

FINAL Report

OXO GAME



May 10, 2019

by BHEKANANI cELE & MATTHEW wEPPENAAR

TABLE OF CONTENT

[Introduction 2](#_Toc8543662)

[Project Management 2](#_Toc8543663)

[Project Plan 2](#_Toc8543664)

[Teamwork 2](#_Toc8543665)

[Design 3](#_Toc8543666)

[Messaging Protocol 3](#_Toc8543667)

[Implementation 6](#_Toc8543668)

[Functionality 7](#_Toc8543669)

[Testing 8](#_Toc8543670)

[User evaluation 8](#_Toc8543671)

[Report on their experience: 8](#_Toc8543672)

[Text based client 8](#_Toc8543673)

[GUI Prototype 8](#_Toc8543674)

[System Correctness 9](#_Toc8543675)

[Scenarios 9](#_Toc8543676)

[Playing where opponent already played 9](#_Toc8543677)

[Restarting 10](#_Toc8543678)

[Placing a move “outside” of the board 10](#_Toc8543679)

[Enhancements & Future work 10](#_Toc8543680)

[Enhancements 10](#_Toc8543681)

[User-experience 10](#_Toc8543682)

[Aesthetic 11](#_Toc8543683)

[Future Work 11](#_Toc8543684)

[Conclusion 12](#_Toc8543685)

[Appendices i](#_Toc8543686)

[OXOGame.py i](#_Toc8543687)

[Game\_Over.py xviii](#_Toc8543688)

[check\_running\_program.py xx](#_Toc8543689)

# Introduction

In early April Ten Eleven games approached us with a request: Create a unique innovative version of the classic game OXO. A daunting task. Not only did we have to improve on a well working formula, but also create and design a program on the scale we had never done before.

This report will serve as documentation of the journey we set upon whilst creating this game. From how we divided duties in the beginning to the final product and all the stops in between (design, user-testing, etc.). The end goal was to create a fully functional, enhanced version of OXO, that could be played over the network.

# Project Management

## Project Plan

We submitted progress weekly according to the schedule below. These progress reports took the form of deliverables (assignments).

[schedule from assignment 6]

## Teamwork

Because Python is Bhekanani’s second language, he volunteered to do some of the more challenging code, while Matthew handled basic code associated with the program and general design. We consulted each other regularly with suggestions and advice on the tasks we were handling. We also created a repository on GitHub, which enabled us to remotely edit code and then upload it to the repository. This streamlined the process of editing and completing the code.

# Design

## Messaging Protocol

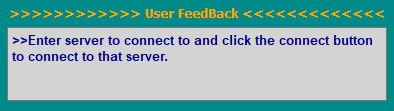
we put this on our GUI to show the user events in game.

Figure : Picture of message-box

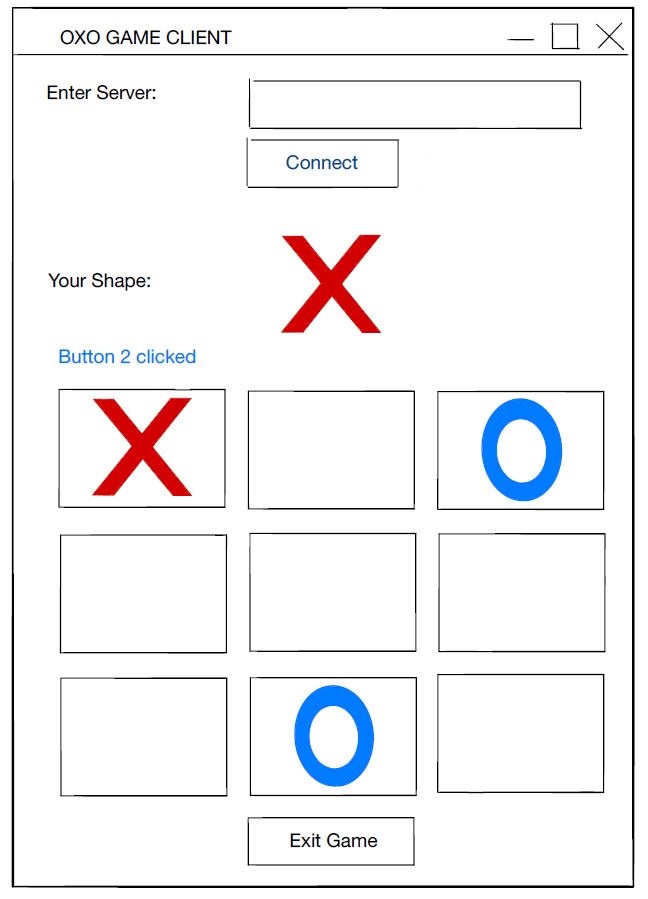
We decided on a game where players would click on blocks to place their moves.

Figure : Picture of low fidelity

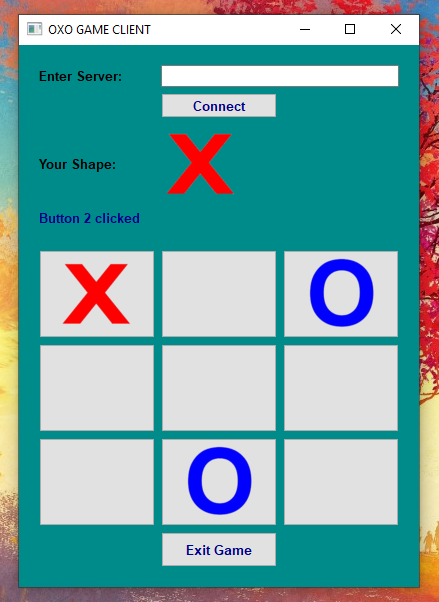
This is the GUI based on the sketch.

