Simple Graphics With tkinter

This tutorial was created by **Matt Stephenson** in the 2009-2010 school year as his Peer Tutoring Project.

For those of you who have experience with drawing graphics with Turing, you will find some of the concepts similar when drawing graphics in Python. Graphics in Python are a little more complex than Turing because a new module called **tkinter** has to be imported. As well, until this point your programs have been displayed by the Python GUI (IDLE) window (screen) or Python Console. In Python, graphics must have our programs displayed on a different window (screen) called a canvas.

A simple graphics program

 # This is a program that draws a rectangle. Each line is explained below

from tkinter import *
root = **Tk**()
myCanvas = **Canvas** (**root**, **width** = 200, **height** = 200)
myCanvas.**pack** ()
myCanvas.**create_rectangle** (25,25, 150, 150, **outline** = 'blue', **fill** = 'red')
root.mainloop ()

Explanation of the new commands

Note that some of the commands may not make a lot of sense at this point as they refer to creating objects based on classes. We will discuss this further in Grade 12.

from tkinter import * *‘’‘ This command makes available all of the methods from the tkinter graphics library. Note that this style of importing means that when accessing items from the tkinter library, we do note need to state “tkinter.” (ie. Like you must when using items from math library, “math.sqrt”)’‘’*

root = **Tk()** *''' - Creates a tkinter object*
- “root” is our tkobject which we must "attach" our drawing canvas (window) to
*- any name can be used instead of root but use **root** to keep things simple*

myCanvas = **Canvas** (root, **width**=200, **height**=200) *''' this creates a canvas (drawing window) named “myCanvas” with a width and height of 200 pixels. Of course any name can be used instead of myCanvas '''*

myCanvas.**pack**() *'''this basically prepares the canvas window named “myCanvas” to be used/drawn onto. **If you do not put this line** nothing will be drawn. The canvas is now ready to be drawn on.'''*

myCanvas.**create_rectangle**(0, 0, 150, 150, **fill** = ’blue’, **outline** = ’red’)
''' This line actually draws the rectangle on the“myCanvas” window. The first 2 parameters are the top left corner of the rectangle (top left corner of canvas is coordinates 0,0) and the second two parameters are the bottom right corner. fill is the fill colour while outline is the colour of the outline of the rectangle

root.**mainloop** () *'''This is the "listening" loop.* ***Without this line at the end of the program, the canvas will not be opened and nothing will be drawn.*** *Up until this line, the program is simply preparing the canvas and the items to be drawn to it. Once the program reaches this line, the output is made visible to us.'''*

Drawing Other Shapes in tkinter

- "myCanvas" is replaced with whatever the name of your canvas is
- x1 and y1 is the first coordinate pair, x2 and y2 are the second.
- Options (such as fill,outline and width) affect the appearance of the shape. Use width = 0 to make the border invisible.

- Draw a straight line - myCanvas.**create_line** (x1, y1, x2, y2)
- Draw a rectangle - myCanvas.**create_rectangle** (x1, y1, x2, y2, options)
- Draw an oval - myCanvas.create_oval (x1, y1, x2, y2, options)
- Draw a text graphic - myCanvas.**create_text** (x1,y1, **text**="your text", **fill**='black', **font**=('verdana',20))

For a very comprehensive discussion of drawing on the canvas including what can be drawn and what options there are, use the following web site:

http://effbot.org/tkinterbook/canvas.htm

Assignment #1

Create a window with at least 8 different shapes with different fills and outlines.

Animation(Moving Graphics)

Many times when we are using graphics, we want the images to move in some way. For example, in Major Assignment #1, the extention to the horse race question asked you to create a graphical horse race. A simple program to make a rectangle (a "horse") move across the screen would look something like this:

```
from tkinter import * #Same as before
import time   # We must import the time library so we can use a delay later in the program

root = Tk ()              # Same as before
myCanvas = Canvas (root, width =1000, height = 200, background = 'white')
                          # Same as before, except the background colour is set to white
myCanvas.pack()           # Same as before

xPositionTop = 50         # Variable that stores the current x position of the rectangle's top left corner
xPositionBottom  =  xPositionTop + 75       '''Variable that stores the x position of the rectangle's bottom
                                            right corner'''

for i in range (1, 50):       # The loop which will draw the moving rectangle moving to the right 50 times

    myCanvas.create_rectangle (xPositionTop ,50 ,xPositionBottom ,150 ,fill ='white', outline = 'white')
                              # Draws a white rectangle over the red one (erases it)

    xPositionTop += 10        # x position of top left corner increases by 10
    xPositionBottom +=10      # x position of bottom right corner increases by 10

    myCanvas.create_rectangle (xPositionTop, 50, xPositionBottom , 150, fill = 'red')
                              # Draws the red rectangle at the new position
```

```
    myCanvas.update( )      # Refreshes the canvas window each time the rectangle makes one movement.
                            # This line is needed in any program with moving graphics because it redraws
                            # the window.

    time.sleep (0.5)        # Calls the "sleep" function from the library "time". This function halts the program
                            # for the given amount of time (in this case, half a second). In programming, this is
                            # called a delay

root.mainloop ()            # Same as before
```

Assignment #2

Create a car race game which will have two shapes start at one end of the window and "race" to a finish line at the other end of the window. A winner will be declared and the program will end when one of the shapes reaches the finish line.

Hints: 1) Each shape needs its own coordinate variable(s).
2) To have a real race, the amount each shape moves every time through the loop should be random.

Extentions: 1) Add more shapes using a list.
2) Have the shapes "turn around" when they touch the original finish line and finish the race at the original finish line.
3) Make the winner "celebrate" - be original.