Unit 3

parameters and graphics

Special thanks to Roy McElmurry, John Kurkowski, Scott Shawcroft, Ryan Tucker, Paul Beck for their work.

Constants

- Python doesn't really have constants.
  - Instead, declare a variable at the top of your code.
  - All methods will be able to use this "constant" value.

constant.py

```python
MAX_VALUE = 3

def print_top():
    for i in range(MAX_VALUE):
        for j in range(i):
            print(j)
        print()

def print_bottom():
    for i in range(MAX_VALUE, 0, -1):
        for j in range(i, 0, -1):
            print(MAX_VALUE)
        print()
```

Exercise

- Rewrite the Mirror lecture program in Python. Its output:

```
#================#
|      <><>      |
|    <>....<>    |
|  <>........<>  |
|<>............<>|
|<>............<>|
|  <>........<>  |
|    <>....<>    |
|      <><>      |
#================#
```

  – Make the mirror resizable by using a "constant."

Exercise Solution

```
SIZE = 4

def bar():
    print("#" + 4 * SIZE * "=" + "#")

def top():
    for line in range(1, SIZE + 1):
        # split a long line by ending it with \
        print("|" + (-2 * line + 2 * SIZE) * " " + \
              "<>" + (4 * line - 4) * "." + "<>" + \
              (-2 * line + 2 * SIZE) * " " + "|")

def bottom():
    for line in range(SIZE, 0, -1):
        print("|" + (-2 * line + 2 * SIZE) * " " + \
              "<>" + (4 * line - 4) * "." + "<>" + \
              (-2 * line + 2 * SIZE) * " " + "|")

# main
bar()
top()
bottom()
bar()
```

Parameters

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

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

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

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

Exercise

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

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

*******

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

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

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

Exercise Solution

stars.py

```python
# Draws a box of stars with the given width and height.
def box(width, height):
    print(width * "*")
    for i in range(height - 2):
        print("*" + (width - 2) * " " + "*")
    print(width * "*")

# main
print(13 * "*")
print( 7 * "*")
print(35 * "*")
box(10, 3)
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(message, n=1):
...     for i in range(n):
...         print(message)

>>> print_many("shrubbery")
shrubbery
>>> print_many("shrubbery", 3)
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

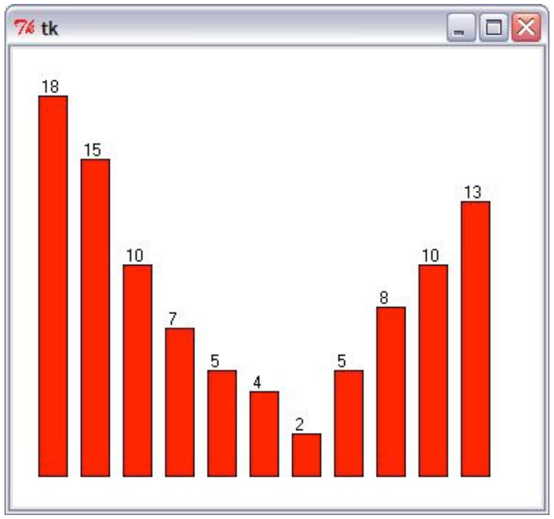
name(**parameter**=**value**, ..., **parameter**=**value**)

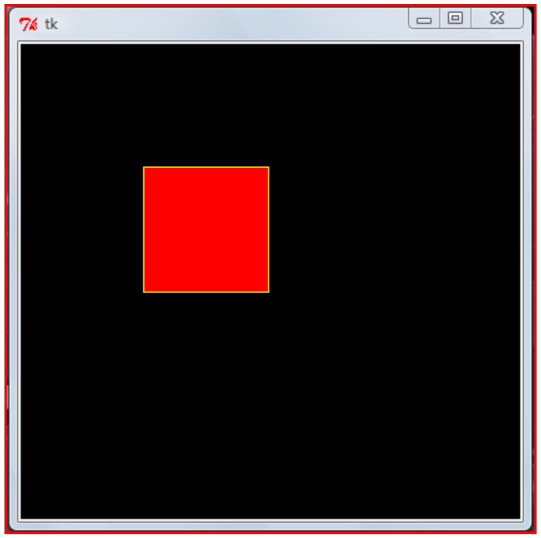
- Can specify name of each parameter as you call a function
- This allows you to pass the parameters in any order

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

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

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

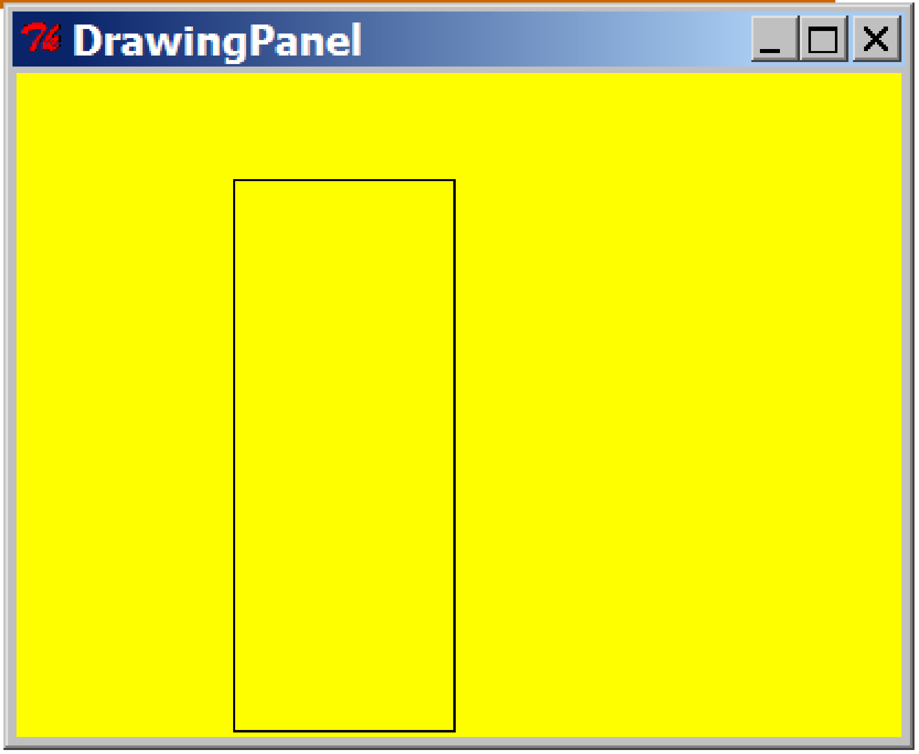




DrawingPanel Example

draw1.py

