Creating a basic game in pygame

Brief

For this challenge you are going to create a simple game within pygame that will have a square moving around the screen!

What you will need

- Thonny
- pygame

Wait I don't have pygame!

Don't panic we can fix this! in Thonny:

1. click on Tools -> Manage Packages...
2. In the search box type pygame
3. click pygame
4. click install

Starter code

Type the following code into Thonny

```
 import pygame

pygame.init() # starts pygame
screen = pygame.display.set_mode((600, 500)) # sets the size of the window
clock = pygame.time.clock()
running = True

while running: # the game loop
    for event in pygame.event.get(): # checks the event queue
        if event.type == pygame.QUIT: # looks for the quit event
            running = False # Exits the game loop

    screen.fill("purple") # makes the screen purple
    pygame.display.flip() # Updates the screen
    clock.tick(60) # Makes games run at 60 frames a second

pygame.quit() # closes pygame
```

Challenges

- Run the code and see what happens, you should see a purple window
- Try changeing the screen to a light blue colour (Colour names don't have spaces in)
- Try changing the size of the window

Adding a Rectangle

A colourful window is good and all but we need to add more! lets add a rectangle... okay it will be a square however pygame treats every 4 sided shape as a rectangle.

Add the following code between the `running = True` line and the `while running` line.

```
 running = True

square = pygame.Rect(100, 100, 100, 100)

while running:
```

Run the code what has changed?

A bit underwhelming? Nevermind we can fix it. We have told Pygame that we have a rectangle however nothing was on the screen! We forgot to tell it to draw the rectangle onto the screen. Update your game loop so it looks like this

```
 #earlier game loop
screen.fill("purple")
pygame.draw.rect(screen, (255, 0, 0), square)
pygame.display.flip()
# Rest of game loop
```

Run the code you should see a red square! In the game loop the draw line takes 3 parameters, the surface to draw on, the colour, and the rectangle to draw.

Challenges

1. Change the numbers in the line: `square = pygame.Rect(100,100,100,100)` see what changes
2. Change the numbers inside the line: `pygame.draw.rect(screen, (255, 0, 0), square)` See what colours you can make!

Moving the Rectangle

Our little "game" is coming to life however lets make the square move!

Within the event loop add this code:

```
 # Previous code
for event in pygame.event.get(): # checks the event queue
    if event.type == pygame.QUIT: # looks for the quit event
        running = False # Exits the game loop
    # New code
    elif event.type == pygame.KEYDOWN:
        if event.key == pygame.K_RIGHT:
            square.x += 10
# Rest of code
```

The code checks the event queue for the event of keydown. It then checks if the key pressed was the right arrow key. If the right arrow has been pressed, the x position of the square is increased by 10 pixels.

Run the code, if you press the right arrow key the square should move to the right

Challenges

1. Add code to the event queue checker to see if the other 3 arrow keys are pressed
2. Change the x and y position of the square (the coordinates of the top left corner of the window are 0, 0)

Super Challenges

Well done you now have a moveable square on the screen! now it is time for a super challenge!

1. Add another square onto the screen
2. Change the colour so that it is not the same colour as the original square
3. Position the new square the bottom right of the screen
4. Allow it to move using the w, a, s, d keys. For the event queue that is: `pygame.K_w` for example