交互式图形用户接口

黄天羽

北京理工大学

python 语言程序设计

- 图形用户界面（图形用户接口），
  - 采用图形方式显示的计算机操作用户界面
  - 用于程序的输入和输出
  - 事件驱动

python 语言程序设计

- 事件
  - 移动鼠标
  - 点击按钮
    - 按钮事件
    - 按钮处理代码
  - 键盘输入

事件驱动程序需要编程人员知道任何指定的时刻 “谁在负责”。

python 语言程序设计

- Graphics模块
  - 隐藏了底层事件的处理机制，
  - 提供了获得用户在窗口中的输入
    - 捕捉鼠标点击
    - 处理文本输入

连续点击10次鼠标，返回其坐标值.

```
from graphics import *
def main():
    win = GraphWin("Click Me!")
    for i in range(10):
        p = win.getMouse()
        print("You clicked at:", p.getX(), p.getY())

if __name__ == '__main__':
    main()
```

python 语言程序设计

运行程序可以看到Python Shell输出了这十个点的坐标

```
>>>
You clicked at: 29 21
You clicked at: 36 65
You clicked at: 66 111
You clicked at: 116 78
You clicked at: 115 136
You clicked at: 143 81
You clicked at: 150 136
You clicked at: 91 76
You clicked at: 122 38
You clicked at: 57 157
```

python 语言程序设计

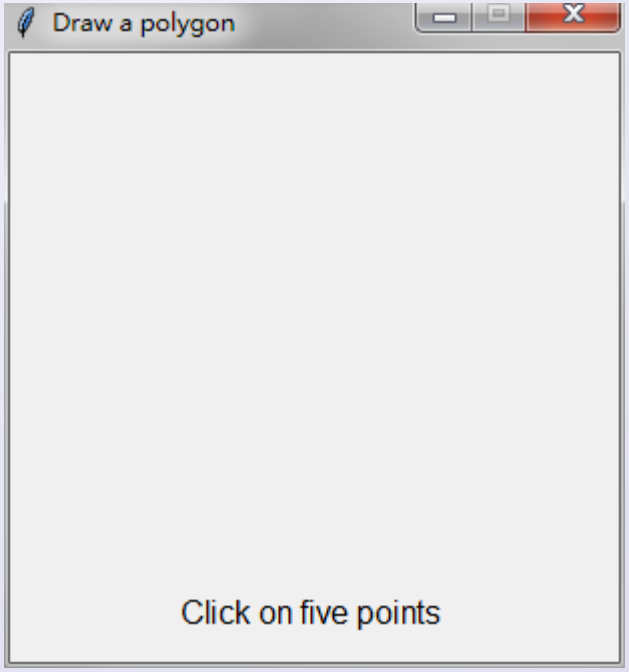
在窗口中点击5个点来画一个五边形。

```
from graphics import *

win = GraphWin("Draw a polygon",300,300)
win.setCoords(0.0, 0.0, 300.0, 300.0)
message = Text(Point(150, 20), "Click on five points")
message.draw(win)
#获得多边形的5个点
p1 = win.getMouse()
p1.draw(win)
p2 = win.getMouse()
p2.draw(win)
p3 = win.getMouse()
p3.draw(win)
p4 = win.getMouse()
p4.draw(win)
p5 = win.getMouse()
p5.draw(win)
# 使用Polygon对象绘制多边形
polygon = Polygon(p1,p2,p3,p4,p5)
polygon.setFill("peachpuff")
polygon.setOutline("black")
polygon.draw(win)
# 等待响应鼠标事件，退出程序
message.setText("Click anywhere to quit.")
win.getMouse()
```