```
from drawingpanel import *

panel = DrawingPanel(400, 300)
panel.set_background("yellow")
panel.canvas.create_rectangle(100, 50, 200, 300)
```



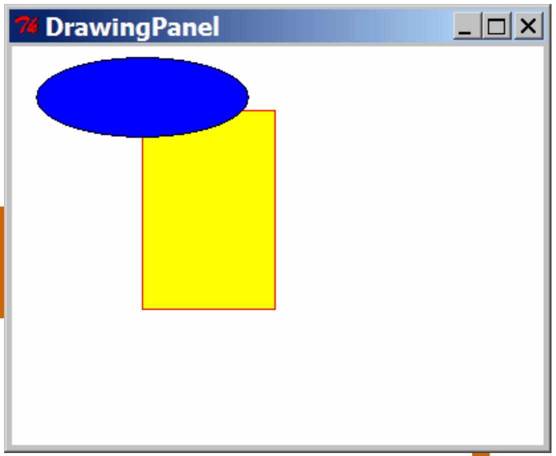
Drawing Methods

Java	Python
`drawLine`	**panel**`.canvas.create_line(`**x1**`, `**y1**`, `**x2**`, `**y2**`)`
`drawRect`, `fillRect`	**panel**`.canvas.create_rect(`**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
  - See class web site for visual index of colors!



drawcolors.py

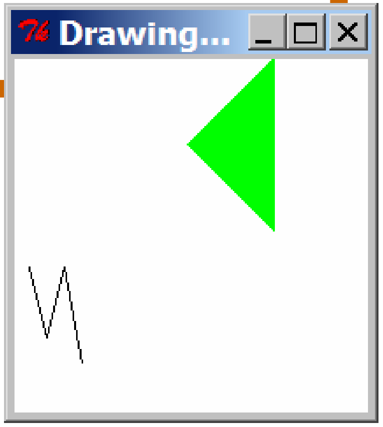
```
from drawingpanel import *

panel = DrawingPanel(400, 300)
panel.canvas.create_rectangle(100, 50, 200, 200,
        outline="red", fill="yellow")
panel.canvas.create_oval(20, 10, 180, 70, fill="blue")
```

Polygons

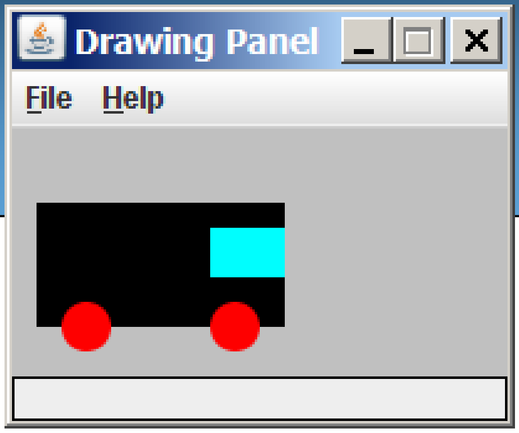
- Draw arbitrary polygons with `create_polygon`
- Draw line groups by passing more params to `create_line`

drawpoly.py

