Welcome

Intro to Game Programming with

by Kevin Richey

Kevin Richey

- Fourth grade, graphics and games in BASIC

What About You?

- Favorite games
- Use a computer & keyboard
- Programming
- Python or PyGame

Hands-On Programming!

1. Python statements
2. Structure of a PyGame program
3. Screen & Drawing
4. Animation
5. User input
6. Collision detection
7. See a working game!

Edit “MyGame.py”

1. Double-click “IntroToPyGame-master”
2. Right-click “MyGame.py”
3. Select “Open”

1. Python Statements

- Assignment

```
name = something
```

- Function

```
function(param1, param2, …)
```

- Conditional

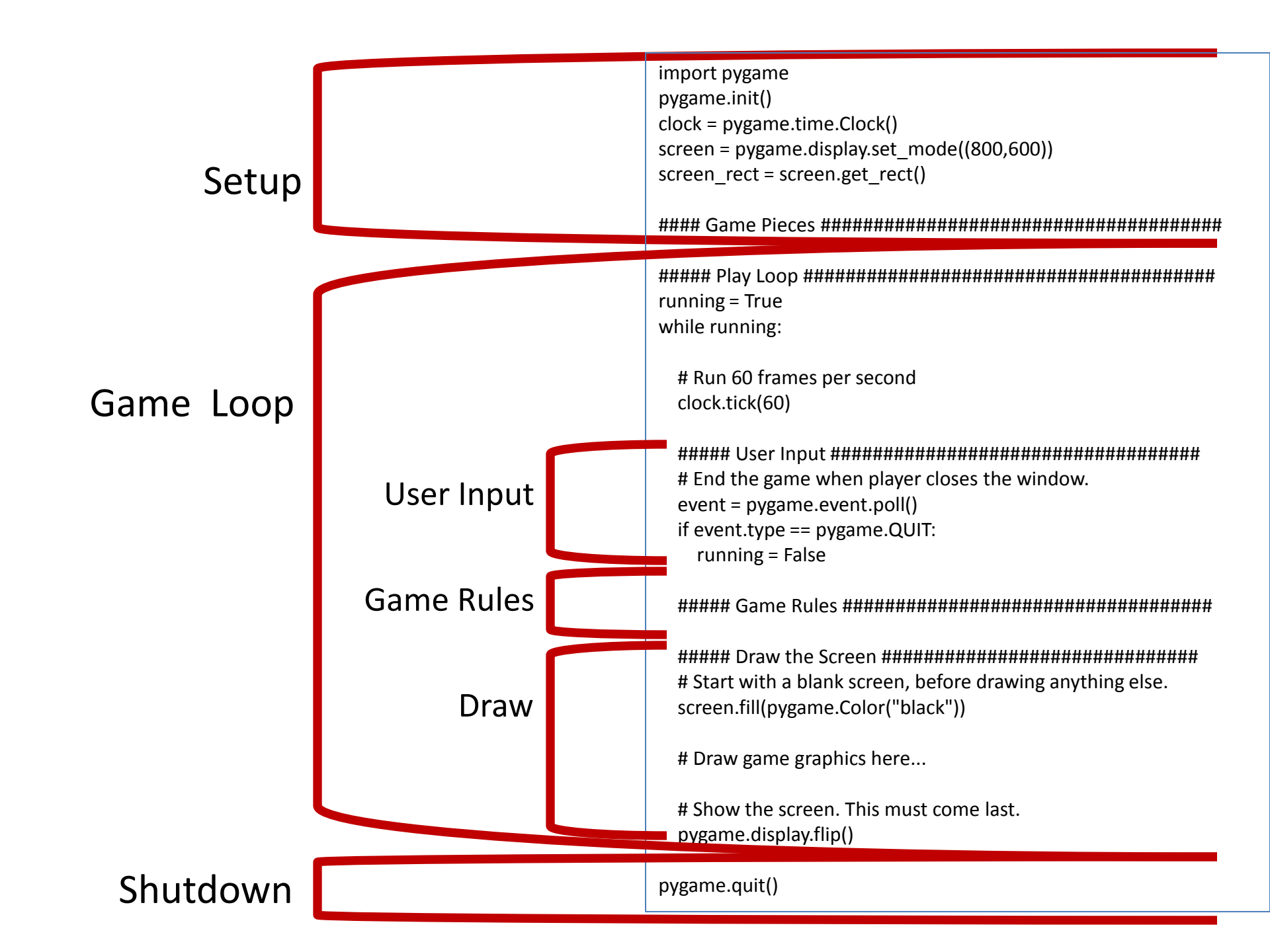
```
if condition is true:
    do this
elif condition is true:
    do this instead
```

