Installation

Here are instructions to download VSCode, Python and PyGame

Installing vscode

If you already have VSCode, or use a different editor, you can skip this step

Windows

Download VSCode from its website, and follow the instructions to install https://code.visualstudio.com/Download

Linux

Install through your package manager. Find further instructions on this page: https://code.visualstudio.com/docs/setup/linux

Installing Python

Windows

There are multiple ways of getting python on Windows
- Download it through the microsoft store
- Download through the Python website https://www.python.org/downloads/

Linux

Open the commandline and run the following command

```
$ python --version
```

if there are no errors, congrats! You already have Python installed.

otherwise, install through your distribution's package manager.

```
$ sudo pacman -S python
```

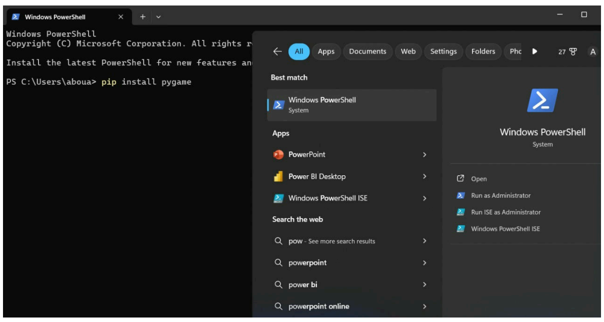
Installing PyGame

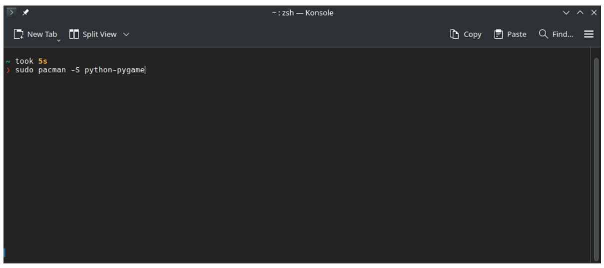
Open a terminal window (or powershell on windows), and type the following to install PyGame through pip

```
$ pip install pygame
```

or

```
$ python3 -m pip install -U pygame
```





Or install the package, python-pygame on linux.

Need help with Python?

Check the Python primer in the appendix if you need to get familiar with python basics. And, the cheatsheet if you need a reminder.

Quickstart

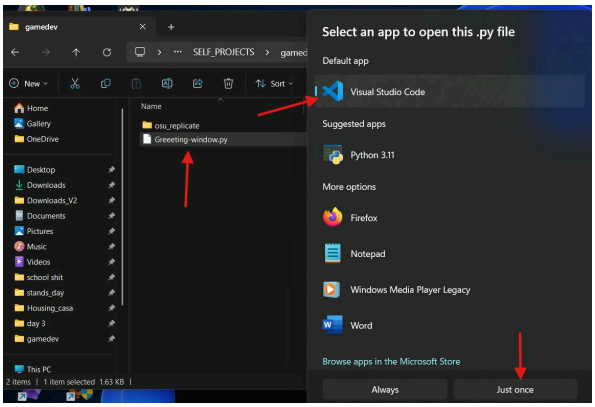
Let's run our first Python PyGame script! This script will render a black window and display the text "Hello, world!". Check the Greeting-window.py file. And run it.

```
$ python Greeting-window.py
```

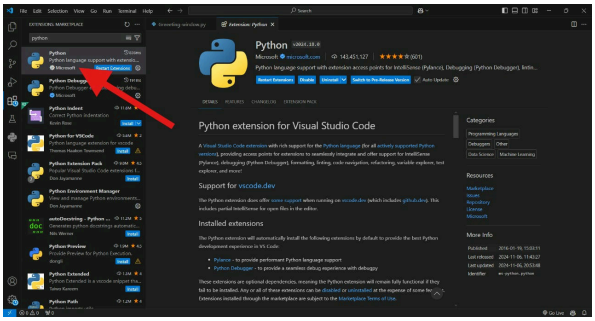
Hello world

If you don't know how to run a python script try this method (vscode required):

