Science & Technology
Facilities Council

pygame – Lesson 2

Adrian Coveney
Scientific Computing Department

The Basics

1. Import and initialise pygame
2. Create a game window - your starting point
3. Add a Game Loop

Some useful things …

- PyGame has some methods that can be used to find the edges of objects

```
rectangle = pygame.Rect(0, 0, 60, 60)
pygame.draw.rect(windowSurface, RED, rectangle)

rectangle.left
rectangle.right
rectangle.top
rectangle.bottom
```

Animations

We can use the module ‘time’ to create animations

```
clock = pygame.time.Clock()

# run the game loop
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
    ## ANIMATION CODE IN HERE!
    pygame.display.update()
    clock.tick(30)
```

Keyboard Input

```
# run the game loop
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
        elif event.type == pygame.KEYDOWN:
                if event.key == pygame.K_ESCAPE:
                        pygame.quit()

    ## ANIMATION CODE IN HERE!
    pygame.display.update()
    clock.tick(30)
```

Detecting when a key is held

`pygame.key.get_pressed()` returns a list of all keys currently pressed.

`pygame.key.get_pressed()[pygame.K_UP]` returns True if, and only if, the “Up” key is currently pressed/held down.

`pygame.K_UP` is the key constant value for the “Up” key. For a full list of these key constant values, see: https://www.pygame.org/docs/ref/key.html

- Draw a block on the screen
- Move the block across the screen
- Bounce the block off the sides of the screen
- Add another moving block
- Make the blocks bounce off one another

Angry Blocks

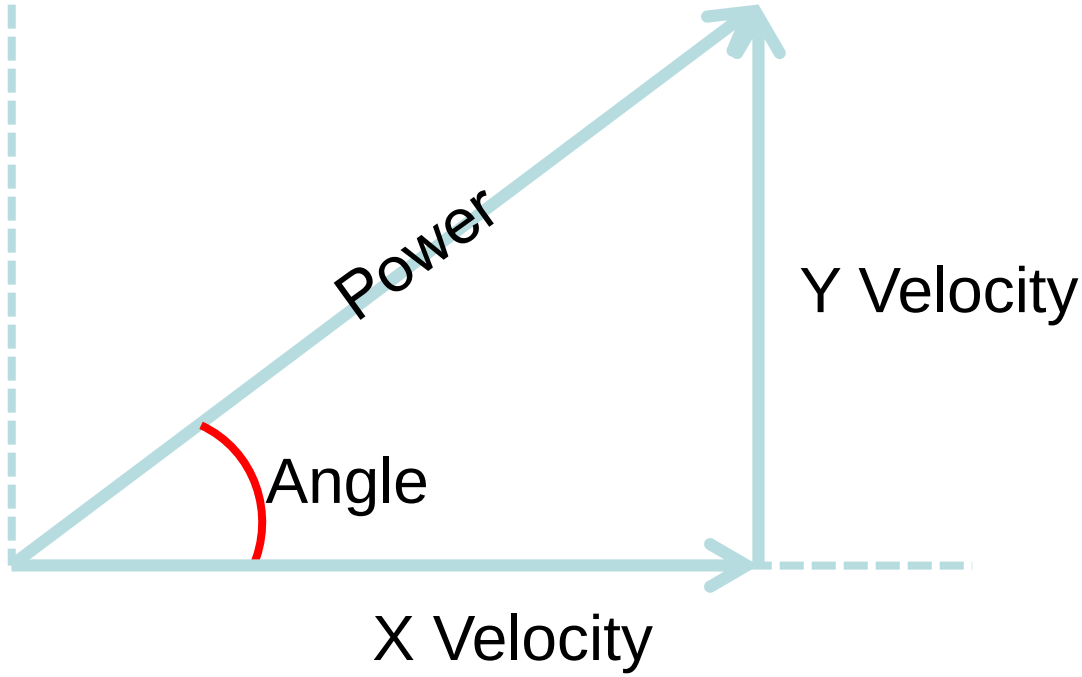
- Draw two boxes
- Create a function that draws the boxes
- Create a function that moves one box towards the other
- Only move the box when the space bar is pressed
- Output whether it is a hit or a miss
- Repeat the game until it is a hit

Angry Blocks

- Assumptions
  - No wind or air resistance
  - Gravity is fixed
- Start position
  - Bottom left corner: (0,screenHeight)
- Start Velocity
  - Based on start power and angle

Angry Blocks - Projectiles

Angry Blocks - Projectiles




```
angle = math.radians(45)
x_velocity = power * math.cos(angle)
y_velocity = - power * math.sin(angle)
```

Defining Gravity

- Velocity implemented by adding to position in loop

```
y_pos += y_vel
```

- Acceleration implements by adding to **velocity**

```
y_vel += y_accel
```

- Gravity is a downward acceleration of **2**

Leaving the screen

- If the projectile leaves the screen, something should happen.

- We could bounce, or reset the position…

```
if not windowSurface.get_rect().contains(projectile):
        x = 50
        y = 100
```

- Set the box's movement according to ANGLE, POWER and GRAVITY
- Change the POWER with the keyboard
- Show the POWER on screen

Angry Blocks