Figure : Picture of high fidelity

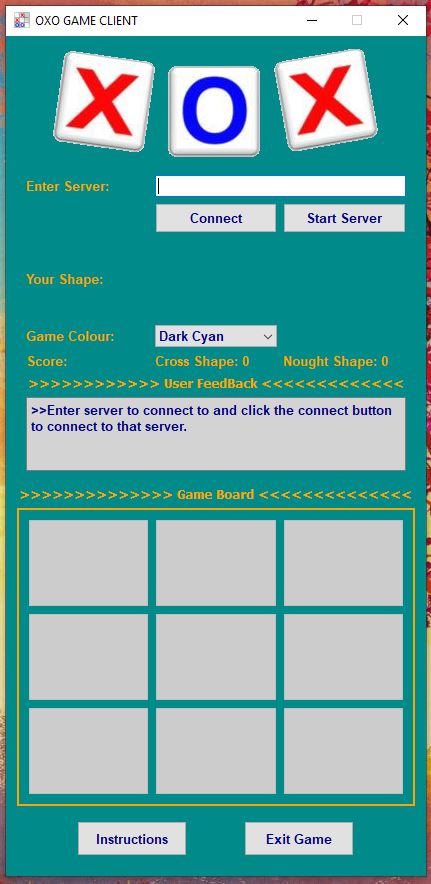


Figure : Picture of final user interface

This final design has all the visible enhancements we made to the program

# Implementation

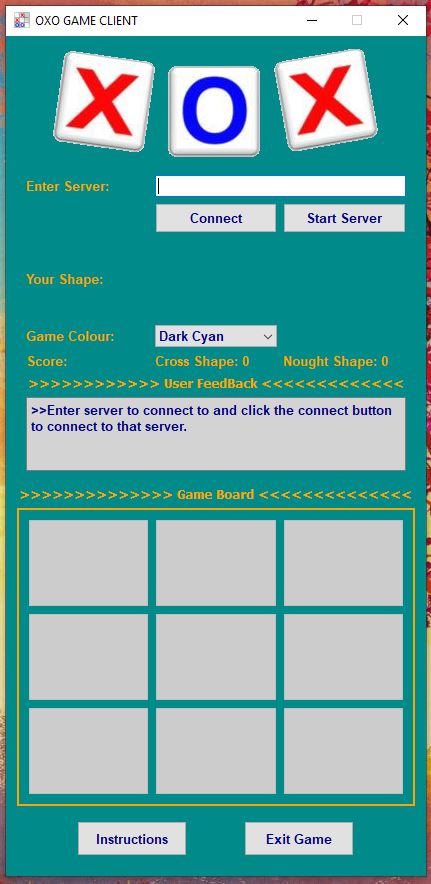
Our final program had the following design

Figure : Picture of final design

## Functionality

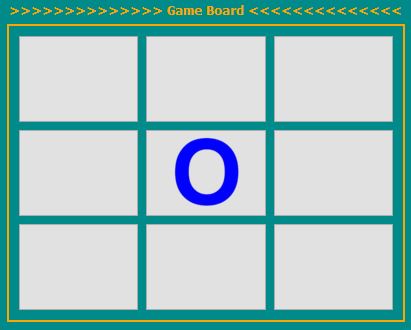


Figure : Picture of move being made after click

We decided to have players click onto open spaces to place their moves, rather than entering a number in a line edit.

Figure : Picture of pop-up dialog

Once an end condition has been met (win, lose or Tie) a pop-up dialog appear requesting if the player wants to restart. It will restart if both players agree to play again and close otherwise.

Figure : Picture of pop-up dialog

Once an end condition has been met (win, lose or Tie) an appropriate window appears to each player according to end condition

# Testing

## User evaluation

The following aspects of the game were given to testers: Text Client, showed the functionality of the game and the High Fidelity GUI, to show how the final product might look like.

## Report on their experience:

### Text based client

The consensus on the usability and expected operation of the program is one of satisfaction. That is to say that all testers either agreed or strongly agreed on these aspects of the program. Some notes provided, suggest that we could be a bit clearer on what position numbers of the board and that we could give a more explicit message (to the user) when the opponent makes an incorrect move.

### GUI Prototype

The consensus on the layout and the aesthetics of the GUI is on of satisfaction. That is to say that all testers either agreed or strongly agreed on these aspects of the GUI. There is a request for a “Text Edit” that shows server messages. It was also suggested that we remove the “Play” button from the GUI, so that the game starts immediately, thus streamlining the game playing process.

# System Correctness

Considering the suggestions from the user feedback, we made the following changes: Added a ‘message-box’ to the GUI that displayed events during the game. Improved messages sent to player based on events occurring, specifically when an opponent makes an incorrect move. We also made board placements clearer in our Text-Client.

We asked testers to play multiple rounds of the game, with the aim of breaking it. We had two ways of obtaining this. We created some scenarios for the players, and we let testers just play with no guidelines in order to generate problems naturally through gameplay, with the only instructions being how to place moves

## Scenarios

### Playing where opponent already played

We wanted to see if the game will allow the player to play on an already filled space. The system prevented this from occurring and sent the player an ‘invalid move’ message and allowed them to play again.

### Restarting

We wanted what happens at the end of the game when players decided if they wanted to replay. If both players wanted a rematch and clicked ‘yes’ then the game would restart. Else the game would close. The game responded as expected.

### Placing a move “outside” of the board

The Text-Client allowed the player to enter a number (0-8) to place a move. We wanted to see what happened when a player entered ‘9’ for example. The game responded by sending the player an ‘invalid move’ message and allowed them to play again.

No problems arose from natural play. No functional issues arose from the GUI

# Enhancements & Future work

## Enhancements

We made the following user-experience and aesthetic enhancements

### User-experience

Added an “Instructions” button to the GUI that would display a pop-up that gives instructions on how connect to the server and how to play the game. [Bhekanani]

Made the game installable on the PC. [Bhekanani]

Made it possible to start the game server from within the game. [Bhekanani]

Disabled Buttons when it is the opponent’s turn. [Bhekanani]

### Aesthetic

Added a ‘Please Wait’ Gif in between moves. and when the both players connect to the server. [Bhekanani]

Added pop-up window with Gif Effects at the end of a match to each player that shows whether player won or lost or played a draw (Tie). [Matthew]

Give the user to change the theme of the GUI. [Bhekanani]

Added tips on buttons and Line Edit when user hovered over them, to give a description on what the button or line edit is for. [Bhekanani]

Added Game Sound Effect at the end of the match. [Bhekanani]

## Future Work

Make the game with a bigger board where the player wins with 4 cross or noughts.

Make the game run faster.

# Conclusion

This game was the biggest thing we have ever done. From start to finish, it was incredibly challenging. The tasks were split up in accordance with our strengths and talents. We worked together to obtain our final vision of the OXO game, using GitHub to streamline the process.

The user feedback helped us sculpt the final product. We believe that the enhancements we made both aesthetically and functionally will give players a better experience whilst playing our game. We could improve the speed of our game in the future.

Finally, we would like to thank Ten Eleven Games for giving us the opportunity to make this game as it has truly been an enriching experience.

