1/22/2008

>>> Overview

* parameters
* graphics

>>> Parameters

Parameters are easy in Python once you know Java's. Simply remove all types from the method header and do the normal conversion.

PrintSum.java

```
1  public static void printSum(int x, int y) {
2          System.out.println(x + y);
3  }
4
```

print_sum.py

```
1  def print_sum(x, y):
2          print str(x + y)
3
4  sum(2, 3)
```

>>> Parameters Example

```
**********

*******

**************************************************

**********
*        *
**********

*****
*   *
*   *
*****
```

>>> Example Solution

```
def draw_line(num):
    print "*" * num


def draw_box(width, height):
    draw_line(width)
    for i in range(height-2):
        print "*" + " " * (width-2) + "*"
    draw_line(width)

#main
draw_line(13)
draw_line(7)
draw_line(35)
draw_box(10,3)
draw_box(5,4)
```

>>> Defaults

Unlike Java, Python's parameters can have default values to use when one is not given.

print_range.py

```
1  def print_range(start=1, end=1, interval=1, sep=" "):
2          for i in range(start, end, interval):
3                  print str(i) + sep,
4          print end
5
6  print range(0,7)
7  print_range(1,7,1,", ")
```

>>> Keywords

When calling a function with a number of parameters with defaults you can modify particular parameters with a keyword so that you do not need to specify all preceding parameters.

print_range.py

```
1  def print_range(start=1,end=1,interval=1,sep=" "):
2          for i in range(start,end,interval):
3                  print str(i) + sep,
4          print end
5
6  print range(0,7)
7  print_range(1,7,1,", ")
8
9  print_range(end=7,sep=", ")
```

>>> Graphics

Graphics in Python are similar to graphics in Java

- drawingpanel.py needs to be in the same directory as your program that uses it
- The Graphics (g) in Python behaves like the Graphics (g) in Java and is passed as a parameter in the same fashion.
- To let the Python interpreter know that you want to use the drawingpanel.py file you must add “from drawingpanel import *” at the *top* of your file
- panel.mainloop() must be put at the end of the program

>>> Graphics Methods

Java	Python
g.drawLine(x1, y1, x2, y2);	g.create_line(x1, y1, x2, y2) g.create_line(x1, y1, x2, y2, x3, y3,…, xN, yN)
g.drawOval(x1, y1, width, height);	g.create_oval(x1, y1, x2, y2)
g.drawRect(x1, y1, width, height);	g.create_rectangle(x1, y1, x2, y2)
panel.setBackground(Color);	g[“bg”] = “ <color> “

>>> Graphics Methods

Java

```
1  DrawingPanel panel = new DrawingPanel(300, 200);
2  Graphics g = panel.getGraphics();
3  panel.setBackground(Color.YELLOW);
4
```

Python

```
1  panel = DrawingPanel(300, 200)
2  g = panel.get_graphics()
3  g["bg"] = "yellow"
4
```

>>> Graphics

- What about...?
  - g.setColor()
  - g.fillRect(), g.fillOval()

- Fill colors and borders in Python are set as parameters in the methods.

Java

```
1 g.setColor(Color.RED);
2 g.fillRect(x, y, w, h,);
3 g.setColor(Color.BLACK);
4 g.drawRect(x, y, w, h);
```

