python™

Week 3

parameters, return, math, graphics

Special thanks to Scott Shawcroft, Ryan Tucker, and Paul Beck for their work on these slides.

Parameters

`def` **`name(parameter, parameter, ..., parameter):`**
`statements`

– Parameters are declared by writing their names (no types)

```
>>> def print_many(word, n):
...     for i in range(n):
...         print(word)

>>> print_many("hello", 4)
hello
hello
hello
hello
```

Exercise

- Recreate the lines/boxes of stars example from lecture:

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

*******

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

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

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

Exercise Solution

stars.py

```
 1  # Draws a box of stars with the given width and height.
 2  def box(width, height):
 3      print width * "*"
 4      for i in range(height - 2):
 5          print("*" + (width - 2) * " " + "*")
 6      print width * "*"
 7
 8  # main
 9  print(13 * "*")
10  print(7 * "*")
11  print(35 * "*")
12  box(10, 3)
13  box(5, 4)
```

Default Parameter Values

`def` **name**`(`**parameter**`=`**value**`, ..., `**parameter**`=`**value**`):`
    statements

- Can make parameter(s) optional by specifying a default value

```
>>> def print_many(word, n=1):
...     for i in range(n):
...         print(word)

>>> print_many("shrubbery")
shrubbery
>>> print_many("shrubbery", 4)
shrubbery
shrubbery
shrubbery
shrubbery
```

- **Exercise:** Modify `stars.py` to add an optional parameter for the character to use for the outline of the box (default `"*"`).

Parameter Keywords

name(parameter=value, ..., parameter=value)

- Can specify the names of parameters as you call a function
- This allows you to pass the parameters in any order

```
>>> def print_many(word, n):
...     for i in range(n):
...         print(word)

>>> print_many(word="shrubbery", n=4)
shrubbery
shrubbery
shrubbery
shrubbery
>>> print_many(n=3, word="Ni!")
Ni!
Ni!
Ni!
```

python

Math commands

```
from math import *
```

Function name	Description
`ceil(`**value**`)`	rounds up
`cos(`**value**`)`	cosine, in radians
`degrees(`**value**`)`	convert radians to degrees
`floor(`**value**`)`	rounds down
`log(`**value**`, `**base**`)`	logarithm in any base
`log10(`**value**`)`	logarithm, base 10
`max(`**value1**`, `**value2, ...**`)`	largest of two (or more) values
`min(`**value1**`, `**value2, ...**`)`	smallest of two (or more) values
`radians(`**value**`)`	convert degrees to radians
`round(`**value**`)`	nearest whole number
`sin(`**value**`)`	sine, in radians
`sqrt(`**value**`)`	square root
`tan(`**value**`)`	tangent

Constant	Description
`e`	2.7182818...
`pi`	3.1415926...

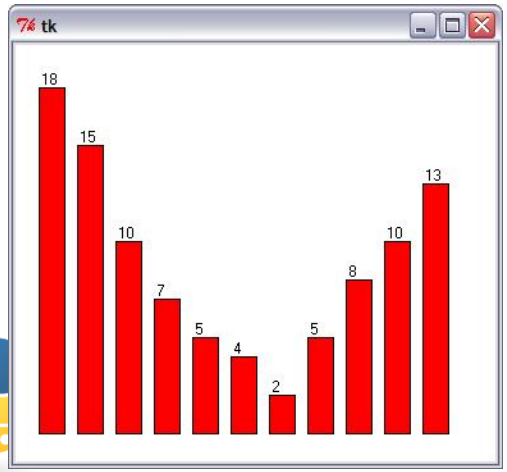
Returning Values

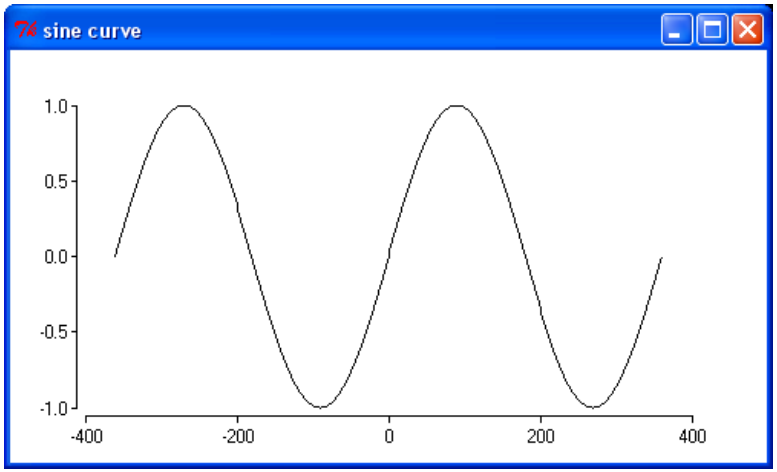
```
def name(parameters):
    statements
    ...
    return value
```