# Appendices

## OXOGame.py

1. # OXO GAME
2. # Bhekanani Cele & Matthew Weppenaar
3. # 08 April 2019
5. import os
6. import sys
7. from PyQt5.QtCore import\*
8. from PyQt5.QtWidgets import\*
9. from PyQt5.QtGui import\*
10. from GameClient import \*
11. from Game\_Over import\*  #"enhancement" - Game Over windows
12. from check\_running\_program import\*
14. class Instruction\_Window(QWidget):
15. def \_\_init\_\_(self, parent=None):
16. QWidget.\_\_init\_\_(self, parent)
17. self.setGeometry(300, 40, 1280, 720)
18. self.setFixedSize(1280, 720)
19. self.setWindowFlags(Qt.WindowCloseButtonHint | Qt.WindowMinimizeButtonHint) #"enhancement"- disabling maximise window flag
20. self.setWindowIcon(QIcon('icon.jpg')) #window  icon
21. self.setWindowTitle('Instructions')  # window title
23. self.setPalette(QPalette(QColor("darkgray"))) #setting game window colour
25. self.instructions\_label=QLabel() #creating label
26. self.instructions\_picture=QPixmap("Instructions.png")
27. self.instructions\_label.setPixmap(self.instructions\_picture)
29. vbox=QVBoxLayout() #creates horizontal box layout
30. vbox.addWidget(self.instructions\_label) #adding widget to the hozintal box layout
31. self.setLayout(vbox)
33. # a slots called when "Instructions" push button is clicked
34. def instructions\_clicked(self):
35. self.show()
37. class OXO\_game(QWidget,GameClient):
38. def \_\_init\_\_(self,parent=None):  #constructor
39. QWidget.\_\_init\_\_(self,parent)
40. GameClient.\_\_init\_\_(self)
41. self.setGeometry(500,40,420,820)
42. self.setFixedSize(420,840)
43. self.setWindowFlags(Qt.WindowCloseButtonHint | Qt.WindowMinimizeButtonHint) #"enhancement"- disabling maximise window flag
44. self.setWindowTitle("OXO GAME CLIENT") # window title
45. self.setWindowIcon(QIcon('icon.jpg')) #window  icon
46. self.server=QLabel("Enter Server:")#creating label
47. #formating label
48. self.server.setFont(QFont("Arial",10,weight=QFont.Bold))
50. self.server\_input=QLineEdit()#creating QLine edit for entering server to connect to
51. self.server\_input.setToolTip("Enter Server IP Address") #"enhancement"- set tool tip
52. #formating QLine edit
53. self.server\_input.setStyleSheet('color: red')
54. self.server\_input.setFont(QFont("Arial",10,weight=QFont.Bold))
56. self.connect=QPushButton("Connect")#creating QPushButton
57. #formating button
58. self.connect.setFont(QFont("Arial",10,weight=QFont.Bold))
59. self.connect.setMinimumHeight(25)
60. self.connect.setStyleSheet("color: darkblue")
61. self.connect.setMinimumHeight(30)
62. self.connect.setMaximumWidth(200)
63. self.connect.setToolTip("Connect To The Server") #"enhancement"- set tool tip
65. self.start\_server=QPushButton("Start Server")#creating QPushButton
66. #formating button
67. self.start\_server.setToolTip("Start Server To Connect To")
68. self.start\_server.setFont(QFont("Arial",10,weight=QFont.Bold))
69. self.start\_server.setMinimumHeight(25)
70. self.start\_server.setStyleSheet("color: darkblue")
71. self.start\_server.setMinimumHeight(30)
72. self.start\_server.setMaximumWidth(200)
74. self.mute=QPixmap("mute.png") #mute image
75. self.un\_mute=QPixmap("un\_mute.png") #mute image
77. self.your\_shape\_label=QLabel("Your Shape:")#creating label
78. #formating label
79. self.your\_shape\_label.setFont(QFont("Arial",10,weight=QFont.Bold))
81. self.exit=QPushButton("Exit Game")#creating QPushButton
82. #formating button
83. self.exit.setStyleSheet('color: darkblue')
84. self.exit.setFont(QFont("Arial",10,weight=QFont.Bold))
85. self.exit.setMinimumHeight(35)
86. self.exit.setMaximumWidth(110)
87. self.exit.setToolTip("Close Game") #"enhancement"- set tool tip
89. self.instructions=QPushButton("Instructions")#creating QPushButton
90. #formating button
91. self.instructions.setStyleSheet('color: darkblue')
92. self.instructions.setFont(QFont("Arial",10,weight=QFont.Bold))
93. self.instructions.setMinimumHeight(35)
94. self.instructions.setMaximumWidth(110)
95. self.instructions.setToolTip("Game Instructions") #"enhancement"- set tool tip
97. self.O\_Shape=QPixmap("nought.png") # O\_Shape picture
98. self.X\_Shape=QPixmap("cross.png") # X\_Shape picture
99. self.blank\_Shape= QPixmap("blank.png") # blank picture
100. self.OXO=QPixmap("OXO.png") #OXO picture
102. self.OXO\_label=QLabel() # creating label to store OXO picture
103. #formating label
104. self.OXO\_label.setPixmap(self.OXO) #insert picture to the OXO label
105. self.OXO\_label.setAlignment(Qt.AlignCenter)
107. self.your\_shape=QLabel() # label to store players shape picture
108. self.your\_shape.setPixmap(self.blank\_Shape) #insert picture to the shape label
110. self.effect=QLabel() # creating label to store effects
112. #Creating Effects - "enhancement"
113. self.please\_wait\_movie=QMovie("PleaseWait.gif")
115. self.user\_feedback=QTextEdit() # Text edit to display users instructions
116. #formating Text edit
117. self.user\_feedback.setReadOnly(True) #"enhancement" - make Text edit to read only
118. self.user\_feedback.setStyleSheet('color: darkblue;background-color: lightgray')
119. self.user\_feedback.setFont(QFont("Arial",10,weight=QFont.Bold))
121. self.position0=QPushButton()# creating QPushButton
122. #formating QPushButton
123. self.icon0 = QIcon(self.blank\_Shape)
124. self.position0.setIcon(self.icon0)
125. self.position0.setIconSize(QSize(80, 80))
127. self.position1=QPushButton()# creating QPushButton
128. #formating QPushButton
129. self.icon1 = QIcon(self.blank\_Shape)
130. self.position1.setIcon(self.icon1)
131. self.position1.setIconSize(QSize(80, 80))
133. self.position2=QPushButton()# creating QPushButton
134. #formating QPushButton
135. self.icon2 = QIcon(self.blank\_Shape)
136. self.position2.setIcon(self.icon2)
137. self.position2.setIconSize(QSize(80, 80))
139. self.position3=QPushButton()# creating QPushButton
140. #formating QPushButton
141. self.icon3 = QIcon(self.blank\_Shape)
142. self.position3.setIcon(self.icon3)
143. self.position3.setIconSize(QSize(80, 80))
145. self.position4=QPushButton()# creating QPushButton
146. #formating QPushButton
147. self.icon4 = QIcon(self.blank\_Shape)
148. self.position4.setIcon(self.icon4)
149. self.position4.setIconSize(QSize(80, 80))
151. self.position5=QPushButton()# creating QPushButton
152. #formating QPushButton
153. self.icon5 = QIcon(self.blank\_Shape)