- Loop

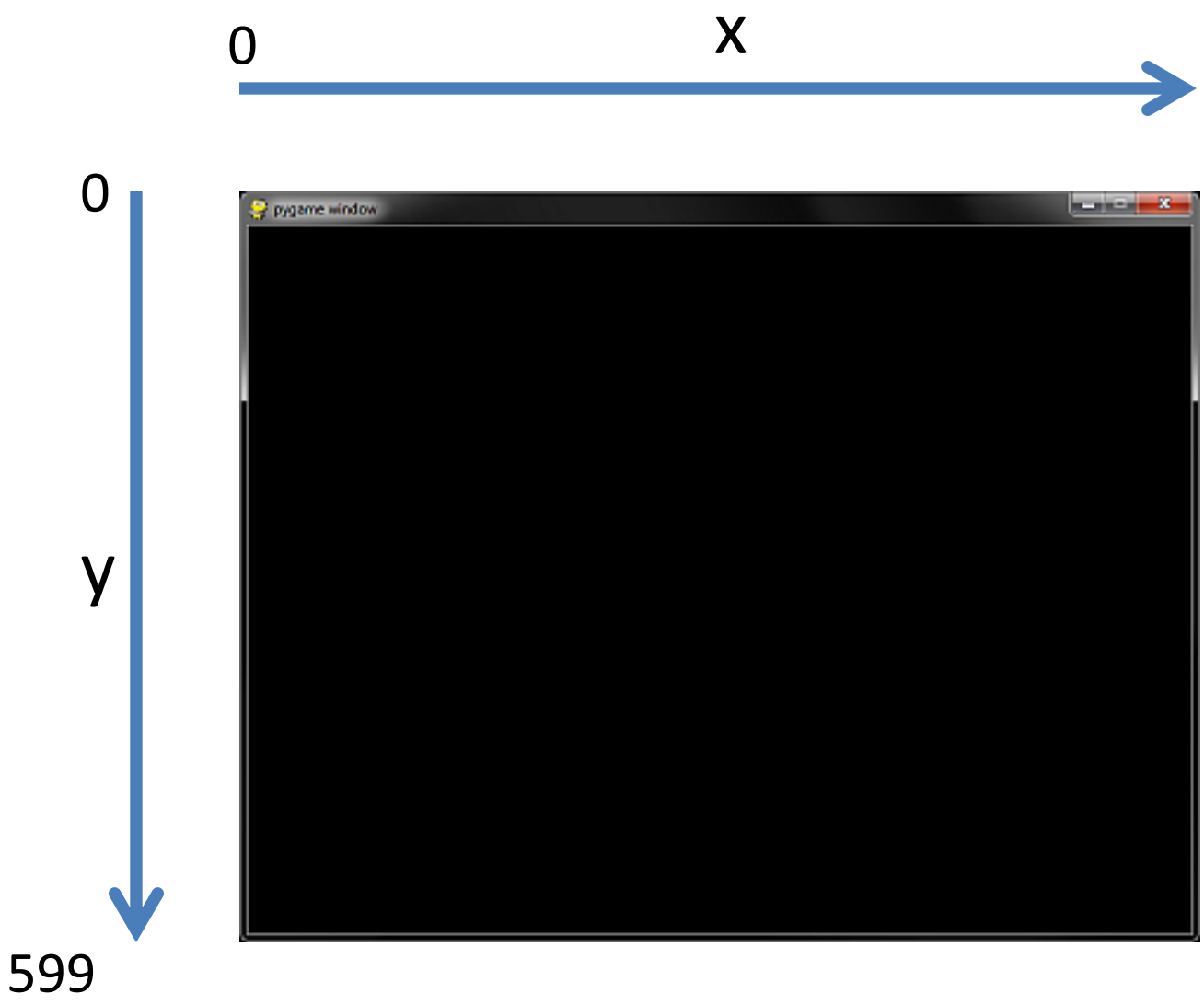
```
while condition is true:
    do this
```

Run “MyGame.py”

1. Double-click “MyGame.py” icon
2. Click “Run” button


Setup
Game Loop
User Input
Game Rules
Draw
Shutdown
import pygame
pygame.init()
clock = pygame.time.Clock()
screen = pygame.display.set_mode((800,600))
screen_rect = screen.get_rect()
Game Pieces
Play Loop
running = True
while running:
# Run 60 frames per second
clock.tick(60)
User Input
# End the game when player closes the window.
event = pygame.event.poll()
if event.type == pygame.QUIT:
running = False
Game Rules
Draw the Screen
# Start with a blank screen, before drawing anything else.
screen.fill(pygame.Color("black"))
# Draw game graphics here...
# Show the screen. This must come last.
pygame.display.flip()
pygame.quit()

screen = pygame.display.set_mode((800,600))



Unit: pixels

Discrete integers
(no decimals!)

