**Unit 8:** Game Programming in Python

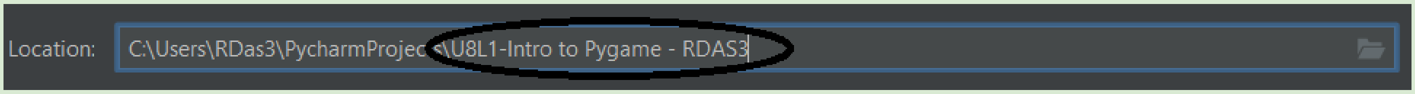
# Lesson 1: Intro to Pygame (and PyCharm)

| **Name:** | |
|---|---|

## PyCharm Tutorial

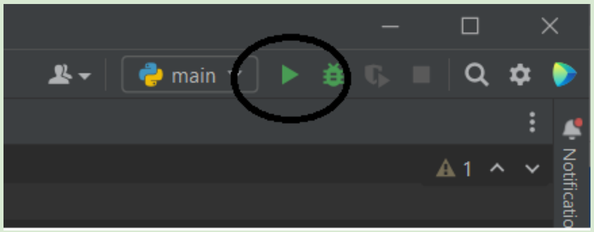
**1.** Open up PyCharm on your desktop. You may need to click "Agree" and go through a few other dialogues. After that, you should see an open to create a **New Project**

**2.** Name your project "U8L1-Intro to Pygame-YOURNAME"
(First Initial, Last Name) ***as long as it's unique!***

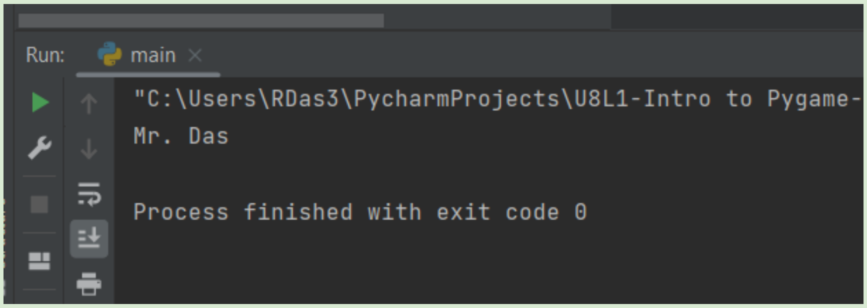

Location: C:\Users\RDas3\PycharmProjects\U8L1-Intro to Pygame - RDAS3

**3.** Your **main.py** file will have a bunch of stuff in it. Delete it and replace it with a simple print statement to print your name.

**4.** Run your program to make sure everything works! At the top right, you should see a Green Arrow. Click it to run your program.



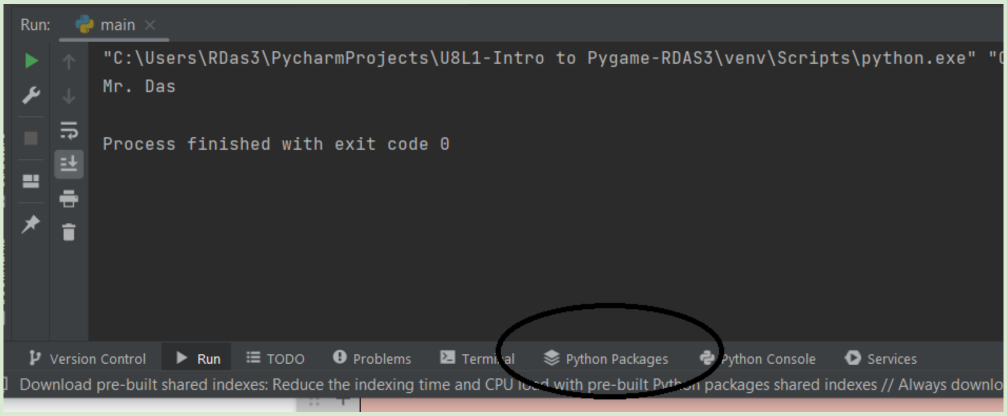
**5.** The output of your program will be at the bottom of the window.


Run: main

```
"C:\Users\RDas3\PycharmProjects\U8L1-Intro to Pygame-
Mr. Das

Process finished with exit code 0
```

**6.** Final step! For each project, you'll have to install **Pygame**. Go to the bottom of the screen and click on **Python Packages**.

