Lecture: 6-1

- Prerequisites for this lecture are: 5-1, 5-2 and 5-3.

John Romero Programming Proverbs

- 6. “As soon as you see a bug, you fix it. Do not continue on. If you don’t fix your bugs your new code will be built on a buggy codebase and ensure an unstable foundation.”

- John Romero, “The Early Days of Id Software - John Romero @ WeAreDevelopers Conference 2017”

Python Classes

- you can use the `class` keyword to create your own classes
- here is a tiny example of a class

Python Classes

tinyclass.py

```
#!/usr/bin/env python3

import math

class vector:
    def __init__ (self, pair):
        self.pair = pair
    def get_x (self):
        return self.pair[0]
    def get_y (self):
        return self.pair[1]
    def get_pair (self):
        return self.pair
    def get_length (self):
        return math.sqrt(sqr(self.pair[0]) + sqr(self.pair[1]))
```

Python Classes

```
def sqr (x):
    return x*x

v = vector ([3, 4])
print ("using the Pythagorean theory the hypotenuse", end=' ')
print ("(or magnitude) of the vector", end=' ')
print (v.get_pair(), "=", v.get_length())
```

Extending last weeks tutorial to use classes

- create a ball class with a constructor which has an initial position and a velocity vector

Extending last weeks tutorial to use classes

```
#!/usr/bin/env python3

import sys, pygame, random

class ball:
    def __init__ (self, initial_position, velocity):
        self.curpos = initial_position
        self.velocity = velocity
        self.ball = pygame.image.load("ball.png").convert ()
        self.ballrect = self.ball.get_rect ()
        self.ballrect = self.ballrect.move (initial_position)

    def update (self, surface):
        self.ballrect = self.ballrect.move (self.velocity)
        if self.ballrect.left < 0 or self.ballrect.right > width:
            self.velocity[0] = -self.velocity[0]
        if self.ballrect.top < 0 or self.ballrect.bottom > height:
            self.velocity[1] = -self.velocity[1]
        surface.blit (self.ball, self.ballrect)
```

Extending last weeks tutorial to use classes

```
width = 1900
height= 1000
black = (0, 0, 0)

pygame.init ()
screen = pygame.display.set_mode([width, height])

bl = []
for i in range (80):
    bl += [ball ([random.randint (0, width-1),
                  random.randint (0, height-1)],
                 [random.randint(-2, 2),
                  random.randint(-2, 2)])]
```

Extending last weeks tutorial to use classes

```
while True:
    for event in pygame.event.get ():
        if event.type == pygame.QUIT:
            sys.exit(0)

    screen.fill (black)
    for b in bl:
        b.update (screen)
    pygame.display.flip()
```