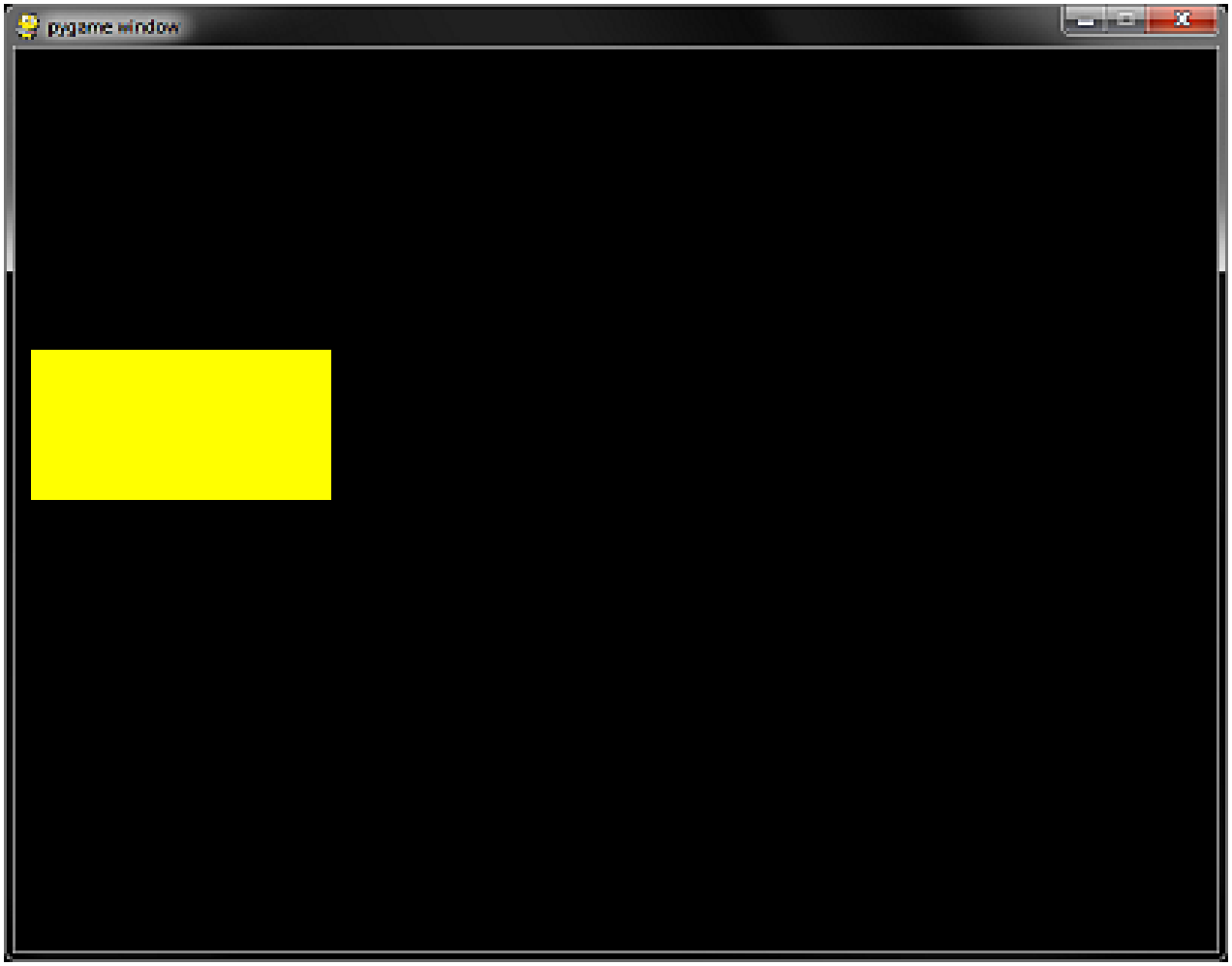
3. Drawing Graphics

```
#### Game Pieces ##############################
box = pygame.Rect(10, 200, 200, 100)
box_color = pygame.Color("yellow")
```

```
##### Draw the Screen ######################
# Start with a blank screen
screen.fill(pygame.Color("black"))

# Draw game graphics here...
pygame.draw.rect(screen, box_color, box)
```

pygame.draw.rect(screen, color, box)