154. self.position5.setIcon(self.icon5)
155. self.position5.setIconSize(QSize(80, 80))
157. self.position6=QPushButton()# creating QPushButton
158. #formating QPushButton
159. self.icon6 = QIcon(self.blank\_Shape)
160. self.position6.setIcon(self.icon6)
161. self.position6.setIconSize(QSize(80, 80))
163. self.position7=QPushButton()# creating QPushButton
164. #formating QPushButton
165. self.icon7 = QIcon(self.blank\_Shape)
166. self.position7.setIcon(self.icon7)
167. self.position7.setIconSize(QSize(80, 80))
169. self.position8=QPushButton()# creating QPushButton
170. #formating QPushButton
171. self.icon8 = QIcon(self.blank\_Shape)
172. self.position8.setIcon(self.icon8)
173. self.position8.setIconSize(QSize(80, 80))
175. #players scores
176. self.X\_Shape\_Score=0
177. self.O\_Shape\_Score=0
179. self.score=QLabel("Score:") #creating label
180. #formating label
181. self.score.setFont(QFont("Arial",10,weight=QFont.Bold))
183. self.O\_Shape\_Score\_label=QLabel("Nought Shape: "+str(self.O\_Shape\_Score)) #creating label
184. #formating label
185. self.O\_Shape\_Score\_label.setFont(QFont("Arial",10,weight=QFont.Bold))
187. self.X\_Shape\_Score\_label=QLabel("Cross Shape: "+str(self.X\_Shape\_Score)) #creating label
188. #formating label
189. self.X\_Shape\_Score\_label.setFont(QFont("Arial",10,weight=QFont.Bold))
191. #"enhancement" - setting tool tips for all move buttons
192. self.position0.setToolTip("Place Move")
193. self.position1.setToolTip("Place Move")
194. self.position2.setToolTip("Place Move")
195. self.position3.setToolTip("Place Move")
196. self.position4.setToolTip("Place Move")
197. self.position5.setToolTip("Place Move")
198. self.position6.setToolTip("Place Move")
199. self.position7.setToolTip("Place Move")
200. self.position8.setToolTip("Place Move")
202. self.colour=QLabel("Game Colour:") #creating label
203. #formating label
204. self.colour.setFont(QFont("Arial",10,weight=QFont.Bold))
206. #create colour Combo box
207. self.colour\_combo=QComboBox()
208. # formating combo box
209. self.colour\_combo.setToolTip("Change Theme") #"enhancement"- set tool tip
210. self.colour\_combo.setStyleSheet('color: darkblue')
211. self.colour\_combo.setFont(QFont("Arial",10,weight=QFont.Bold))
212. #add items to the colour combo
213. self.colour\_combo.addItem("Dark Cyan")
214. self.colour\_combo.addItem("Dark Mode")
215. self.colour\_combo.addItem("Dark Magenta")
216. self.colour\_combo.addItem("Dark Red")
217. self.colour\_combo.addItem("Dark Blue")
218. self.colour\_combo.addItem("Dark Orange")
220. self.game\_server\_msg=QLabel(">>>>>>>>>>>> User FeedBack <<<<<<<<<<<<<") #create label
221. #formating label
222. self.game\_server\_msg.setAlignment(Qt.AlignCenter)
224. self.game\_board=QLabel(">>>>>>>>>>>>>> Game Board <<<<<<<<<<<<<<") #create label
225. #formating label
226. self.game\_board.setAlignment(Qt.AlignCenter)
228. grid\_1=QGridLayout()#creates grid layout
229. #adding widgets to the grid layout
230. grid\_1.addWidget(self.server,0,0)
231. grid\_1.addWidget(self.server\_input,0,1,1,2)
232. grid\_1.addWidget(self.connect,1,1)
233. grid\_1.addWidget(self.start\_server,1,2)
234. grid\_1.addWidget(self.score,4,0)
235. grid\_1.addWidget(self.X\_Shape\_Score\_label,4,1)
236. grid\_1.addWidget(self.O\_Shape\_Score\_label,4,2)
237. grid\_1.addWidget(self.your\_shape\_label,2,0)
238. grid\_1.addWidget(self.your\_shape,2,1)
239. grid\_1.addWidget(self.effect,2,2)
240. grid\_1.addWidget(self.colour,3,0)
241. grid\_1.addWidget(self.colour\_combo,3,1)
242. grid\_1.addWidget(self.game\_server\_msg,5,0,1,3)
243. grid\_1.addWidget(self.user\_feedback,6,0,1,3)
244. grid\_1\_widget=QWidget()
245. grid\_1\_widget.setLayout(grid\_1)
247. grid=QGridLayout()#creates grid layout
248. #adding widgets to the grid layout
249. grid.addWidget(self.position0,0,0)
250. grid.addWidget(self.position1,0,1)
251. grid.addWidget(self.position2,0,2)
252. grid.addWidget(self.position3,1,0)
253. grid.addWidget(self.position4,1,1)
254. grid.addWidget(self.position5,1,2)
255. grid.addWidget(self.position6,2,0)
256. grid.addWidget(self.position7,2,1)
257. grid.addWidget(self.position8,2,2)
259. self.grid\_widget=QGroupBox() #creating group box
260. #formating group box
261. self.grid\_widget.setObjectName("board")
262. self.grid\_widget.setStyleSheet("QGroupBox#board {font-weight: bold;font-size:14px;border: 2px solid orange}")
263. self.grid\_widget.setLayout(grid) #setting layout
265. hbox=QHBoxLayout()#creates horizontal box layout
266. #adding widgets to the hozintal box layout
267. hbox.addWidget(self.instructions)
268. hbox.addWidget(self.exit)
269. hbox\_widget=QWidget()
270. hbox\_widget.setLayout(hbox)
272. vbox=QVBoxLayout()#creates vertical box layout
273. #adding widgets to the vertical box layout
274. vbox.addWidget(self.OXO\_label)
275. vbox.addWidget(grid\_1\_widget)
276. vbox.addWidget(self.game\_board)
277. vbox.addWidget(self.grid\_widget)
278. vbox.addWidget(hbox\_widget)
279. self.setLayout(vbox)
281. self.button\_group=QButtonGroup() #creating group buttons
282. #adding buttons to the group buttons
283. self.button\_group.addButton(self.position0,0)
284. self.button\_group.addButton(self.position1,1)
285. self.button\_group.addButton(self.position2,2)
286. self.button\_group.addButton(self.position3,3)
287. self.button\_group.addButton(self.position4,4)
288. self.button\_group.addButton(self.position5,5)
289. self.button\_group.addButton(self.position6,6)
290. self.button\_group.addButton(self.position7,7)
291. self.button\_group.addButton(self.position8,8)
293. self.board = [' '] \* BOARD\_SIZE #Board list
294. self.shape = None
295. self.mute\_status=False #mute status
296. self.disable\_buttons() #"enhancement" - disabling buttons for placing moves since it a not yet any player turn to move
298. self.text\_colour("orange") #setting game text colour
299. self.setPalette(QPalette(QColor(self.colour\_combo.currentText()))) #setting game window colour
301. #signals connected to slots
302. self.button\_group.buttonClicked.connect(self.position\_clicked)
303. self.exit.clicked.connect(self.exit\_clicked)
304. self.connect.clicked.connect(self.connect\_clicked)
305. self.colour\_combo.currentTextChanged.connect(self.combo\_changed)
306. self.start\_server.clicked.connect(self.start\_server\_clicked)