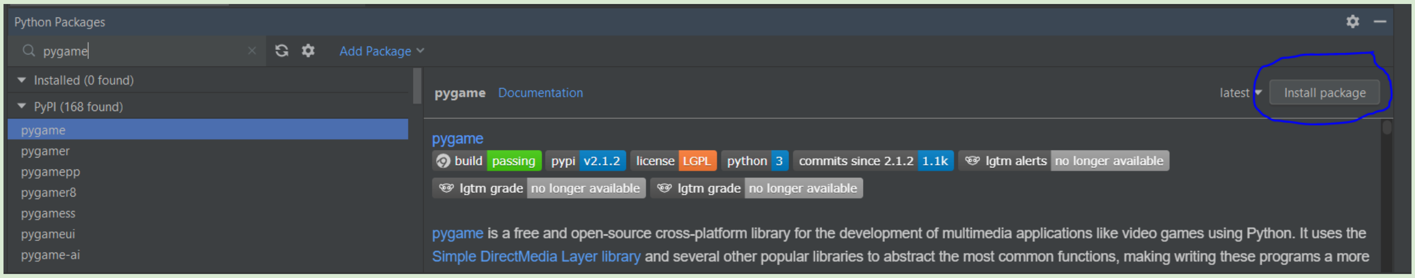


```
"C:\Users\RDas3\PycharmProjects\U8L1-Intro to Pygame-RDAS3\venv\Scripts\python.exe" "C
Mr. Das

Process finished with exit code 0
```

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always downlo

Click that, then type "pygame" in the search field. Once it shows up, click the **Install Package** button on the right.



Python Packages

pygame

Installed (0 found)

PyPI (168 found)

- pygame
- pygamer
- pygamepp
- pygamer8
- pygamess
- pygameui
- pygame-ai

pygame Documentation

pygame

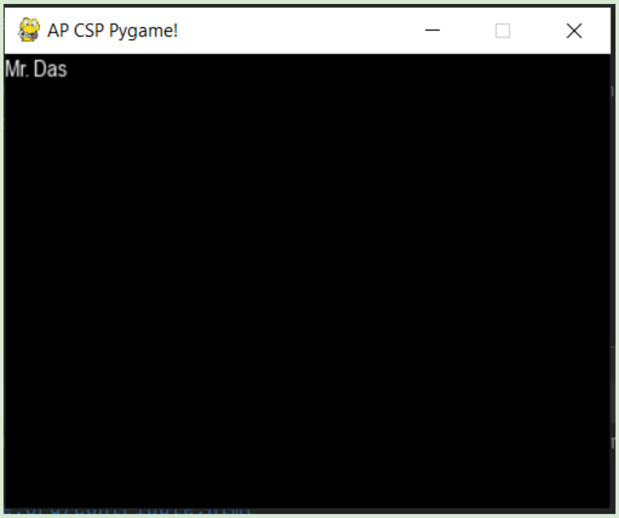
build passing | pypi v2.1.2 | license LGPL | python 3 | commits since 2.1.2 1.1k | lgtm alerts no longer available | lgtm grade no longer available | lgtm grade no longer available

pygame is a free and open-source cross-platform library for the development of multimedia applications like video games using Python. It uses the Simple DirectMedia Layer library and several other popular libraries to abstract the most common functions, making writing these programs a more

Now you're ready to go!!! **Remember** that all of your code is only on the school computer. Make sure to **save** your code (you can either do it on replit or just create **Google Doc** with your code in it).

## Creating a Window and "Drawing" Text

Our goal today is to create a basic Window using pygame, draw some text on it and experiment with **Fonts**, **Background Colors** and **Text Positioning!** Here is an example of what this guided example will walk you through:



AP CSP Pygame!

Mr. Das

**1.** In your *PyCharm* project, remove the initial print line that you added previously.

The first thing you'll need to do is add the following lines of code (you can copy and paste it!)

```
import pygame

# set up pygame modules
pygame.init()
pygame.font.init()
font = pygame.font.SysFont('Arial', 15)
pygame.display.set_caption("AP CSP Pygame!")

# set up variables for the display
size = (400, 300)
screen = pygame.display.set_mode(size)
```

Every **pygame** project you will write should start like this. The only thing that may change is the **Font** and **Font Size** you want to use, along with the size of the screen.