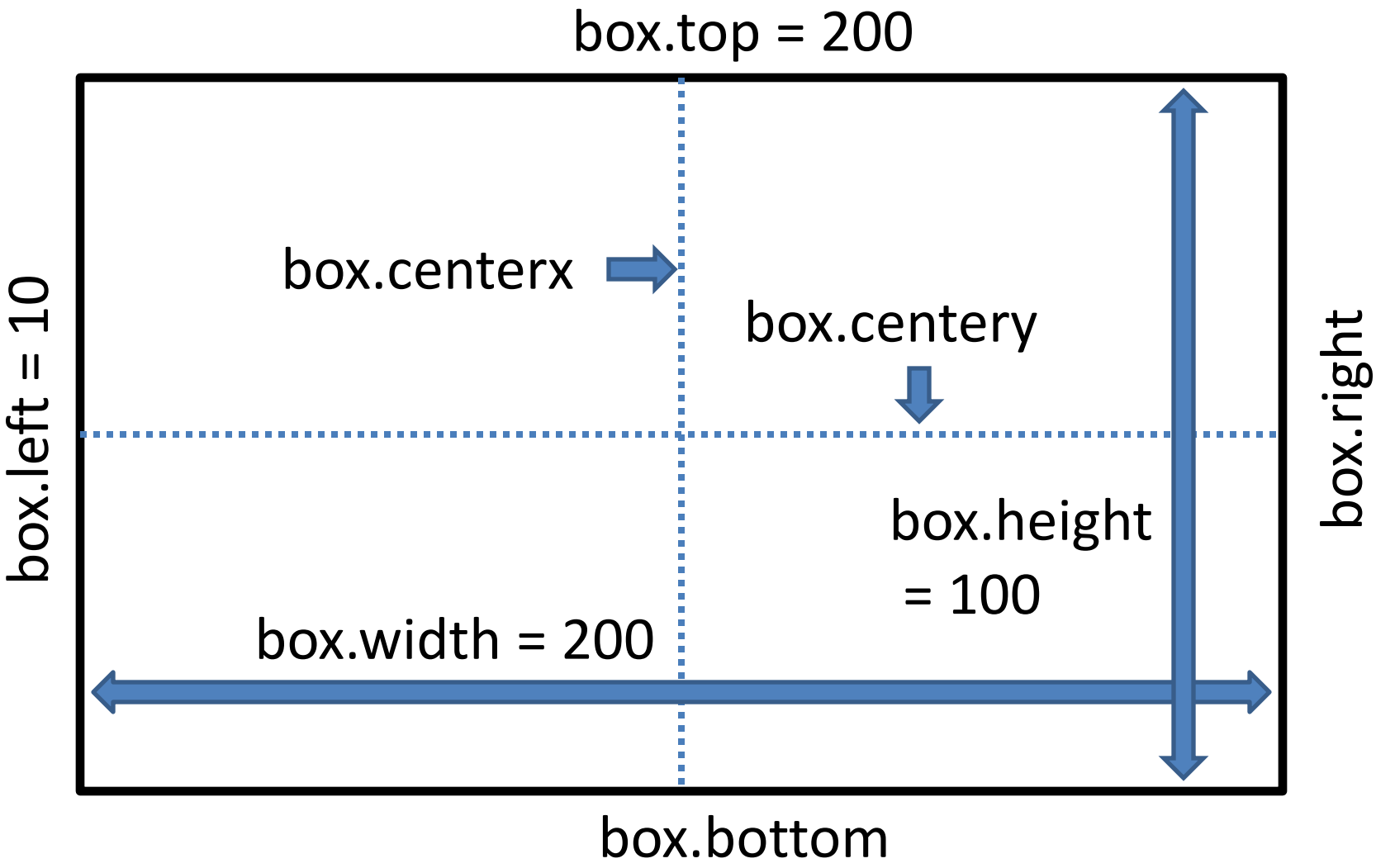


pygame.draw Functions

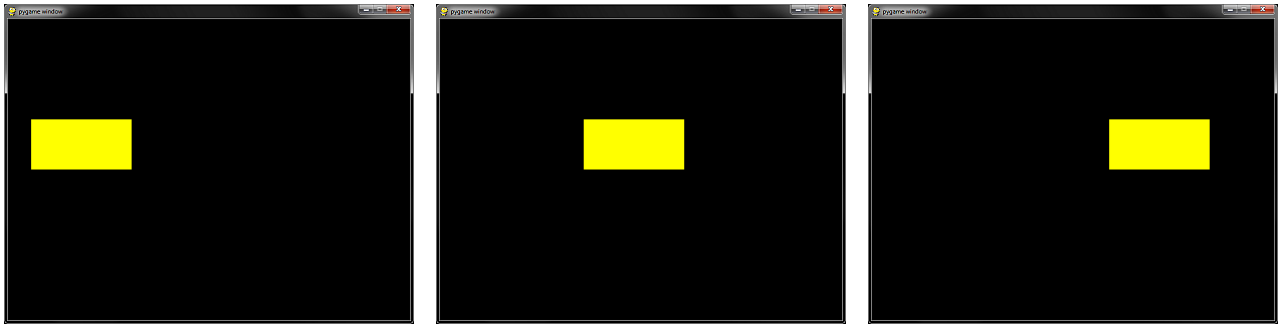
- `pygame.draw.rect()`
- `pygame.draw.polygon()`
- `pygame.draw.circle()`
- `pygame.draw.ellipse()`
- `pygame.draw.arc()`
- `pygame.draw.line()`
- `pygame.draw.lines()`
- `pygame.draw.aaline()`
- `pygame.draw.aalines()`

`http://www.pygame.org/docs/ref/draw.html`
or Google “pygame.draw”

```
box = pygame.Rect(10, 200, 200, 100)
```



4. Animation!



```
#### Game Pieces ##############################
box = pygame.Rect(10, 200, 200, 100)
box_color = pygame.Color("yellow")
box_speed = 3
```

```
    ##### Game Rules ############################
    box.centerx += box_speed


    ##### Draw the Screen ######################
```

Clear, Draw, & Flip

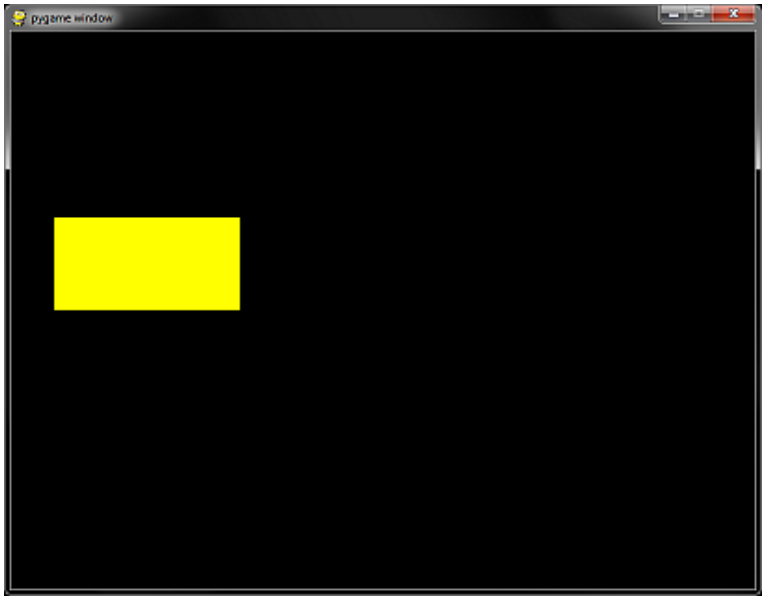
```
##### Draw the Screen ######################
# Start with a blank screen
screen.fill(pygame.Color("black"))

# Draw game graphics here...
pygame.draw.rect(screen, color, box)

# Show the screen. This must come last.
pygame.display.flip()
```