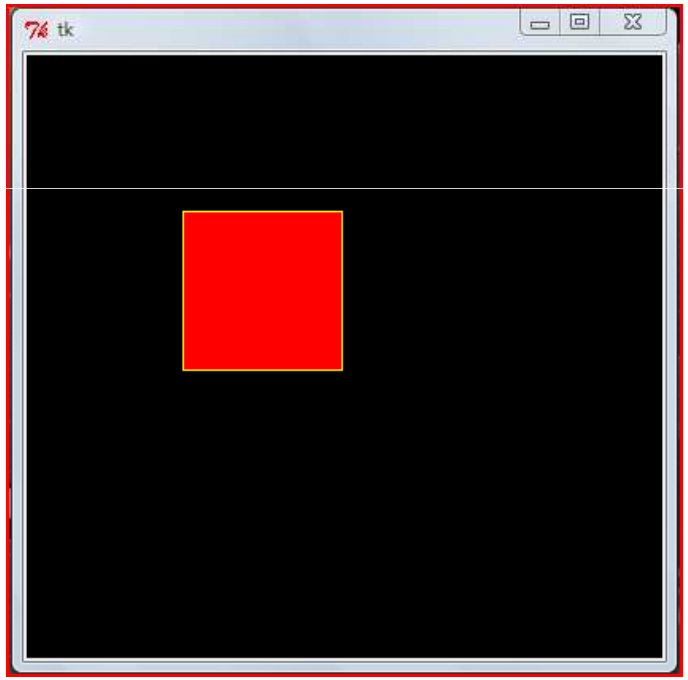
Python

```
1
2 g.create_rectangle(x, y, x+w, y+h, fill="red", outline="black")
3
4
```

>>> Graphics Example 1

Python

```
1  from drawingpanel import *
2  panel = DrawingPanel(400,380)
3  g = panel.get_graphics()
4  g["bg"]="black“
5  g.create_rectangle(100, 100, 200, 200, fill="red", outline="yellow")
6  panel.mainloop()
```



python™

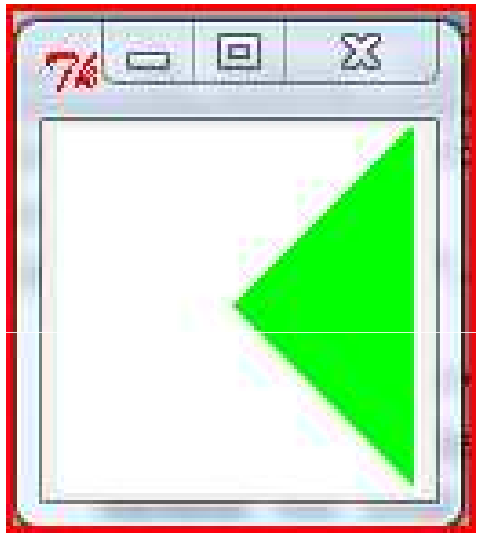
>>> Graphics

- What about...?
  - Triangles
  - Hexagons
  - Etc.

- g.create_polygon(x1, y1, x2, y2,..., xN, yN)

Python

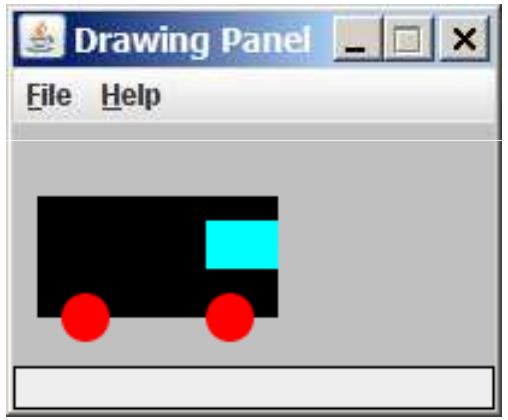
```
1  from drawingpanel import *
2  panel = DrawingPanel(100,100)
3  g = panel.get_graphics()
4  g.create_polygon(50, 50, 100, 0, 100, 100, fill="green")
5  panel.mainloop()
```



>>> Graphics Example 2

Let's recreate the Java car example in Python:

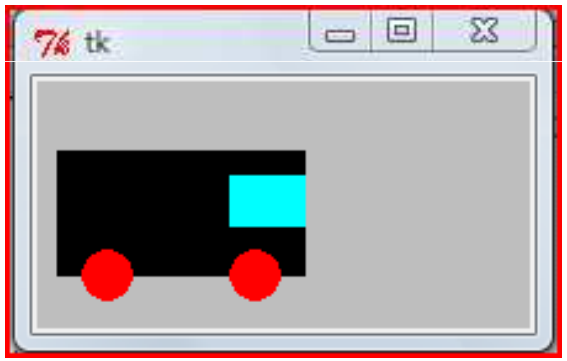
```
import java.awt.*;
public class DrawCar {
    public static void main(String[] args) {
        DrawingPanel panel = new DrawingPanel(200, 100);
        panel.setBackground(Color.LIGHT_GRAY);
        Graphics g = panel.getGraphics();
        g.setColor(Color.BLACK);
        g.fillRect(10, 30, 100, 50);
        g.setColor(Color.RED);
        g.fillOval(20, 70, 20, 20);
        g.fillOval(80, 70, 20, 20);
        g.setColor(Color.CYAN);
        g.fillRect(80, 40, 30, 20);
    }
}
```