python 语言程序设计

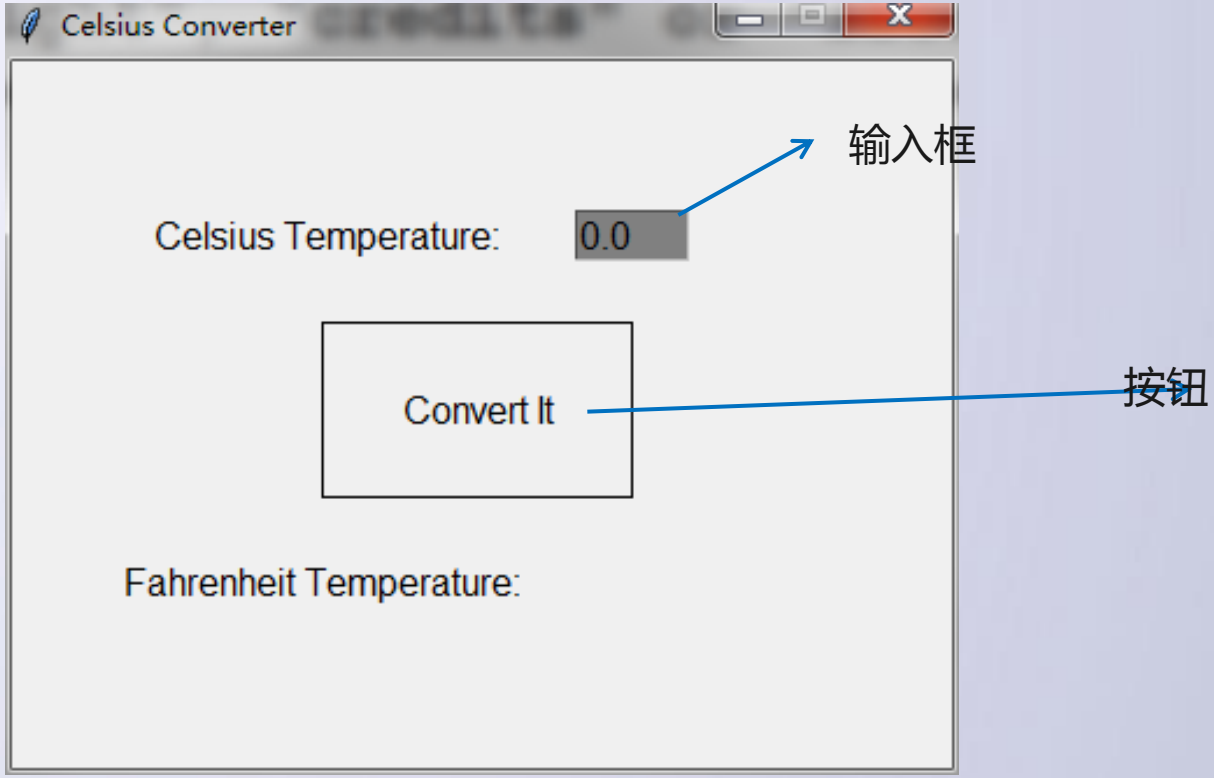
程序结果如下:



- Text对象: setText()和getText()
- Entry对象: setText()和getText()
  - 内容可以被用户修改

```
from graphics import *
win = GraphWin("Celsius Converter", 400, 300)
win.setCoords(0.0, 0.0, 3.0, 4.0)
# 绘制接口
Text(Point(1,3), " Celsius Temperature:").draw(win)
Text(Point(1,1), "Fahrenheit Temperature:").draw(win)
input = Entry(Point(2,3), 5)
input.setText("0.0")
input.draw(win)
output = Text(Point(2,1),"")
output.draw(win)
button = Text(Point(1.5,2.0),"Convert It")
button.draw(win)
Rectangle(Point(1,1.5), Point(2,2.5)).draw(win)
# 等待鼠标点击
win.getMouse()
# 转换输入
celsius = eval(input.getText())
fahrenheit = 9.0/5.0 * celsius + 32.0
# 显示输出，改变按钮
output.setText(fahrenheit)
button.setText("Quit")
# 等待响应鼠标点击，退出程序
win.getMouse()
win.close()
```

python 语言程序设计



python 语言程序设计

创建GUI程序的基本步骤为:

- 导入Tk模块.
- 创建GUI应用程序的主窗口.
- 添加控件或GUI应用程序.
- 进入主事件循环，等待响应用户触发事件.

15种常见的 Tk 控件

Button, Canvas, Checkbutton, Entry, Frame, Label,

Listbox, Menubutton, Menu, Message , Radiobutton,

Scale Scrollbar, Text, Toplevel, Spinbox

PanedWindow, LabelFrame, tkMessageBox

- 共同属性
  - Dimensions ：尺寸
  - Colors：颜色
  - Fonts：字体
  - Anchors：锚
  - Relief styles：浮雕式
  - Bitmaps：显示位图
  - Cursors：光标的外形
- 特有属性