**2. Rectangles, Rectangles, Rectangles!!** Everything that gets drawn to a window in pygame is represented by a **Rectangle** object (even text).

So, to create a text display at the top right of the screen, we first need to "render" a **Rectangle** object like so (change the name to your name!):

```
display_name = font.render("Mr. Das", True, (255, 255, 255))
```

`display_name` represents a **Rectangle** object.

The `render` function takes in three arguments:

1. The `String` you want to render (in this case, "Mr. Das")
2. A boolean value that allows you to **smooth** out the font so it looks nicer (we should really always set this to `True`).
3. A tuple that represents the RGB value for the text (in this case, we are making the white).

**3. Finally!** The main program loop! Every pygame program will have a **main loop** and what is called your "event loop". In this case, the main loop will stop when the user closes the window. We will talk more about the event loop in a later lesson. Copy and paste this code into your program.

```
run = True

# -------- Main Program Loop -----------
while run:
   screen.fill((0, 0, 0))
   screen.blit(display_name, (0, 0))
   # --- Main event loop
   for event in pygame.event.get():  # User did something
       if event.type == pygame.QUIT:  # If user clicked close
           run = False
   pygame.display.update()
```

```
# Once we have exited the main program loop we can stop the game engine:
pygame.quit()
```

**4.** Understanding `screen.fill` and `screen.blit`

**`screen`** is created earlier in the program and refers to the main window of our program.

The **`fill`** function allows us to set a background color for the window (here it's black).

The **`blit`** function allows us to place **Rectangles** that we've previously rendered onto our main surface. Here, we are rendering `display_name` onto the position 0,0 (the top left of the window is 0, 0).

`pygame.display.update()` does a FULL refresh of the screen based on what you placed on it with `fill` and `blit`.

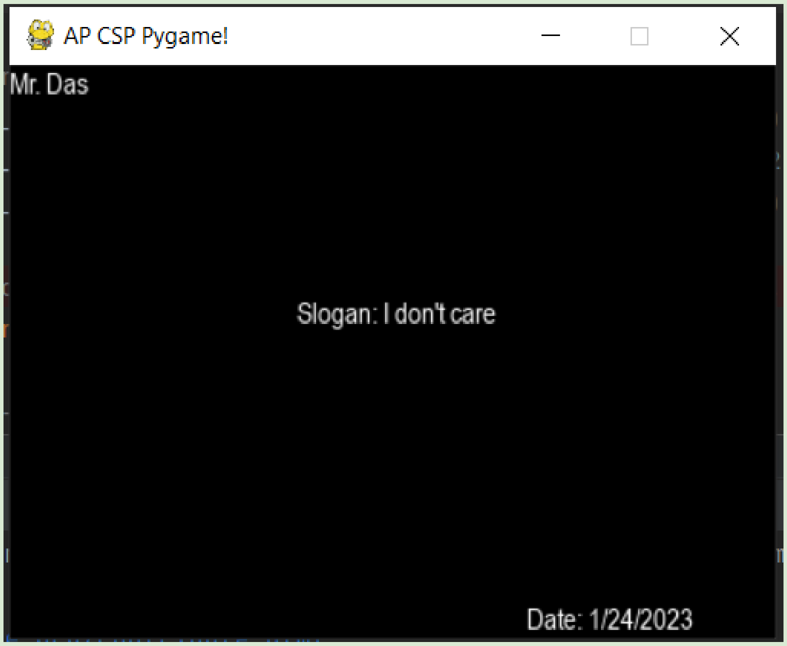
## Adding more to the window!

Your task is to add your "Slogan" and the date to the window and position them appropriately. Your name should already be at the top right. Add your slogan and position it in the center, add the date and position it in the bottom right.

You can get the current date in Python like this!

```
from datetime import date
current_date = "Date: " + str(date.today())
```

Your final window should look something like this:



Mr. Das

Slogan: I don't care

Date: 1/24/2023

Try experimenting with different background colors, fonts, font sizes and more!

**Take a screen snippet of your window and paste it here!**

→ **Done!** ←

Submit this document in Google Classroom