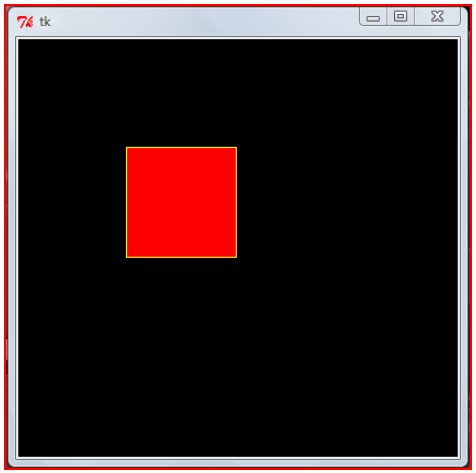
```
>>> def ftoc(temp):
...     tempc = 5.0 / 9.0 * (temp - 32)
...     return tempc

>>> ftoc(98.6)
37.0
```

DrawingPanel

- Instructor-provided `drawingpanel.py` file must be in the same folder as your Python program
- At the top of your program, write:
  - `from drawingpanel import *`
- Panel's `canvas` field behaves like `Graphics g` in Java



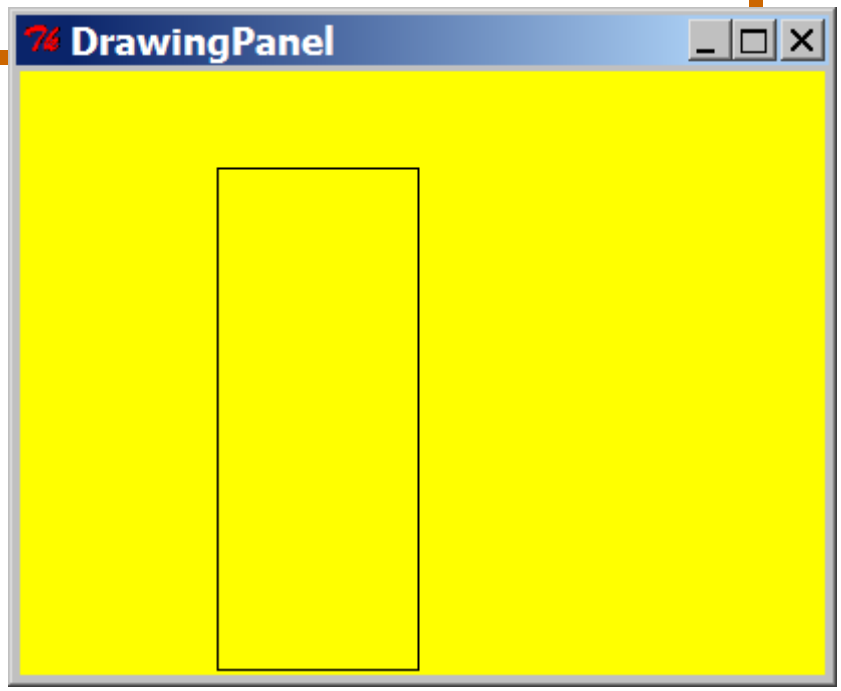




DrawingPanel Example

draw1.py

```
1  from drawingpanel import *
2
3  panel = DrawingPanel(400, 300)
4  panel.set_background("yellow")
5  panel.canvas.create_rectangle(100, 50, 200, 300)
6
```



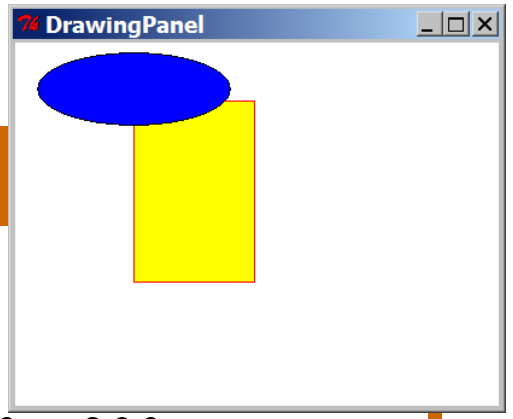
Drawing Methods

Java	Python
`drawLine`	**panel**`.canvas.create_line(`**x1**`, `**y1**`, `**x2**`, `**y2**`)`
`drawRect, fillRect`	**panel**`.canvas.create_rectangle(`**x1**`, `**y1**`, `**x2**`, `**y2**`)`
`drawOval, fillOval`	**panel**`.canvas.create_oval(`**x1**`, `**y1**`, `**x2**`, `**y2**`)`
`drawString`	**panel**`.canvas.create_text(`**x**`, `**y**`, text="`**text**`")`
`setColor`	*(see next slide)*
`setBackground`	**panel**`.set_background(`**color**`)`