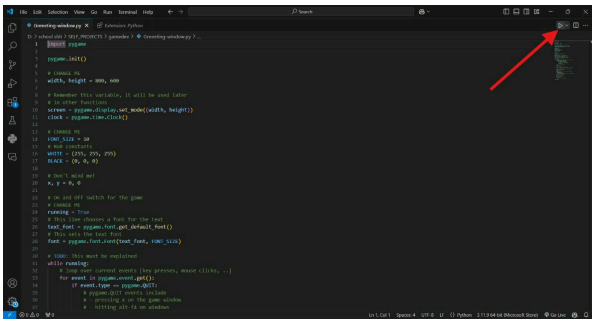
1. Open The script in VSCode. One way to do this is to find the script in file explorer and choose the 'open with' option.



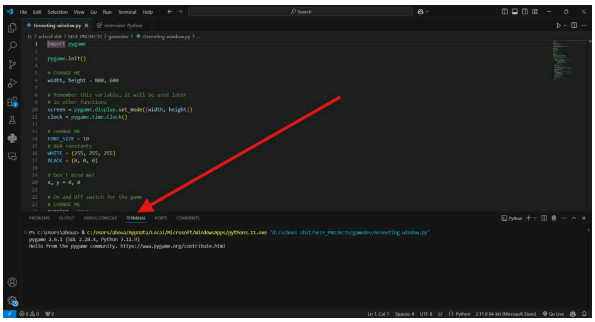
2. Install the python extension if you don't have it yet



3. Go back to the script editor and press the run button



4. it should open a terminal window a greeting from the Pygame community. Then a window greeting the world.



In this script you'll find comments explaining each line. Try commenting or deleting a line to find out what it does. And, let your team know!

For example: try changing the width here to a different value

- From this

```
# CHANGE ME
width = 800
```

- To this

```
# CHANGE ME
width = 1000,
```

Coordinates

In PyGame (and in game development in general), you will find the coordinate system works as described in the figure below.

- The horizontal x coordinates start at 0 at the left of the screen and increase to the right.
- The horizontal y coordinates start at 0 at the top of the screen and increase downwards.

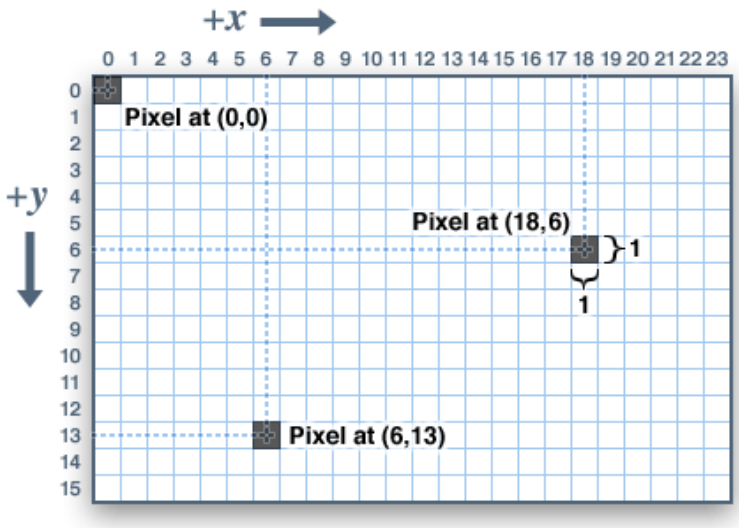


Figure 6: PyGame coordinate system. [1]

Shapes ＼ □ ○

Let's learn to draw a few shapes in pygame.

Line

To draw a line, use the following function, with the following parameters.

```
pygame.draw.line(screen, color, (start_x, start_y), (end_x, end_y))
```

- screen: pygame screen surface. In the previous example, it was defined as `screen = pygame.display.set_mode((width, height))`
- color: The color of the line. An RGB value defined like so: `WHITE = (255, 255, 255) #  (255 Red, 255 Green, 255 Blue)`
- Start: x and y of one end of the line
- end: x and y of the other end

```
pygame.draw.line(screen, color, (start_x, start_y), (end_x, end_y))
```

In the example below, start_x = 300, start_y = 200

```
while running:
  # preceding code here...
  screen.fill(BLACK)
  # ...
  # ADD THIS LINE
  pygame.draw.line(screen, WHITE, (300, 200), (500, 400))

  # ...
  pygame.display.flip()
```