界面布局

- Tkinter三种几何管理方法
  - pack()
  - grid()
  - place()

python 语言程序设计

创建GUI应用程序窗口代码模板

```
from tkinter import *
tk = Tk()
#此处添加控件代码
...
tk.mainloop()
```

python 语言程序设计

简单GUI示例

```
from tkinter import *

tk = Tk()
label = Label(tk, text = "Welcome to Python Tkinter")
button = Button(tk, text= "Click Me")
label.pack()
button.pack()
tk.mainloop()
```

python 语言程序设计

响应用户事件示例

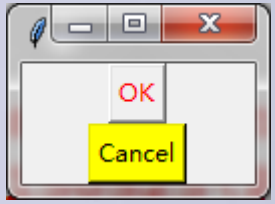
```
from tkinter import *

def processOK():
    print("OK button is clicked")

def processCancel():
    print("Cancel button is clicked")

def main():
    tk = Tk()
    btnOK = Button(tk, text = "OK", fg = "red",
                   command = processOK)
    btnCancel = Button(tk, text = "Cancel", bg = "yellow",
                       command = processCancel)
    btnOK.pack()
    btnCancel.pack()

    tk.mainloop()
```



python 语言程序设计

显示文字、图片、绘制图形

```
from tkinter import *

tk = Tk()
canvas = Canvas(tk, width = 200, height = 200)
canvas.pack()
canvas.create_text(100, 40, text = "Welcome to Tkinter",
                   fill = "blue", font = ("Times", 16))
myImage = PhotoImage(file = "python_logo.gif")
canvas.create_image(10, 70, anchor = NW, image = myImage)
canvas.create_rectangle(10,70, 190, 130)

tk.mainloop()
```

python 语言程序设计

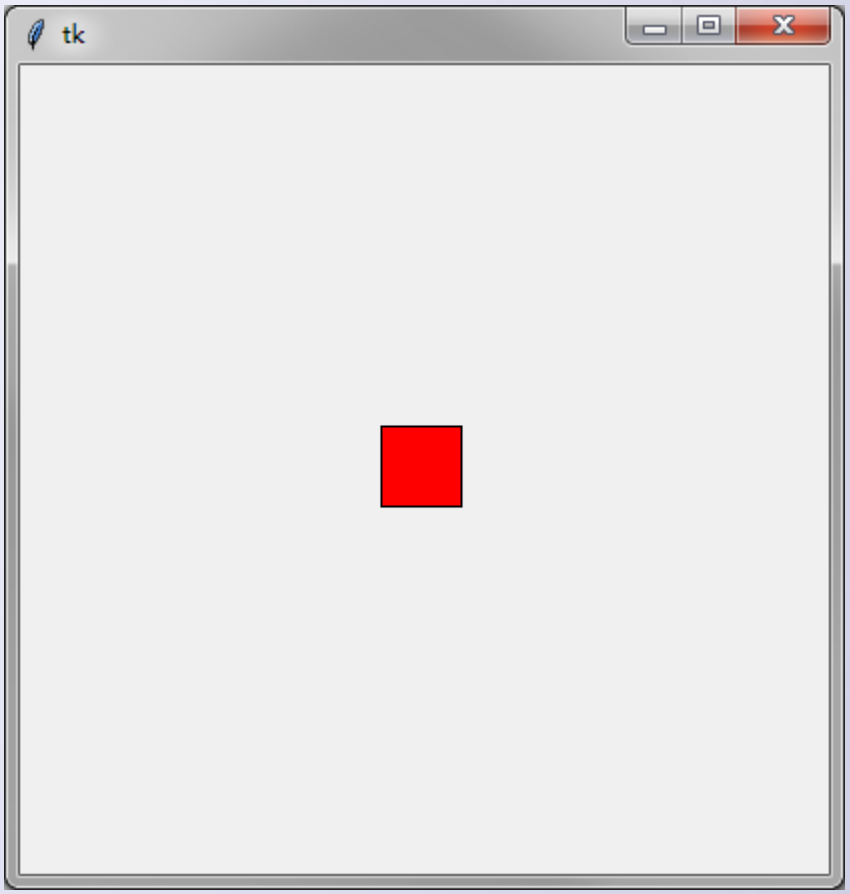
控制图形移动的示例

```
from tkinter import *

tk = Tk()
canvas = Canvas(tk, width = 400, height = 400)
canvas.pack()

def moverectangle(event):
    if event.keysym == "Up":
        canvas.move(1,0,-5)
    elif event.keysym == "Down":
        canvas.move(1,0,5)
    elif event.keysym == "Left":
        canvas.move(1,-5,0)
    elif event.keysym == "Right":
        canvas.move(1,5,0)

canvas.create_rectangle(10,10,50,50,fill="red")
canvas.bind_all("<KeyPress-Up>",moverectangle)
canvas.bind_all("<KeyPress-Down>",moverectangle)
canvas.bind_all("<KeyPress-Left>",moverectangle)
canvas.bind_all("<KeyPress-Right>",moverectangle)
```



python
语言程序设计