**Interactive Buttons and Sounds in PyGame**

In Pygame, we can create simple buttons by drawing a rectangular object on the screen and write codes to make it become interactive buttons. Hence, when we use a mouse to click on that button to take actions (start a game, quit a game, etc.). Another use case, if we press button “p” to pause a game, two buttons are shown letting us choose to continue or quit a game.

First thing first, we need a function that draw a rectangular object (with a pre-defined size) on the screen. Below function is to be included in your Pygame.

|  |
| --- |
| def button(msg,x,y,w,h,ic,ac,action=None):  mouse = pygame.mouse.get\_pos() #To get the position of a mouse cursor  click = pygame.mouse.get\_pressed() #Mouse click event  if x+w > mouse[0] > x and y+h > mouse[1] > y: # If a mouse cursor appears in the button  pygame.draw.rect(gameDisplay, ac,(x,y,w,h)) # Draw the button with status ACTIVE (ac). # Color of rect. object will also change.  if click[0] == 1 and action != None: # For a mouse click,  action() # call a function that is passed into action()  else:  pygame.draw.rect(gameDisplay, ic,(x,y,w,h)) # If a mouse cursor is not on the button,  # draw the button with inactive status.  # No color change  smallText = pygame.font.SysFont(None,20) # Specify font type and size  textSurf, textRect = text\_objects(msg, smallText) # Create object for font  textRect.center = ( (x+(w/2)), (y+(h/2)) ) # Make it center of the button  gameDisplay.blit(textSurf, textRect) # Blit (display) font on the button |

From the previous worksheet, a function crash( ) is to be modified to include interactive buttons (Play Again and Quit). If we click on the Play Again button, a game is restarted. Otherwise, we quit the game if the other button is clicked. Now let’s see the codes that have to be added to the function crash( ).

|  |
| --- |
| while True:  for event in pygame.event.get(): #เช็ค event ว่ามีการปิดเกมส์หรือไม่ ถ้าใช่ก็ออกจากเกมส์  if event.type == pygame.QUIT:  pygame.quit()  quit()  #gameDisplay.fill(WHITE)  button("Play Again",150,450,100,50,GREEN,BRIGHTGREEN,gameLoop)  button("Quit",550,450,100,50,RED,BRIGHTRED,quitgame)  pygame.display.update()  clock.tick(30) |

Explanation for all parameters passing to function button( )

button("Play Again",150,450,100,50,GREEN,BRIGHTGREEN,gameLoop)

* Play Again – Text to be put in the rectangular area.
* x, y, – Location of the button.
* w, h – Width and height of the button.
* GREEN, BRIGHTGREEN – Colors of the button when it is active or inactive.
* gameLoop( ) – This is a function that will be called (activated) when we click the button. In other word, action( ) in function button( ) takes gameLoop( ) as an input parameter.

**1) Modify the function gameIntro( ) to include two buttions, namely (START) and (QUIT) Click START to play the game or QUIT to quit the game.**

While we are playing the game, our boss may come around and the game needs to be paused. Hence a programming logic to handle a game pause/unpause should be included in our game. The given function pause() below includes four lines of code that define font type and size. **Complete the rest by including two buttons CONTINUE and QUIT.**

|  |
| --- |
| def pause():  largeText = pygame.font.SysFont("freesansbold.ttf",115)  TextSurf, TextRect = text\_objects("Paused", largeText)  TextRect.center = ((displayWidth/2),(displayHeight/2))  gameDisplay.blit(TextSurf, TextRect)  **# Complete the function by including the codes for two buttons here.**  **# while True:**  **# …** |

Note that for CONTINUE button, function unpause() shall be called. For QUIT button, quitGame( ) shall be called. Function unpause( ) is given below. Note that in function pause( ), you need to slightly modify the code from while True to while pause, since we are going to use this variable to carry True or False.

|  |
| --- |
| def unpause():  global pause  pause = False |

Now back to function gameLoop( ), we need to include codes to detect a key press for key “p”.

For if event.type == pygame.KEYDOWN, the following codes must be added.

|  |
| --- |
| if event.key == pygame.K\_p:  pause = True  pause() |

To make this variable, *pause*, to be recognized as a global variable (you do not need to pass this variable into the function or return it out), in gameLoop( ), we need to define *pause* as a global variable by including this line.

|  |
| --- |
| global pause |

**Sound in PyGame**

In Pygame, there are ready-to-use and built-in functions for sound management. What we need to do is to load sound in Pygame and initialize the sound mixer before playing the sound in the game. At the beginning of your code, add these two lines.

|  |
| --- |
| pygame.mixer.init() #Initialize the sound mixer  crash\_sound = pygame.mixer.Sound("crash.wav") # Sound for crash is loaded. |

After loading .wave, we can play the sound using the following codes:

Note that the given codes are to play a sound we the player crashes with the meteor. The given codes are to be included in function crash( ).

|  |
| --- |
| pygame.mixer.Sound.play(crash\_sound)  pygame.mixer.music.stop()  #time.sleep(1) #You can hold the game for a second or more. |

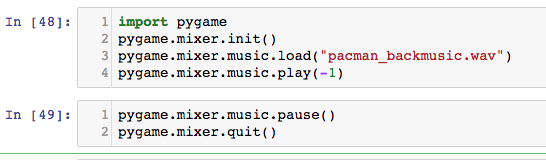
**Play sounds for**

* **collecting special object(s), e.g., durian or other object that you use.**
* **Level up.**

**In the case that you want o continuously play a background music, Pygame does have a function for you. What you need to do is 1) Load the music and 2) Play it with an option -1 (continuous play) as shown below.**

|  |
| --- |
| **############**  **pygame.mixer.music.load('pacman\_backmusic.wav')**  **pygame.mixer.music.play(-1)**  **############** |

**Hint: PyGame may not be able to support all .wav files. To test whether .wave file can be played or not, use the following codes. New jupyter notebook is recommended so you won’t mix this code with your Pygame code. Put the codes in two cells as shown below. Run the first cell to play and the second cell to stop playing.**

****

**If you want the background music to be paused while the game is pausing, two built-in functions must be included in your Pygame.**

**pygame.mixer.music.pause() are pygame.mixer.music.unpause() are to be included in pause( ) and unpuase( ), respectively.**