- Notice, methods take x2/y2 parameters, not width/height

Colors and Fill

- Python doesn't have `fillRect`, `fillOval`, or `setColor`.
  - Instead, pass outline and fill colors when drawing a shape.
  - List of all color names: http://wiki.tcl.tk/16166
  - Visual display of all colors



drawcolors.py

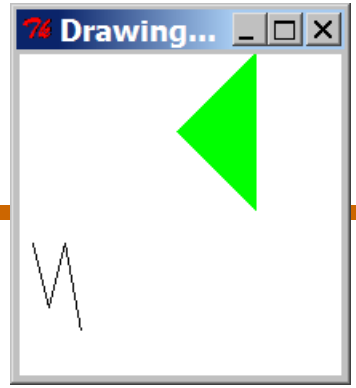
```
1  from drawingpanel import *
2
3  panel = DrawingPanel(400, 300)
4  panel.canvas.create_rectangle(100, 50, 200, 200,
           outline="red", fill="yellow")
5  panel.canvas.create_oval(20, 10, 180, 70, fill="blue")
6
```

Polygons

- Draw arbitrary polygons with `create_polygon`
- Draw line groups by passing more params to `create_line`

drawpoly.py

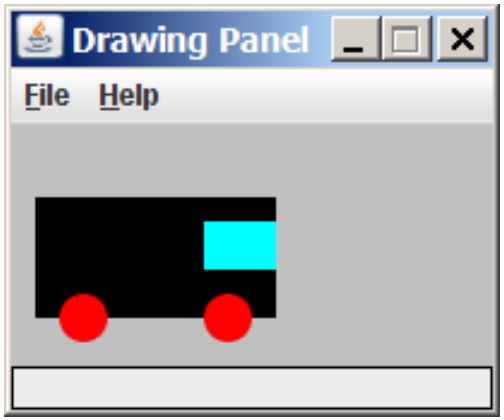
```
1  from drawingpanel import *
2
3  panel = DrawingPanel(200, 200)
4  panel.canvas.create_polygon(100, 50, 150, 0,
                               150, 100, fill="green")
5  panel.canvas.create_line(10, 120, 20, 160,
                            30, 120, 40, 175)
6
```



Exercise

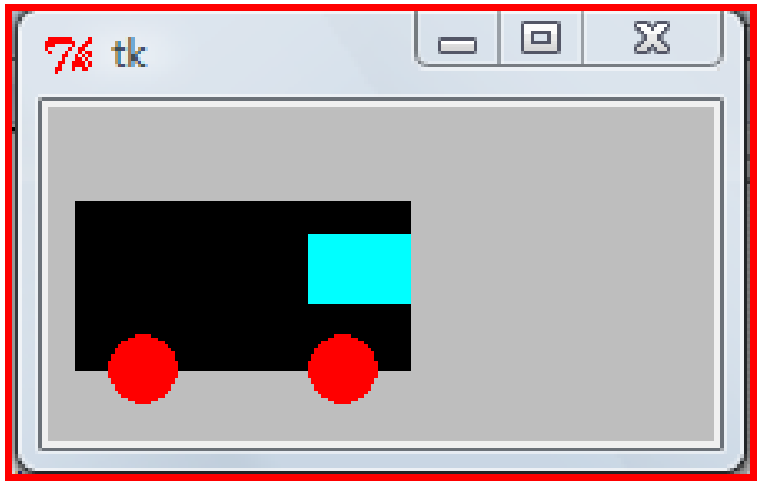
Let's create a car in Python:

```
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);
    }
}
```



Exercise

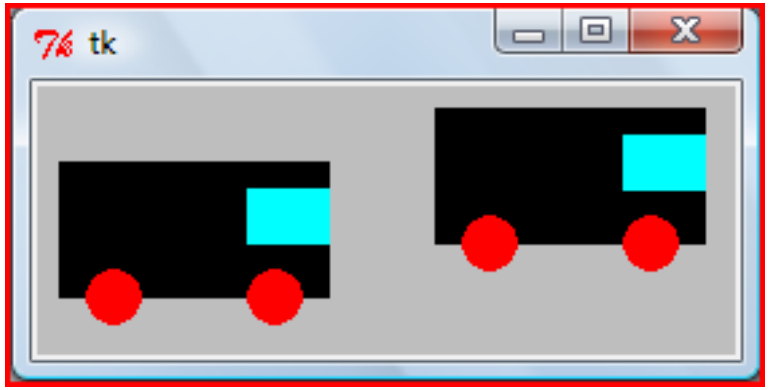
Draw a car in Python



python™

Exercise

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



Exercise

Animate it using panel.sleep()