Double Buffer

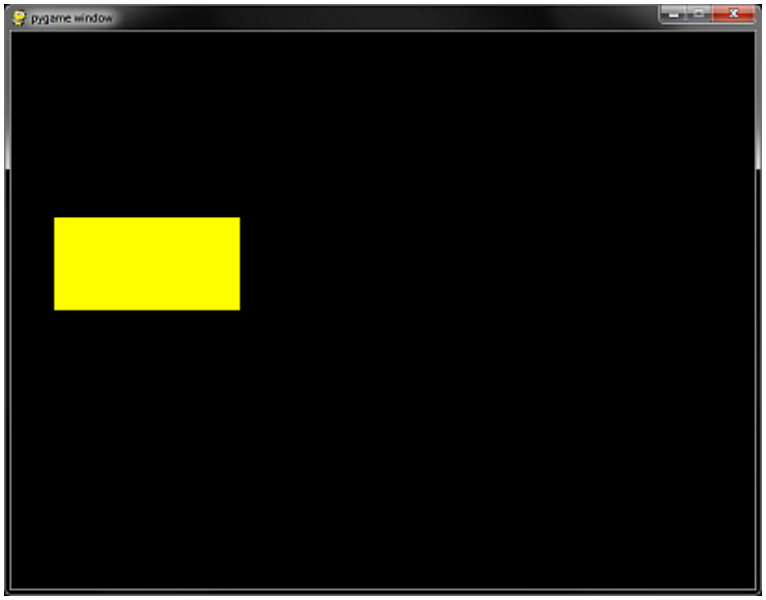
Hidden surface



Visible window

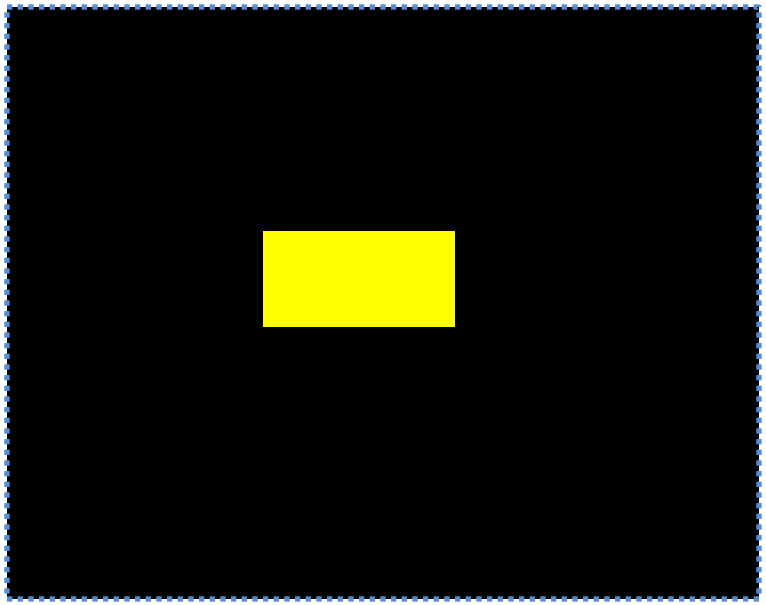
```
screen.fill(pygame.Color("black"))
```