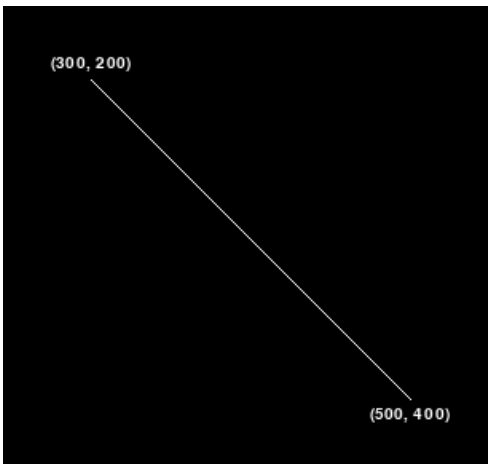


Figure 7: Line starting at (300, 200) and ending at (500, 400)

Circle

To draw a Circle, use the following function, with the following parameters.

```
pygame.draw.circle(screen, color, (center_x, center_y), radius)
```

- screen: pygame screen surface. In the previous example, it was defined as `screen = pygame.display.set_mode((width, height))`
- color: The color of the line. An RGB value defined like so `WHITE = (255, 255, 255) #  (255 Red, 255 Green, 255 Blue)`
- center: x and y of the circle center

- radius: value of the radius of the circle

`pygame.draw.circle(screen, color, (center_x, center_y), radius)`

In the example below, center_x = 400, center_y = 300, radius = 100

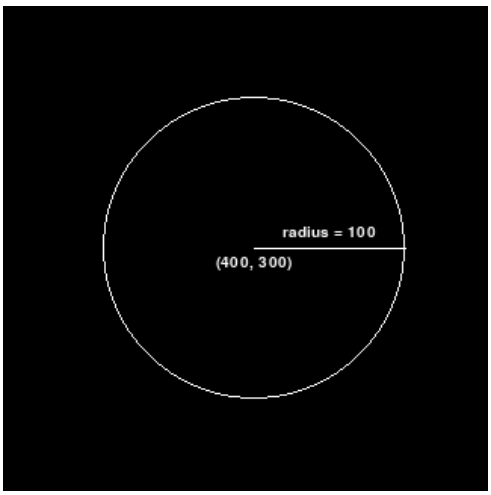


Figure 8: Circle centered at (400, 300) with a radius of 100

Random numbers 🎲

- To generate random numbers in Python, you can use the python random module. Here are a few examples:

```
import random

x = random.random()
print("random value from 0 to 1: ", x)
```

```
random value from 0 to 1: 0.2385292740937286
```

```
import random

# Returns a number between the two values provided
x = random.randrange(100, 200)
print("random value from 100 to 200: ", x)
```

```
random value from 100 to 200:  176
```

input

To get player input, you can use the pygame event system. There was already a demonstration of this system in the quickstart file.

```
while running:
    # loop over current events [key presses, mouse clicks, ..]
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            # pygame.QUIT events include
            # - pressing x on the game window
            # - hitting alt-f4 on windows
            running = False
```

Mouse Input 🖱️

Mouse click

To catch a mouse click event you can use the `pygame.event.MOUSEBUTTONDOWN` event.

- You may detect it in this way

```
while running:
    # loop over current events [key presses, mouse clicks, ..]
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            # pygame.QUIT events include
            # - pressing x on the game window
            # - hitting alt-f4 on windows
            running = False
        if event.type == pygame.event.MOUSEBUTTONDOWN:
            # Do something, like maybe draw a circle if the button was clicked?
            print("Clicked!")
    # ... rest of the code
```

Mouse position

To get the current mouse position on the screen (its x and y coordinates), you can use the `pygame.mouse.get_pos()` function.

- Here is an example

```
while running:
    # ... event code
    mouse_position = pygame.mouse.get_pos()
    # ... rest of the code
```

Experiment! 🧪 🔬 👩‍🔬 ⚗️

If you haven't looked at and changed the original code, now is the best time to do so! Once you are satisfied, you can move on to the next section

Objectives

You've already played or watched Osu (the game) being played. Your mission now is to replicate it. Let's begin with something simple.