>>> Example 2 (auf Python)

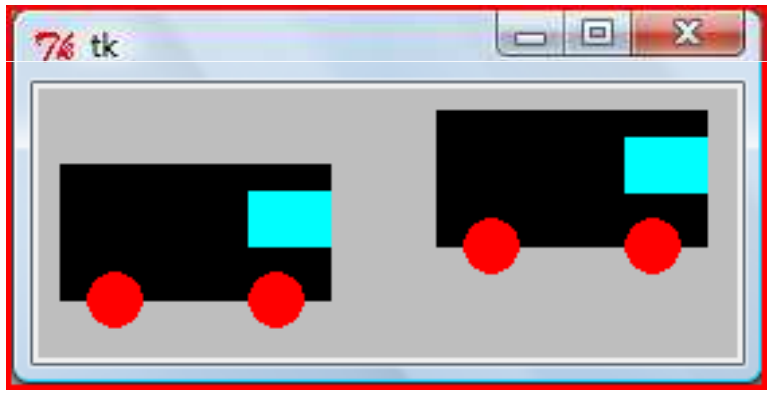
Let's recreate the Java car example in Python:

```
from drawingpanel import *

panel = DrawingPanel(200,100)
g = panel.get_graphics()
g["bg"] = "gray"

g.create_rectangle(10, 30, 10+100, 30+50, fill="black")
g.create_oval(20, 70, 20+20, 70+20,fill="red", outline="red")
g.create_oval(80, 70, 80+20, 70+20, fill="red", outline="red")
g.create_rectangle(80, 40, 80+30, 40+20, fill="cyan", outline="cyan")
```



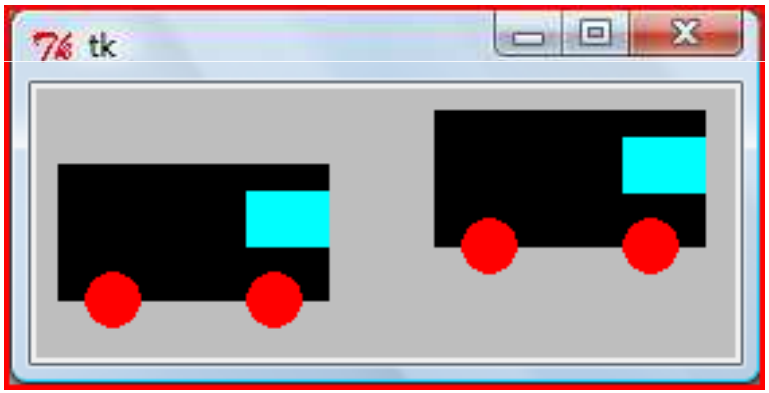
>>> Example 2 - Parameterized

Now, let's use parameters so that we can place the cars all over the DrawingPanel.



>>> Example 2 - Parameterized

```
from drawingpanel import *

def draw_car(g, x, y):
    g.create_rectangle(x, y, x+100, y+50, fill="black")
    g.create_oval(x+10, y+40, x+10+20, y+40+20, fill="red", outline="red")
    g.create_oval(x+70, y+40, x+70+20, y+40+20, fill="red", outline="red")
    g.create_rectangle(x+70, y+10, x+70+30, y+10+20, fill="cyan“, outline=“cyan”)

# main
panel = DrawingPanel(260,100)
g = panel.get_graphics()
g["bg"] = "gray"
draw_car(g, 10, 30)
draw_car(g, 150, 10)
```



python™