309. # a slots called when "Start Server" push button is clicked
310. def start\_server\_clicked(self):
311. if checkIfProcessRunning("OXOGameServer.exe")!=True:
312. try:
313. os.startfile("OXOGameServer.exe")
314. self.user\_feedback.insertPlainText('\n>>Server started successfuly!')
315. self.user\_feedback.moveCursor(QTextCursor.End)
316. except:
317. self.user\_feedback.insertPlainText('\n>>"OXOGameServer.exe" could not be found!')
318. self.user\_feedback.moveCursor(QTextCursor.End)
319. else:
320. self.user\_feedback.insertPlainText('\n>>Server has already started!')
321. self.user\_feedback.moveCursor(QTextCursor.End)
322. #text colour change method
323. def text\_colour(self,colour): #"enhancement" - method
324. self.O\_Shape\_Score\_label.setStyleSheet('color: '+str(colour))
325. self.X\_Shape\_Score\_label.setStyleSheet('color: '+str(colour))
326. self.score.setStyleSheet('color: '+str(colour))
327. self.your\_shape\_label.setStyleSheet('color: '+str(colour))
328. self.server.setStyleSheet('color: '+str(colour))
329. self.colour.setStyleSheet('color: '+str(colour))
330. self.game\_board.setStyleSheet("color: "+str(colour)+";font-weight: bold;font-size:13px")
331. self.game\_server\_msg.setStyleSheet("color: "+str(colour)+";font-weight: bold;font-size:13px")
332. self.grid\_widget.setStyleSheet("QGroupBox#board {font-weight: bold;font-size:14px;border: 2px solid "+str(colour)+"}")
334. # a slots called when "combo box" text is Changed
335. def combo\_changed(self): #"enhancement" - method
336. if self.colour\_combo.currentText()=="Dark Mode":
337. self.text\_colour("Lightgreen") #setting game text colour
338. elif self.colour\_combo.currentText()=="Dark Magenta":
339. self.text\_colour("Cyan") #setting game text colour
340. elif self.colour\_combo.currentText()=="Dark Cyan":
341. self.text\_colour("Orange")  #setting game text colour
342. elif self.colour\_combo.currentText()=="Dark Orange":
343. self.text\_colour("LightGray")  #setting game text colour
344. elif self.colour\_combo.currentText()=="Dark Blue":
345. self.text\_colour("White")   #setting game text colour
346. elif self.colour\_combo.currentText()=="Dark Red":
347. self.text\_colour("Green") #setting game text colour
349. self.setPalette(QPalette(QColor(self.colour\_combo.currentText()))) #setting game window colour
351. #clear board method
352. def clear\_board(self):
353. self.board = [' '] \* BOARD\_SIZE
355. # a slots called when "button\_group" push buttons are clicked
356. def position\_clicked(self,button):
357. clicked\_position=self.button\_group.id(button)
358. try:
359. self.send\_message(str(clicked\_position))  #sends message to the server
360. except:
361. self.server\_disconnected() # alert the player that the sever has been disconnected
362. else:
363. try:
364. self.play\_loop() #Get message from the server
365. except:
366. self.server\_disconnected() # alert the player that the sever has been disconnected
368. # a slots called when "Exit Game" push button is clicked
369. def exit\_clicked(self):
370. self.close()
372. # a slots called when "Connect" push button is clicked
373. def connect\_clicked(self):
374. try:
375. self.connect\_to\_server(self.server\_input.text()) #connect to the server
376. #lets the players know they have connected to the server successfuly
377. self.user\_feedback.insertPlainText("\n>>Connected to the server successfuly!")
378. self.connect.setText("Connected")
379. self.user\_feedback.moveCursor(QTextCursor.End) #"enhancement" - autoscroll text eqit
380. self.connect.setEnabled(False) # "enhancement" - disable connect button
381. self.connect.setStyleSheet('color: darkgray')
382. except:
383. #lets the players know they have not connected to the server successfuly
384. self.user\_feedback.insertPlainText("\n>>Error connecting to server!")
385. self.user\_feedback.moveCursor(QTextCursor.End) #"enhancement" - autoscroll text eqit
386. else:
387. #"enhancement" - start please wait movie effect
388. self.effect.setMovie(self.please\_wait\_movie)
389. self.please\_wait\_movie.start()
390. #wait for 5 seconds
391. self.wait(5000)
392. #"enhancement" - stop please wait movie effect
393. self.please\_wait\_movie.stop()
394. self.effect.clear()#clear effect label
395. try:
396. self.play\_loop() #Get message from the server
397. except:
398. self.server\_disconnected() # alert the player that the sever has been disconnected
399. # a slots called when LoopThread emit a signal push button is clicked
400. def handle\_message(self,msg):
401. if msg[:msg.find(",")]=="new game":
402. self.shape=msg[-1] #index shape
403. if self.shape=="X":
404. shape\_name="cross"
405. self.your\_shape.setPixmap(self.X\_Shape)
406. else:
407. shape\_name="nought"
408. self.your\_shape.setPixmap(self.O\_Shape)
409. #lets the players know their shape
410. self.user\_feedback.insertPlainText("\n>>New game is about to start, your character is a "+shape\_name+" shape.")
411. self.user\_feedback.moveCursor(QTextCursor.End) #"enhancement" - autoscroll text eqit
413. elif msg=="your move":
414. self.display\_board() # display Board
415. self.user\_feedback.insertPlainText("\n>>It's your turn to move.") #lets the players know it their turn to move
416. self.user\_feedback.moveCursor(QTextCursor.End) #"enhancement" - autoscroll text eqit
417. self.enable\_buttons() #"enhancement" - enabling buttons for placing moves since it player turn to move
419. elif msg=="opponents move":
420. self.user\_feedback.insertPlainText("\n>>It's the opponent turn to move.") #lets the players know it their opponent turn to move