Hidden surface

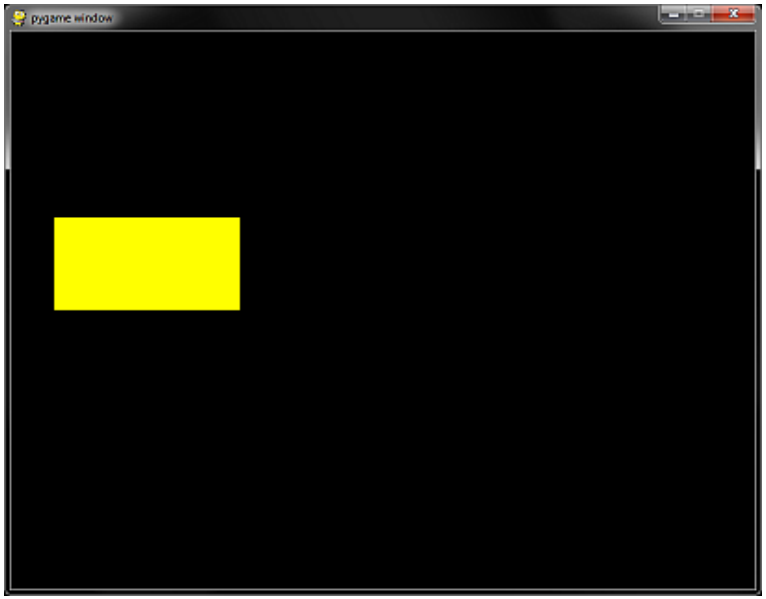


Visible window

```
pygame.draw.rect(screen, color, box)
```

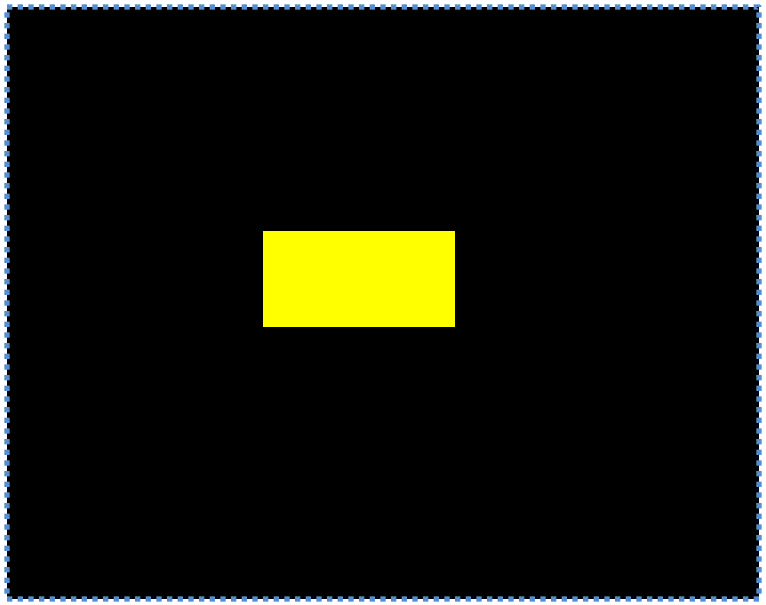


Hidden surface

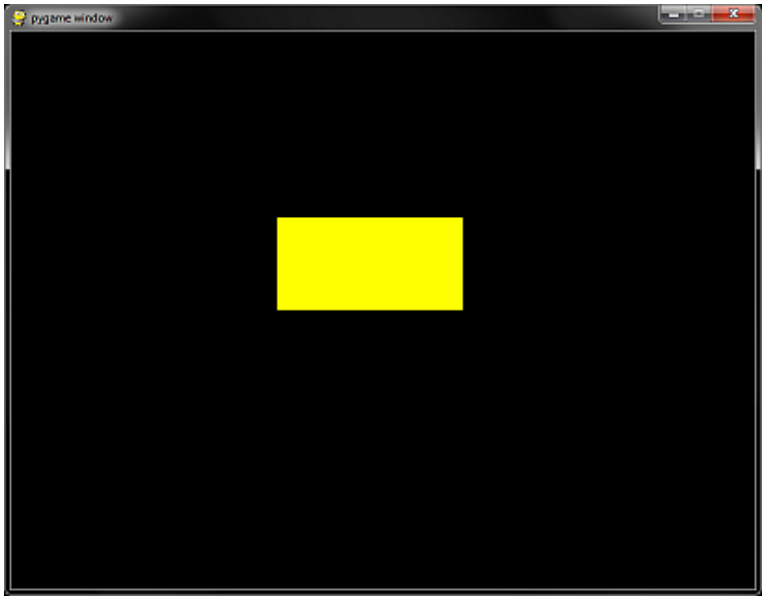


Visible window

`pygame.display.flip()`

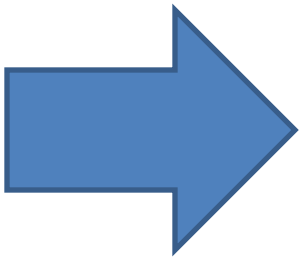


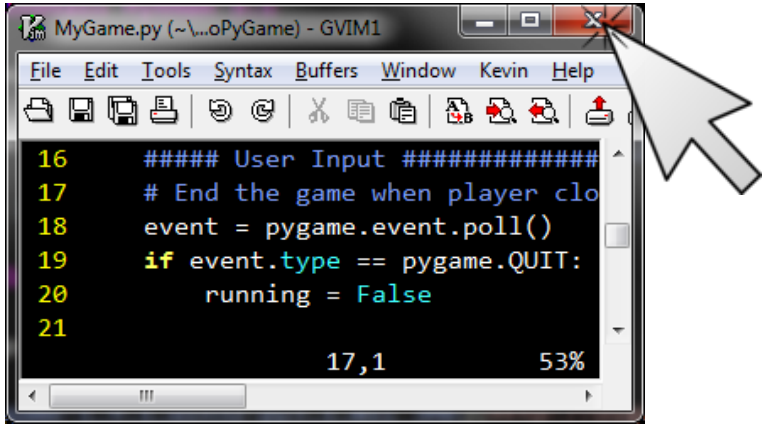
Hidden surface



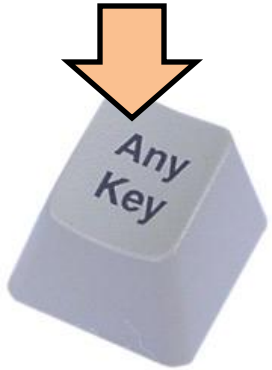
Visible window

5. User Input

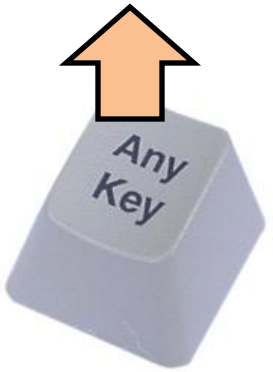




QUIT



KEYDOWN



KEYUP

```
#### Game Pieces ##############################
box = pygame.Rect(10, 200, 200, 100)
color = pygame.Color("yellow")
box_speed = 0
```

