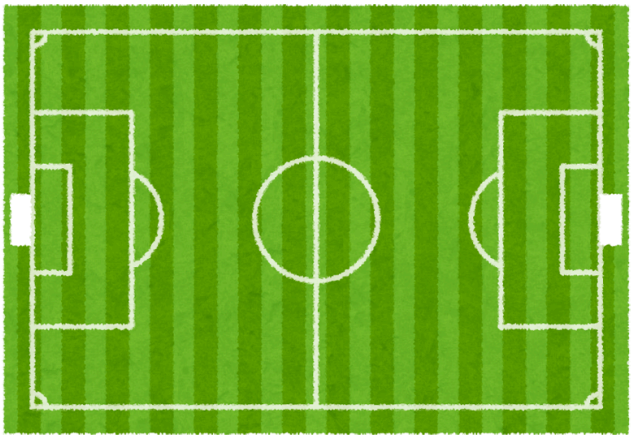
```
                Change to goalKeeperGoalUL.png
           Elif direction is between 81~100:
                Make the ball move to bottom left (100,450)
                Change to goalKeeperGoalBL.png
elif bottom key is pressed
   If right key is pressed (or btnBottomRight is pressed)
           If direction is between  1~20:
                Make the ball stay in the center (150,450)
                Change to goalKeeperGoalC.png
           elIf direction is between  21~40:
                Make the ball go to upper right (200,500)
                If catchRate is between 1 to 50
                     Change to goalKeeperCatchBR.png
                Elif catchRate is between 51 to 100
                     Change to goalKeeperGoalBR.png
           Elif direction is between 41~60:
                Make the ball move to bottom right(200,450)
                Change to goalKeeperCatchBR.png
           Elif direction is between 61~80:
                Make the ball move to upper left (100,500)
                Change to goalKeeperGoalUL.png
           Elif direction is between 81~100:
                Make the ball move to bottom left (100,450)
                Change to goalKeeperGoalBL.png
   elIf left key is pressed (or btnBottomLeft is pressed)
           If direction is between  1~20:
                Make the ball move to the center (150,450)
                Change to goalKeeperGoalC.png
           elIf direction is between  21~40:
                Make the ball move to upper right (200,500)
                Change to goalKeeperGoalUR.png
           Elif direction is between 41~60:
                Make the ball move to bottom right(200,450)
                Change to goalKeeperGoalBR
           Elif direction is between 61~80:
                Make the ball move to upper left (100,500)
                If catchRate is between 1 to 50
                     Change to goalKeeperCatchUL.png
                Elif catchRate is between 51 to 100
                     Change to goalKeeperGoalUL.png
           Elif direction is between 81~100:
                Make the ball move to bottom left (100,450)
                Change to goalKeeperCatchBL.png
```

Goal Post:

```
init:

  Set image to GoalPost.png
  Set size to 100x40
  Set position to (100, 400)
```

**Asset - Plan**

Background: Field.png



Kicker1: Kicker1.png

Kicker2:Kicker2.png

Goal Keeper: Goalkeeper.png

Goal Keeper Up Right Save: goalKeeperCatchUR.png

Goal Keeper Up Right Goal: goalKeeperGoalUR .png

Goal Keeper Up Left Save: goalKeeperCatchUL.png

Goal Keeper Up Left Goal: goalKeeperGoalUL.png


Goal Keeper Bottom Right Save: goalKeeperCatchBR.png

Goal Keeper Bottom Right Goal: goalKeeperGoalBR.png


Goal Keeper Bottom Left Save: goalKeeperCatchBL.png

Goal Keeper Bottom Left Goal: goalKeeperGoalBL.png


Goal Keeper Center Save: goalKeeperCatchC.png

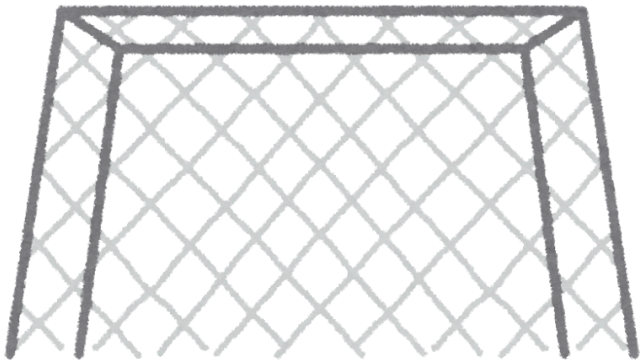
Goal Keeper Center Goal:goalKeeperGoalC.png


Ball: ball.png

Victory: victory.png


Defeat: defeat.png


Goalpost: goalpost.png



Boom: boom1.png

Every image is fair use from the Japanese copyright-free illustration website “Irasutoya” : https://www.irasutoya.com/

sndGoal

goal.mp3

sndSave

save.mp3

sndBall:

ball.mp3

Above 3 audios are fair use from the copy-right free audio website ARSPARK: https://arspark.jp/material/

sndVictory:

victory.mp3

sndDefeat

Defeat.mp3

Above 2 audios are fair use from the copy-right free audio website Pixabay: https://pixabay.com/