```
from drawingpanel import *

panel = DrawingPanel(200, 200)
panel.canvas.create_polygon(100, 50, 150, 0,
                            150, 100, fill="green")
panel.canvas.create_line(10, 120, 20, 160,
                         30, 120, 40, 175)
```



Exercise

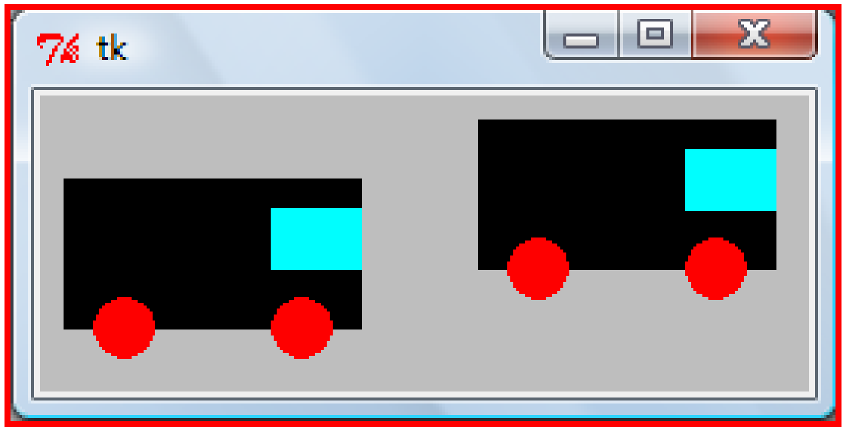


- Write a Python version of the Car program.
  - Convert this Java code to Python:

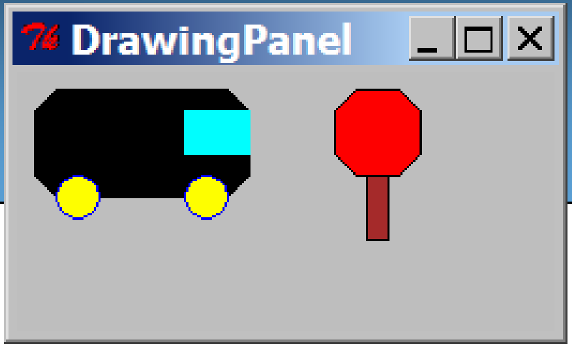
```
DrawingPanel panel = new DrawingPanel(200, 200);
panel.setBackground(Color.LIGHT_GRAY);
Graphics g = panel.getGraphics();

g.setColor(Color.BLACK);          // body
g.fillRect(10, 30, 100, 50);

g.setColor(Color.RED);            // wheels
g.fillOval(20, 70, 20, 20);
g.fillOval(80, 70, 20, 20);