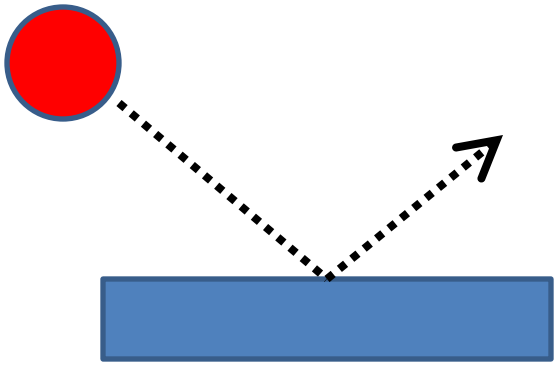
```
    ##### User Input ############################
    # End the game when player closes the window.
    event = pygame.event.poll()
    if event.type == pygame.QUIT:
        running = False
    elif event.type == pygame.KEYDOWN:
        box_speed = 3
    elif event.type == pygame.KEYUP:
        box_speed = 0
```

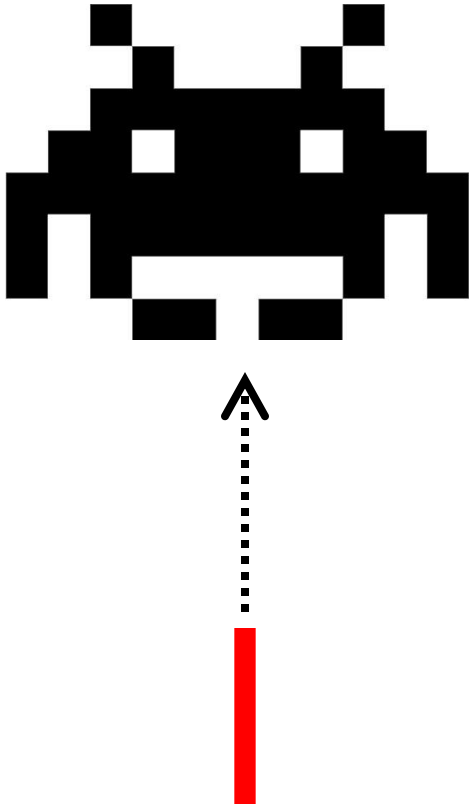
Reading Keyboard Arrow Keys

```
elif event.type == pygame.KEYDOWN:
    if event.key == pygame.K_LEFT:
        ball_dx = -ball_speed
    elif event.key == pygame.K_RIGHT:
        ball_dx = ball_speed
    elif event.key == pygame.K_UP:
        ball_dy = -ball_speed
    elif event.key == pygame.K_DOWN:
        ball_dy = ball_speed

elif event.type == pygame.KEYUP:
    if event.key == pygame.K_LEFT or event.key == pygame.K_RIGHT:
        ball_dx = 0
    elif event.key == pygame.K_UP or event.key == pygame.K_DOWN:
        ball_dy = 0
```

6. Collision Detection



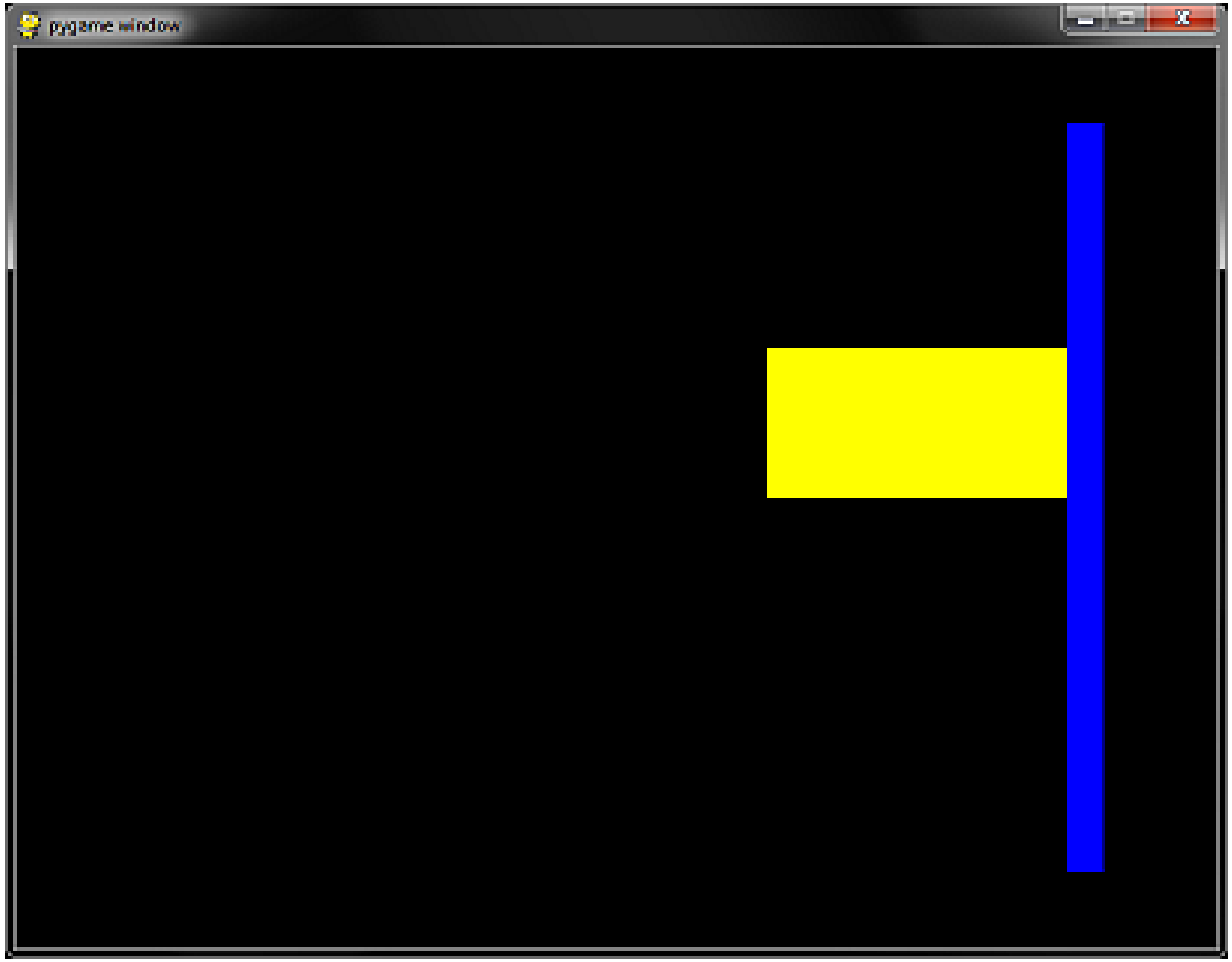


```
#### Game Pieces ##############################
box = pygame.Rect(10, 200, 200, 100)
box_color = pygame.Color("yellow")
box_speed = 0
wall = pygame.Rect(700, 50, 25, 500)
wall_color = pygame.Color("blue")
```

```
    ##### Game Rules ############################
    box.centerx += box_speed
    if box.colliderect(wall):
        box.right = wall.left
```

```
    # Draw game graphics here...
    pygame.draw.rect(screen, box_color, box)
    pygame.draw.rect(screen, wall_color, wall)
```

