Pygame Events

The Game Loop:

Pygame was designed for making games.
https://www.youtube.com/watch?v=umHZ6wnQTyQ

No matter what language is used to develop a game, there is always a Main Game Loop.

The Main game loop looks something like:

```
loop
        <check user input>
        <move good guy>    - move a little bit based on input
        <move bad guy>     - move each one a little bit based on some AI
        <check collisions>
        <draw everything>  - no drawing should happen anywhere else
        <delay>
end loop
```

This loop allows you to have a whole host of game objects moving simultaneously.

It is rare that there would be another loop within the game loop, beyond a loop that checks all the bad guys/good guys, etc.

The use of Booleans is emphasized in a game loop, for example, if the character is invisible (isVisible == False) then collisions do not happen.

Events:

When your program gets any input from the user (an event), Pygame remembers the event and puts it in an "event queue" (a first-in, first-out list).

Whenever something takes place during the running of your computer program/game, such as a key press, a mouse click or even the movement of the mouse, an event is generated.

Every event has a CONSTANT attached to it to identify the type. More on this later.

Using pygame.event.get() which pops events off the list so you can deal with them.

All the events are in a list, so you need to search through the entire list for the type of event you are looking for.

Try the following code which waits for the user to close the window before quitting the program.

```
import pygame
pygame.init()
SIZE = (width,height) = (800,600)
screen = pygame.display.set_mode(SIZE)

GREEN = (0,255,0)
BLACK = (0,0,0)

# clear the screen, draw off screen and then display
def drawScene(screen):
    screen.fill(BLACK)
    pygame.draw.circle(screen, GREEN, (400,200), 100)

running = True
myClock = pygame.time.Clock()

while running:   # this is our game loop

    # Check all the events that happen
    for evnt in pygame.event.get():

        # if the user tries to close the window, then raise the "flag"
        if evnt.type == pygame.QUIT:
            running = False

    drawScene(screen)

    # waits long enough to have 60 fps
    pygame.display.flip()
    myClock.tick(60)

pygame.quit()
```

In the above program we are looking specifically for the user closing the window.

If one of the events was a pygame.QUIT, then you need to write the code that does what you want to do when the user tries to close the window, in this case, set running to be False so the game loop will finish.
There are many events created during the running of a program.

Add a print evnt after the for loop in the previous program, so:

```
for evnt in pygame.event.get():
        print (evnt)
        # if the user tries to close the window, then raise the "flag"
        if evnt.type == pygame.QUIT:
            running = False

drawScene(screen)
```

If you are running the new version of Wing, you should be able to see a steady stream of events passing as you move your mouse about.

Here is a snippet of what I got:

```
<Event(4-MouseMotion {'buttons': (0, 0, 0), 'pos': (303, 247), 'rel': (0, 3)})>
<Event(4-MouseMotion {'buttons': (0, 0, 0), 'pos': (303, 248), 'rel': (0, 1)})>
<Event(5-MouseButtonDown {'button': 1, 'pos': (303, 248)})>
<Event(6-MouseButtonUp {'button': 1, 'pos': (303, 248)})>
<Event(2-KeyDown {'scancode': 36, 'key': 106, 'unicode': u'j', 'mod': 4096})>
<Event(3-KeyUp {'scancode': 36, 'key': 106, 'mod': 4096})>
<Event(2-KeyDown {'scancode': 36, 'key': 106, 'unicode': u'j', 'mod': 4096})>
<Event(3-KeyUp {'scancode': 36, 'key': 106, 'mod': 4096})>
<Event(4-MouseMotion {'buttons': (0, 0, 0), 'pos': (304, 248), 'rel': (1, 0)})>
<Event(4-MouseMotion {'buttons': (0, 0, 0), 'pos': (312, 248), 'rel': (8, 0)})>
<Event(4-MouseMotion {'buttons': (0, 0, 0), 'pos': (321, 248), 'rel': (9, 0)})>
```

There are many types of events, above you can see I moved the mouse, clicked the button and typed some keys.

Event Types:

Event Type	Key Variables	Example
QUIT		if evnt.type == pygame.QUIT:
KEYDOWN	unicode key mod	print (evnt) if evnt.type == pygame.KEYDOWN: print (evnt.unicode) print (evnt.key) print (evnt.mod)
Output: <Event(2-KeyDown {'scancode': 35, 'key': 104, 'unicode': 'h', 'mod': 4096})> h 104 4096		
KEYUP	key mod	if evnt.type == pygame.KEYUP: print (evnt.key) print (evnt.mod)
Output: <Event(3-KeyUp {'scancode': 35, 'key': 104, 'mod': 4096})> 104 4096		
MOUSEMOTION	pos rel buttons	if evnt.type == pygame.MOUSEMOTION: print (evnt.pos) print (evnt.rel) print (evnt.buttons)
Output: <Event(4-MouseMotion {'buttons': (0, 0, 0), 'pos': (446, 131), 'rel': (-18, 59)})> (446, 131) (-18, 59) (0, 0, 0)		
MOUSEBUTTONUP	pos button	if evnt.type == pygame.MOUSEBUTTONUP: print (evnt.pos) print (evnt.button)
<Event(6-MouseButtonUp {'button': 1, 'pos': (388, 195)})> (388, 195) 1		

MOUSEBUTTONDOWN	pos button	if evnt.type == pygame.MOUSEBUTTONDOWN: print evnt.pos print evnt.button
<Event(5-MouseButtonDown {'button': 1, 'pos': (571, 246)})> (571, 246) 1		