421. self.user\_feedback.moveCursor(QTextCursor.End) #"enhancement" - autoscroll text eqit
422. self.disable\_buttons() #"enhancement" - disabling buttons for placing moves since it a not yet any player turn to move
423. #"enhancement" - start please wait movie effect
424. self.effect.setMovie(self.please\_wait\_movie)
425. self.please\_wait\_movie.start()
426. #wait for 250 micro seconds
427. self.wait(250)
428. #"enhancement" - stop please wait movie effect
429. self.please\_wait\_movie.stop()
430. self.effect.clear()#clear effect label
432. elif msg[:msg.find(",")]=="valid move":
433. shape=msg[-3] #index shape
434. position=int(msg[-1]) #index position
435. self.board[position]= shape #insert shape to the board
436. self.display\_board() #display the game board
438. elif msg=="invalid move":
439. self.user\_feedback.insertPlainText("\n>>Invalid move.") #lets the player know that an invalid move was made
440. self.user\_feedback.moveCursor(QTextCursor.End) #"enhancement" - autoscroll text eqit
442. elif msg[:msg.find(",")]=="game over":
443. #"enhancement"
444. #get window x and y geometry
445. x\_geo=self.geometry().x()+27
446. y\_geo=self.geometry().y()+160
448. results=msg[-1] #index shape
449. if results=="X":self.X\_Shape\_Score+=1
450. elif results=="O":self.O\_Shape\_Score+=1
452. if results=="X" and self.shape=="X":
453. self.O\_Shape\_Score\_label.setText("Nought Shape: "+str(self.O\_Shape\_Score))
454. self.X\_Shape\_Score\_label.setText("Cross Shape: "+str(self.X\_Shape\_Score))
455. win=Win\_Window()
456. win.setGeometry(x\_geo,y\_geo,340, 380) #set window geometry
457. win.show()
458. win.win\_play\_movie() #play Movie Effect for Winning -"enhancement"
459. elif results=="O" and self.shape=="O":
460. self.O\_Shape\_Score\_label.setText("Nought Shape: "+str(self.O\_Shape\_Score))
461. self.X\_Shape\_Score\_label.setText("Cross Shape: "+str(self.X\_Shape\_Score))
462. win=Win\_Window() # instance of Win\_Window class
463. win.setGeometry(x\_geo,y\_geo,340, 380) #set window geometry
464. win.show()
465. win.win\_play\_movie() #play Movie Effect for Winning -"enhancement"
466. elif results=="X" and self.shape=="O":
467. self.O\_Shape\_Score\_label.setText("Nought Shape: "+str(self.O\_Shape\_Score))
468. self.X\_Shape\_Score\_label.setText("Cross Shape: "+str(self.X\_Shape\_Score))
469. win=Lose\_Window() # instance of Lose\_Window class
470. win.setGeometry(x\_geo,y\_geo,340, 380) #set window geometry
471. win.show()
472. win.lose\_play\_movie() #play Movie Effect for Losing -"enhancement"
473. elif results=="O" and self.shape=="X":
474. self.O\_Shape\_Score\_label.setText("Nought Shape: "+str(self.O\_Shape\_Score))
475. self.X\_Shape\_Score\_label.setText("Cross Shape: "+str(self.X\_Shape\_Score))
476. win=Lose\_Window() # instance of Lose\_Window class
477. win.setGeometry(x\_geo,y\_geo,340, 380) #set window geometry
478. win.show()
479. win.lose\_play\_movie() #play Movie Effect for Losing -"enhancement"
480. else:
481. win=Draw\_Window() # instance of Draw\_Window class
482. win.setGeometry(x\_geo,y\_geo,344, 428) #set window geometry
483. win.show()
484. win.draw\_play\_movie() #play Movie Effect for a Tie -"enhancement"
486. elif msg=="play again":
487. #Pop up Dialog to Ask the players whether they want to play again
488. answer=QMessageBox.question(self,"Game Over", "Do you want to play again?",QMessageBox.Yes | QMessageBox.No)
489. if answer==QMessageBox.Yes:
490. answer="y"
491. try:
492. self.send\_message(answer) #send message to the server
493. except:
494. self.server\_disconnected() # alert the player that the sever has been disconnected
495. else:
496. self.clear\_board() # clear board if the answer from the player is "yes"
497. self.display\_board() #display the game board
498. else:
499. answer="n"
500. try:
501. self.send\_message(answer) #send message to the server
502. except:
503. self.server\_disconnected() # alert the player that the sever has been disconnected
504. elif msg=="exit game":
505. #lets the players know that the game is about to close
506. self.user\_feedback.insertPlainText("\n>>Game is about to close, one or both of the players doesn't want to play again.")
507. self.user\_feedback.moveCursor(QTextCursor.End) #"enhancement" - autoscroll text eqit
508. if checkIfProcessRunning("OXOGameServer.exe"): #check if the program is running
509. try:
510. os.system("TASKKILL /F /IM OXOGameServer.exe") #close the program
511. except:
512. pass
513. self.wait(2000) #wait for 2 second
514. self.close() #close window
516. #method to alert the user when the server has been disconnected
517. def server\_disconnected(self) :
518. self.user\_feedback.insertPlainText("\n>>Server Disconnected. An existing connection was forcibly closed by the remote host!")
519. self.socket.close()
520. self.user\_feedback.moveCursor(QTextCursor.End)
521. self.connect.setEnabled(True)
522. self.connect.setText("Connect")
523. self.connect.setStyleSheet("color: darkblue")
524. self.your\_shape.setPixmap(self.blank\_Shape)
525. self.clear\_board()
526. self.display\_board()
527. self.disable\_buttons()
528. self.socket = socket(AF\_INET, SOCK\_STREAM)
530. # display board method
531. def display\_board(self):
532. for move in range(9):
533. if self.board[move]=="X" and move==0:
534. self.icon0 = QIcon(self.X\_Shape)
535. self.position0.setIcon(self.icon0)
536. elif self.board[move]=="O" and move==0:
537. self.icon0 = QIcon(self.O\_Shape)
538. self.position0.setIcon(self.icon0)
539. elif self.board[move]==" " and move==0:
540. self.icon0 = QIcon(self.blank\_Shape)