box.colliderect(wall)



7. Edit “blockout.py”

1. Double-click “IntroToPyGame-master”
2. Right-click “blockout.py”
3. Select “Open”

Change the Ball and Paddle

The Blockout game isn't much fun yet. Try to improve it by changing the ball and paddle.

1. Find the "Constants" section near the top of the file.
2. Type new color names inside the quotation marks.
3. Change the numbers after **ball_size =** and **ball_speed =**.
4. Change the numbers after **paddle_size =** and **paddle_speed =**.
5. Play the game after each change to see how it works.

Q. How do these changes affect game play?

Q. Which values do you like?

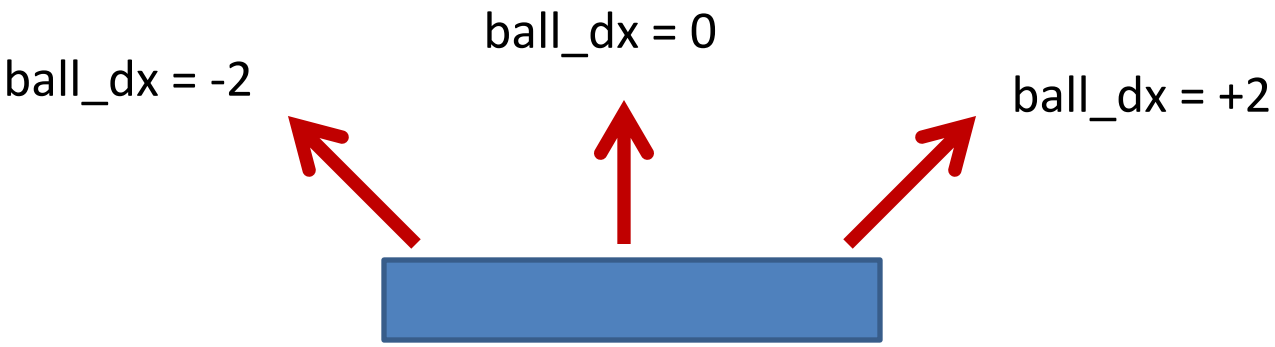
Keeping Score

Make each block worth 10 points and show the player's score on the screen. This requires adding multiple lines of code in different parts of the game program.

1. Create a score counter in "Game Pieces", equal to zero.
2. When the "Ball hits a block", increase the score counter by some points.
3. Under "Draw the Screen", use the draw_text() function to display the score in the top-left corner. You will have to convert the score number into text using the str() function, like this: "str(score)".

Aiming the Ball with the Paddle

Change the angle of the ball when it hits different parts of the paddle. Under “Bounce ball off the paddle”, set the **ball_dx** value between -2 and +2.



Paddle Speed Control

Make the paddle move faster when player holds down the Shift key. The `event.key` values for left and right shift keys are `pygame.K_LSHIFT` and `pygame.K_RSHIFT`.

Extra Lives

Give the player three lives. When the ball hits the bottom, subtract one life and restart the ball in the center. The game is over when the player has zero lives.

Increasing Difficulty

Every time the ball hits a block, shrink the paddle (`paddle.width -= 2`). This will make the game progressively harder.

Try it at Home

1. Install Python 3.2.5:
   https://www.python.org/download/releases/3.2.5/
2. Install PyGame:
   http://www.pygame.org/download.shtml
3. Get the code for this class:
   https://github.com/LetsCodeBlacksburg/IntroToPyGame
4. Learn more
   Free Books: inventwithpython.com
   Python: https://docs.python.org/3.2/
   PyGame: http://www.pygame.org/docs/

The End

HAVE FUN !