- Make a circle spawn on a random screen position.
- Make the circle disappear if clicked.
- Make the circle disappear after some time has passed.

UI and Gamefeel

How should the player know if they managed to click the circle in time? Maybe add some visuals to show that they did?

Hints

A few hints:

- You can make a random position with two random numbers (a random x and a random y)
- check if the mouse is inside a circle
- Maybe first work with a single circle

If you still can't figure it out, Discuss it with your team, or with the coaches.

Counting frames

Video games work by periodically drawing on the screen. Each of these drawings is called a **frame**. So, in our example, we draw the circles on every frame. In your pygame project, what happens inside a frame is what happens during one iteration of the game loop

```
while running:
  # Code that runs during a frame...

  # This tells the game to draw 60 frames every second
  # 60 frames-per-second (or FPS for short).
  pygame.clock.tick(60)
```

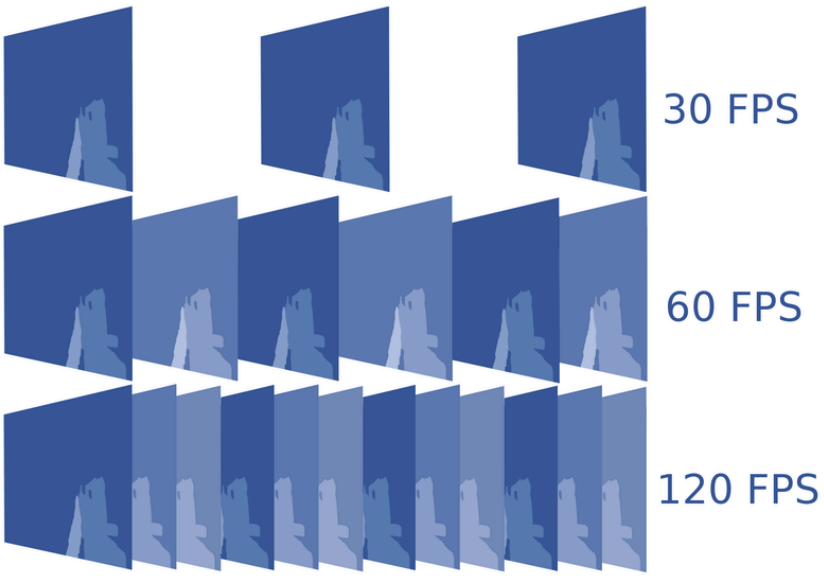


Figure 9: FPS demonstration. [2]

Appendix

Python primer

Variables

In python you can assign variables like so

```
# VARIABLE_NAME = VALUE
x = 16
```

Output

writing to the terminal in Python can be done like this

```
# print(VALUE)
print("hello world")
```

Operations

Basic math operations in python

```
x = 16
y = 32
addition = x + y
subtraction = x - y
multiplication = x * y
division = x / y
modulo = x % y # remainder of division of x by y
```

if statements and boolean expressions

Basic logic with conditional statements

```
x = 16
y = 32
# if CONDITION :
# The lines inside the if statement branch are indented
if x == y:
    print("then x is equal to y")
else:
    print("x is not equal to y")

if x * 2 == y:
    print("then y is equal to double x")
```

Loops and lists

python supports while and for loop syntax

```
# This is a list
x = [1, 2, 3, 4]
for i in x:
    print(i)

# Try me out, iterate 4 times, with i taking on the values, 0, 1, 2, 3
for i in range(4):
    print(i)

while i < 4:
    print(i)
    print(x[i])
```

Cheatsheet

- Python Cheatsheet at https://perso.limsi.fr/pointal/_media/python:cours:mementopython3-english.pdf

Bibliography

[1] “Adafruit GFX Graphics Library | Coordinate System and Units.” [Online]. Available: https://learn.adafruit.com/adafruit-gfx-graphics-library/coordinate-system-and-units

[2] “PCMag | How to See Your Frames Per Second (FPS) in Games.” [Online]. Available: https://me.pcmag.com/en/pc-games/17582/performance-over-fidelity-how-to-see-your-frames-per-second-fps-in-games