g.setColor(Color.CYAN);           // windshield
g.fillRect(80, 40, 30, 20);
```

Exercise

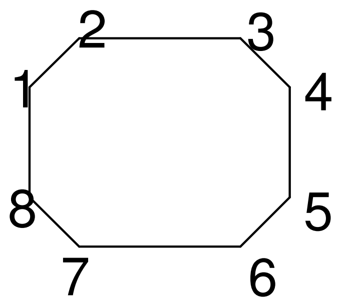
- Modify your car program to use parameters so that cars can be drawn in many different locations.



Exercise



- Write a variation of the Car program where the car body is octagonal and there is a stop sign.



  - Stop sign at (150, 10), size 40
    - post at (165, 50), size 10x30, brown fill
  - Write an **octagon** function to draw the car body / stop sign.
    - Points of car body, located at (10, 10):

      1. (10, 20), 2. (20, 10), 3. (100, 10), 4. (110, 20),
      5. (110, 50), 6. (100, 60), 7. (20, 60), 8. (10, 50)
    - Points of stop sign, located at (150, 10):

      1. (150, 20), 2. (160, 10), 3. (180, 10), 4. (190, 20),
      5. (190, 40), 6. (180, 50), 7. (160, 50), 8. (150, 40)

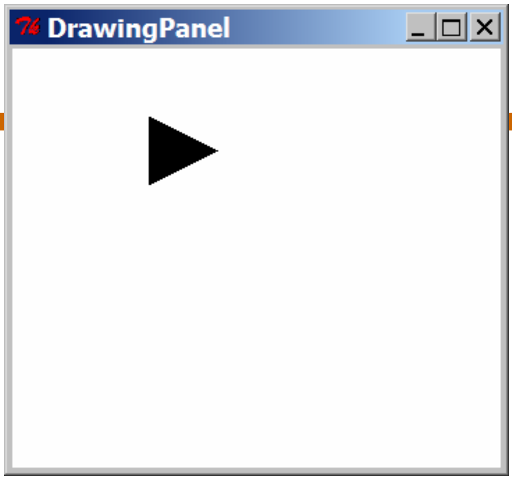
(An octagon has 10x10 triangular cuts in each corner.)

Animation

- Pause the panel by calling `sleep`

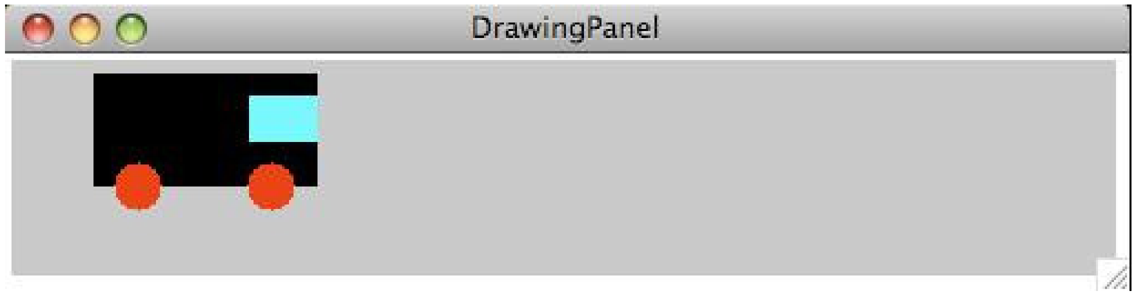
animation.py

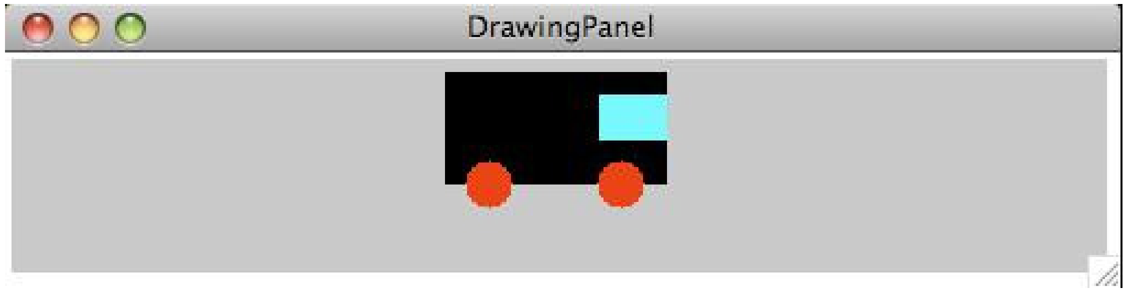
```
from drawingpanel import *

panel = DrawingPanel(350, 300)
for i in range(20):
    # clear any previous image
    panel.canvas.create_rectangle(0, 0, 400, 400,
            outline="white", fill="white")

    panel.canvas.create_polygon(20 * i, 50, 20 * i,
            100, 20 * i + 50, 75)
    panel.sleep(100)
```

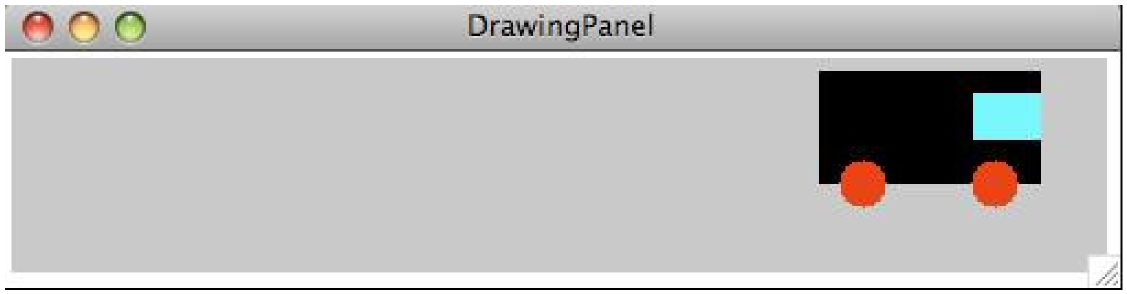


Exercise

- Animate the car to make it drive across the panel using the `sleep` function.


DrawingPanel


DrawingPanel


DrawingPanel