541. self.position0.setIcon(self.icon0)
542. elif self.board[move]=="X" and move==1:
543. self.icon1 = QIcon(self.X\_Shape)
544. self.position1.setIcon(self.icon1)
545. elif self.board[move]=="O" and move==1:
546. self.icon1 = QIcon(self.O\_Shape)
547. self.position1.setIcon(self.icon1)
548. elif self.board[move]==" " and move==1:
549. self.icon1 = QIcon(self.blank\_Shape)
550. self.position1.setIcon(self.icon1)
551. elif self.board[move]=="X" and move==2:
552. self.icon2 = QIcon(self.X\_Shape)
553. self.position2.setIcon(self.icon2)
554. elif self.board[move]=="O" and move==2:
555. self.icon2 = QIcon(self.O\_Shape)
556. self.position2.setIcon(self.icon2)
557. elif self.board[move]==" " and move==2:
558. self.icon2 = QIcon(self.blank\_Shape)
559. self.position2.setIcon(self.icon2)
560. elif self.board[move]=="X" and move==3:
561. self.icon3 = QIcon(self.X\_Shape)
562. self.position3.setIcon(self.icon3)
563. elif self.board[move]=="O" and move==3:
564. self.icon3 = QIcon(self.O\_Shape)
565. self.position3.setIcon(self.icon3)
566. elif self.board[move]==" " and move==3:
567. self.icon3 = QIcon(self.blank\_Shape)
568. self.position3.setIcon(self.icon3)
569. elif self.board[move]=="X" and move==4:
570. self.icon4 = QIcon(self.X\_Shape)
571. self.position4.setIcon(self.icon4)
572. elif self.board[move]=="O" and move==4:
573. self.icon4 = QIcon(self.O\_Shape)
574. self.position4.setIcon(self.icon4)
575. elif self.board[move]==" " and move==4:
576. self.icon4 = QIcon(self.blank\_Shape)
577. self.position4.setIcon(self.icon4)
578. elif self.board[move]=="X" and move==5:
579. self.icon5 = QIcon(self.X\_Shape)
580. self.position5.setIcon(self.icon5)
581. elif self.board[move]=="O" and move==5:
582. self.icon5 = QIcon(self.O\_Shape)
583. self.position5.setIcon(self.icon5)
584. elif self.board[move]==" " and move==5:
585. self.icon5 = QIcon(self.blank\_Shape)
586. self.position5.setIcon(self.icon5)
587. elif self.board[move]=="X" and move==6:
588. self.icon6 = QIcon(self.X\_Shape)
589. self.position6.setIcon(self.icon6)
590. elif self.board[move]=="O" and move==6:
591. self.icon6 = QIcon(self.O\_Shape)
592. self.position6.setIcon(self.icon6)
593. elif self.board[move]==" " and move==6:
594. self.icon6 = QIcon(self.blank\_Shape)
595. self.position6.setIcon(self.icon6)
596. elif self.board[move]=="X" and move==7:
597. self.icon7 = QIcon(self.X\_Shape)
598. self.position7.setIcon(self.icon7)
599. elif self.board[move]=="O" and move==7:
600. self.icon7 = QIcon(self.O\_Shape)
601. self.position7.setIcon(self.icon7)
602. elif self.board[move]==" " and move==7:
603. self.icon7 = QIcon(self.blank\_Shape)
604. self.position7.setIcon(self.icon7)
605. elif self.board[move]=="X" and move==8:
606. self.icon8 = QIcon(self.X\_Shape)
607. self.position8.setIcon(self.icon8)
608. elif self.board[move]=="O" and move==8:
609. self.icon8 = QIcon(self.O\_Shape)
610. self.position8.setIcon(self.icon8)
611. elif self.board[move]==" " and move==8:
612. self.icon8 = QIcon(self.blank\_Shape)
613. self.position8.setIcon(self.icon8)
615. #method to wait for number of micro seconds
616. def wait(self,microsec):
617. loop = QEventLoop()
618. QTimer.singleShot(microsec, loop.quit)
619. loop.exec\_()
620. # method to get message from the server
621. def play\_loop(self):
622. while True:
623. msg = self.receive\_message()
624. if len(msg):
625. self.handle\_message(msg)
626. if msg=="your move":break
627. else:
628. self.opponent\_disconnected() # alert the player that their opponent has  disconnected from the server
629. break
630. #method to alert the user when the opponent has disconnected from the server
631. def opponent\_disconnected(self):
632. self.user\_feedback.insertPlainText("\n>>Your opponnent has disconnected from the server!")
633. self.socket.close()
634. self.user\_feedback.moveCursor(QTextCursor.End)
635. self.connect.setEnabled(True)
636. self.connect.setText("Connect")
637. self.connect.setStyleSheet("color: darkblue")
638. self.your\_shape.setPixmap(self.blank\_Shape)
639. self.clear\_board()
640. self.display\_board()
641. self.disable\_buttons()
642. self.socket = socket(AF\_INET, SOCK\_STREAM)
644. #disable grid buttons method
645. def disable\_buttons(self):
646. self.position0.setEnabled(False)
647. self.position1.setEnabled(False)
648. self.position2.setEnabled(False)
649. self.position3.setEnabled(False)
650. self.position4.setEnabled(False)
651. self.position5.setEnabled(False)
652. self.position6.setEnabled(False)
653. self.position7.setEnabled(False)
654. self.position8.setEnabled(False)
656. #enable grid buttons method
657. def enable\_buttons(self):
658. self.position0.setDisabled(False)
659. self.position1.setDisabled(False)
660. self.position2.setDisabled(False)
661. self.position3.setDisabled(False)
662. self.position4.setDisabled(False)
663. self.position5.setDisabled(False)
664. self.position6.setDisabled(False)
665. self.position7.setDisabled(False)
666. self.position8.setDisabled(False)
668. #main function
669. def main():
670. app=QApplication(sys.argv)
671. game=OXO\_game() # instance of OXO\_game class
672. game.show() # show instance of OXO\_game class
673. # Player instructions
674. game.user\_feedback.insertPlainText(">>Enter server to connect to and click the connect button to connect to that server.")
675. game.user\_feedback.moveCursor(QTextCursor.End)
676. instruc=Instruction\_Window() # instance of Instruction\_Window class
677. game.instructions.clicked.connect(instruc.instructions\_clicked) #connect signal to slot
678. sys.exit(app.exec\_())
679. if \_\_name\_\_=="\_\_main\_\_":
680. main()

