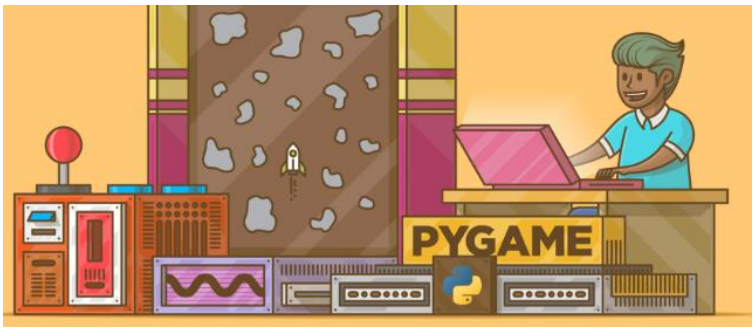
PyGame

Python's game building library.



PyGame is a platform that allows users to build **2D** games easily with custom **libraries**. It is a great way to learn important theory and ideas related to all game construction. With PyGame you can build complex/entertaining applications and:

a) Learning useful programming concepts, like **object-oriented programing**.
b) Learn new Python syntax.
c) Being creative.
d) Share usable, fun, and exciting projects with others.

Know what a **surface** is.

One of the most important concepts in the PyGame library is the **surface**. Just think of a **surface** as a blank piece of paper. You can do a lot of things with a surface -- you can draw lines on it, fill parts of it with color, copy images to and from it, and set or read individual pixel colors on it. A surface can be any size (within reason) and you can have as many of them as you like (again, within reason). One surface is special -- the one you create with: `pygame.display.set_mode()`. This **'display surface**' represents the screen; whatever you do to it will appear on the user's screen. You can only have **one** of these – this is not a pygame limitation, it a limitation of the operating system used to run your game.

Setting Up a Basic Pygame Screen Template:

```
import pygame, sys
from pygame.locals import QUIT

pygame.init()
game_display = pygame.display.set_mode((400, 300))
pygame.display.set_caption('My Awesome Game!')
color = (255,100,0)       # orange
game_display.fill(color) # Changing surface color
pygame.display.flip()
while True:
    for event in pygame.event.get():
        if event.type == QUIT: #pressing 'X' closes window (exits)
            pygame.quit()
            sys.exit()
    pygame.display.update()
```

Exercise #1

Enter the code you see into a pygame Replit and run the code to see what it does. **Then read the descriptions on the next page** to get a sense of what each line of code is doing.

```
import pygame, sys
from pygame.locals import QUIT

pygame.init()
game_display = pygame.display.set_mode((400, 300))
pygame.display.set_caption('My Awesome Game!')
color = (255,100,0)                    # orange
game_display.fill(color)               # Changing surface color
pygame.display.flip()
while True:
    for event in pygame.event.get():
        if event.type == QUIT:         #pressing 'X' closes window (exits)
            pygame.quit()
            sys.exit()
    pygame.display.update()
```

- **`import pygame`** and **`from pygame. locals import QUIT`** – Allows access to all the pygame functionalities and inner modules,

- **`pygame.init()`** – Initialize pygame to provide access to functions and automatically start up all the pygame modules that need to be initialized.

- **`pygame. display.set_mode((width, height))`** – It displays a window of a particular height and width.

- **`pygame. display.set_caption('My Awesome Game')`** – Adding the name of the game on the top of the title screen.

- **`flip()`** allows only a portion of the screen to updated, instead of the entire area. If no parameter is given in the brackets, it updates the entire Surface area like **`pygame.display.update()`**

- **`Function to handle the events`** – One needs to define a function to handle the events happening on the screen. For now, we take into consideration one event i.e. closing the window on pressing 'X' on the window.

- **`pygame.display.update()`** – It is used to make the necessary updates on the display.

- **`.Blit()`** draw an object on the screen a particular location

Exercise #2

Go to link on course page that says: **pygame template**. This will bring you to a min pygame program to help get you started on creating a game.

1. Run the program.
2. **Maximize the screen** and use the arrow keys to move the rocket around. Notice how the image will rotate 90 degrees as you mov it round. Find out where in the code this is happening.
3. Run the program again and notice the blue square moving across the top of the screen. How does this happen? Find it in the code.

   Hand the following in for Exercise#2:

4. Create your own game screen *using the template provided*. Your game screen should have its *own* **background** and **images**. Find and choose these carefully. Make your game screen look cool. Pick a theme that might be interesting, and **you can build upon**.

Exercise#3

Remove all the other rectangles form the template. Draw a green rectangle and make it move smoothly from the bottom right of your screen to the top left of your screen when your game screen starts. Look for the variable 'r' in the template code if you need a hint.

Exercise#4

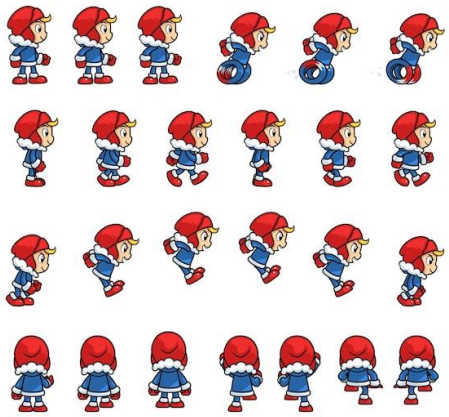
Remove all the rectangles from your screen. Ensure you can move at least one of your images around your screen **using the arrow keys**. See if you can make the motion look professional/cool, by **substituting in other images at appropriate times**. See the "***simple image substitution***" link on the course page for help.

Exercise#5

a) Import new image that stays stationary at the **center** of your screen. Now see if you can have the new image disappear *when the moveable object in Exercise#4 comes in contact with it*. See **collisions** video on course page.
b) Now see if you can add an "animation" of an explosion when you collide with the *center* image. Gif animations are not supported in Pygame, so you will have to create your own "animation" by *subbing in a series of images over time*. See the **simple image substitutions** link on the course page for help.

Exercise#6

One way to spice up your game is to use **sprites**. Sprites are a series of images that represent different "looks" of a game character (or other feature). Using these images in the correct sequence can make the motion of objects in your game look more realistic. Jumping, turning, etc, will all be represented by different images.



Use the **Sprite sheets and Pygame** link on the course page to help you improve the look of a character or object's motion.

Exercise#7

Use **the Scrolling background in Pygame** link on the course page to add the ability of your character/s to move past the current edge of your game display.

Exercise#8

Now you are ready to roll!...come up with an idea for your own game and use Pygame to create it. You can work with a partner, or come up with something on your own. **Please run your idea by Mr. Walzl before you start.**