PreLab13: Introduction to pygame library

Prelab policy

We encourage you to work together on the Pre Lab. The Pre Lab is not graded but will help you prepare for your lab session. In the Pre Lab, you may find questions to answer. We do not require you to provide us the answers, but we do recommend you to try to answer these questions. If you have any questions on the material in the pre lab, first check the book and recitation slides, if you do not find your answer please email your recitation TA or the course instructors.

Installing pygame on your personal machine

For Mac

0. Download and Install XQuartz: http://www.xquartz.org/ Restart your computer.

Open up a terminal:

1. Install homebrew:

```bash
ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

2. Install mercurial and SDL

```bash
brew install mercurial
brew install sdl sdl_image sdl_mixer sdl_ttf portmidi
```

3. Install pygame

```bash
pip3 install hg+http://bitbucket.org/pygame/pygame
```

For Windows

Go to this page: http://www.lfd.uci.edu/~gohlke/pythonlibs/#pygame

Install one of the following:

- pygame-1.9.2a0-cp34-none-win32.whl
- pygame-1.9.2a0-cp34-none-amd64.whl

according to whether you have a 32-bit or 64-bit Windows. (You can check that using right click to
your Computer and click Properties.)

Make sure you also have the correct version of Python 3.4 (32-bit, 64-bit) installed.

After you install, move the “.whl” file into your “C:/Python34/Scripts/”. Open up a terminal. (Windows + R) and write “cmd” and enter.

```
cd C:\Python34\Scripts
pip.exe install "pygame-1.9.2a0-cp34-none-win32.whl" (or the other one.)
```

You should see something like:

```
C:\Python34\Scripts>pip.exe install pygame-1.9.2a0-cp34-none-win_amd64.whl
Unpacking c:\python34\scripts\pygame-1.9.2a0-cp34-none-win_amd64.whl
Installing collected packages: pygame
Successfully installed pygame
```
Cleaning up...

If you have complications about installing pygame to your personal computer, please visit one of the office hours so we can help you out.

Remember that you can always use the labs (LWSN B160 and HAAS G056) to finish your project.

**Pygame test**

To test if pygame works in your machine, you can run this code:

```python
import pygame
import sys

pygame.init()
window = pygame.display.set_mode((600, 600))

while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            sys.exit()

    window.fill((0, 0, 0))  # black background
    pygame.draw.rect(window, (255, 255, 255), (250, 250, 100, 100))  # white square
    pygame.display.update()
```

Or similarly, [https://www.pygame.org/docs/ref/examples.html](https://www.pygame.org/docs/ref/examples.html)

**Drawing a square**

```python
import sys
import pygame

# some initializations
pygame.init()
window = pygame.display.set_mode((600, 600))
pygame.display.set_caption('First rectangle')
clock = pygame.time.Clock()

# color definitions, see RGB color model if this does not make sense:
```
black = (0,0,0)
white = (255,255,255)

while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            sys.exit()

    # draw and update screen
    window.fill( black )
    pygame.draw.rect( window, white, (250, 250, 100, 100) )
    pygame.display.update()

    # fps stuff:
    clock.tick(10)

pygame.quit()

Note that the above program produces the following output:

There are two important lines in the above program:
Create a window of size 600×600: \( \text{window} = \text{pygame.display.set_mode}\((600, 600)\) \\
Plot a rectangle with top-left corner = (250,250), width = 100, and height = 100: \( \text{pygame.draw.rect}(\text{window}, \text{white}, (250, 250, 100, 100)) \)

Please note that the official documentation of draw.rect() function is here: http://www.pygame.org/docs/ref/draw.html#pygame.draw.rect

**TODO: Drawing a square**

Update the above program and (1) Make the rectangle bigger, (2) Change the color of the rectangle, (3) Change the background of the window.

Here are the details:

1. Make the rectangle bigger (width=200, height=200), but still keep it exactly at the center of the screen: (HINT: Modify pygame.draw.rect() function.)
2. Change the color of the bigger rectangle to red: (HINT: Modify `pygame.draw.rect()` function)

3. Change the background to white: (HINT: Modify `window.fill()` function)
Drawing a chain of squares

Write a Python program using the pygame library that it draws a chain of squares on the screen with given size. Details:

- Window size is 600×600 again.
- We are going to draw n squares.
- All of the squares will be aligned to the screen and they will look like a chain.
- The space between rectangles and the width of rectangles are the same.

For example when n = 7:
Annotated version of the above Figure:
Moving a square using keyboard

Copy and run the following in IDLE to move the square on window using arrow-keys.

```python
import pygame

# some initializations
pygame.init()
window = pygame.display.set_mode((600,600))
pygame.display.set_caption('Rectangle move')
clock = pygame.time.Clock()

# color definitions, see RGB color model if this does not make sense:
# https://en.wikipedia.org/wiki/RGB_color_model
black = (0,0,0)
white = (255,255,255)
```
# initial center position for the square
x, y = 275, 275

game_loop = True
while game_loop:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            game_loop = False
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_LEFT:
                x -= 50
            if event.key == pygame.K_RIGHT:
                x += 50
            if event.key == pygame.K_DOWN:
                y += 50
            if event.key == pygame.K_UP:
                y -= 50

    # draw and update screen
    window.fill( black )
    pygame.draw.rect( window, white, (x, y, 50, 50) )
    pygame.display.update()

    # fps stuff:
    clock.tick(10)

    pygame.quit()