## Game\_Over.py

1. # OXO GAME- Game Over Windows
2. # Bhekanani Cele & Matthew Weppenaar
3. # 03 May 2019
4. # Added "enhancement" to the game
6. import sys
7. from PyQt5.QtCore import\*
8. from PyQt5.QtWidgets import\*
9. from PyQt5.QtGui import\*
10. from PyQt5.QtMultimedia import \*
12. class Win\_Window(QWidget):
13. def \_\_init\_\_(self, parent=None):
14. QWidget.\_\_init\_\_(self, parent)
15. self.setGeometry(527, 180, 340, 380)
16. self.setWindowFlags(Qt.WindowCloseButtonHint | Qt.WindowMinimizeButtonHint) #"enhancement"- disabling maximise window flag
17. self.setWindowIcon(QIcon('icon.jpg')) #window  icon
18. self.setWindowTitle('Game Over')  # window title
20. self.setPalette(QPalette(QColor("darkgray"))) #setting game window colour
22. self.win=QLabel() #creating label
23. self.win\_movie=QMovie("win.gif") #Creating Winning Effect - "enhancement"
25. vbox=QVBoxLayout() #creates horizontal box layout
26. vbox.addWidget(self.win) #adding widget to the hozintal box layout
27. self.setLayout(vbox)
29. #Method to play Movie Effect for winning -"enhancement"
30. def win\_play\_movie(self):
31. self.play\_mp3("GameOver\_YouWin.mp3") #play mp3 -"enhancement"
32. self.win.setMovie(self.win\_movie)
33. self.win\_movie.start()  #"enhancement" - start win movie effect
34. self.wait(10000) # wait for 10 seconds
35. self.win\_movie.stop() #"enhancement" - stop win movie effect
36. self.close()  #close window
38. #method to wait for number of micro seconds
39. def wait(self,microsec):
40. loop = QEventLoop()
41. QTimer.singleShot(microsec, loop.quit)
42. loop.exec\_()
44. #method to play mp3 files -"enhancement"
45. def play\_mp3(self,mp3\_file):
46. self.playlist = QMediaPlaylist()
47. self.player = QMediaPlayer()
48. url = QUrl.fromLocalFile("./"+mp3\_file)
49. self.playlist.addMedia(QMediaContent(url))
50. self.player.setPlaylist(self.playlist)
51. self.player.play()
53. class Draw\_Window(QWidget):
54. def \_\_init\_\_(self, parent=None):
55. QWidget.\_\_init\_\_(self, parent)
56. self.setGeometry(527, 180, 344, 428)
57. self.setWindowFlags(Qt.WindowCloseButtonHint | Qt.WindowMinimizeButtonHint) #"enhancement"- disabling maximise window flag
58. self.setWindowTitle('Game Over')  # window title
59. self.setWindowIcon(QIcon('icon.jpg')) #window  icon
61. self.setPalette(QPalette(QColor("darkgray"))) #setting game window colour
63. self.draw=QLabel() #creating label
64. self.draw\_movie=QMovie("tie.gif") #Creating Tie Effect - "enhancement"
66. vbox=QVBoxLayout() #creates horizontal box layout
67. vbox.addWidget(self.draw) #adding widget to the hozintal box layout
68. self.setLayout(vbox)
70. #Method to play Movie Effect for a Tie -"enhancement"
71. def draw\_play\_movie(self):
72. win= Win\_Window() # instance of Win\_Window class
73. win.play\_mp3("GameOver.mp3")  #play mp3 -"enhancement"
74. self.draw.setMovie(self.draw\_movie)
75. self.draw\_movie.start() #"enhancement" - start Tie movie effect
76. win.wait(10000) # wait for 10 seconds
77. self.draw\_movie.stop() #"enhancement" - stop Tie movie effect
78. self.close() #close window
80. class Lose\_Window(QWidget):
81. def \_\_init\_\_(self, parent=None):
82. QWidget.\_\_init\_\_(self, parent)
83. self.setGeometry(527, 180, 340, 380)
84. self.setWindowFlags(Qt.WindowCloseButtonHint | Qt.WindowMinimizeButtonHint) #"enhancement"- disabling maximise window flag
85. self.setWindowTitle('Game Over')  # window title
86. self.setWindowIcon(QIcon('icon.jpg')) #window  icon
88. self.setPalette(QPalette(QColor("darkgray"))) #setting game window colour
90. self.lose=QLabel() #creating label
91. self.lose\_movie=QMovie("lose.gif") #Creating Losing Effect - "enhancement"
93. vbox=QHBoxLayout() #creates horizontal box layout
94. vbox.addWidget(self.lose) #adding widget to the hozintal box layout
95. self.setLayout(vbox)
97. #Method to play Movie Effect for Losing -"enhancement"
98. def lose\_play\_movie(self):
99. win=Win\_Window() # instance of Win\_Window class
100. win.play\_mp3("GameOver\_YouLose.mp3") #play mp3 -"enhancement"
101. self.lose.setMovie(self.lose\_movie)
102. self.lose\_movie.start()  #"enhancement" - start lose movie effect
103. win.wait(10000) # wait for 10 seconds
104. self.lose\_movie.stop() #"enhancement" - stop lose movie effect
105. self.close()  #close window

## check\_running\_program.py

1. import psutil
3. def checkIfProcessRunning(processName):
4. '''''
5. Check if there is any running process that contains the given name processName.
6. '''
7. #Iterate over the all the running process
8. for proc in psutil.process\_iter():
9. try:
10. # Check if process name contains the given name string.
11. if processName.lower() in proc.name().lower():
12. return True
13. except (psutil.NoSuchProcess, psutil.AccessDenied, psutil.ZombieProcess):
14. pass
15. return False