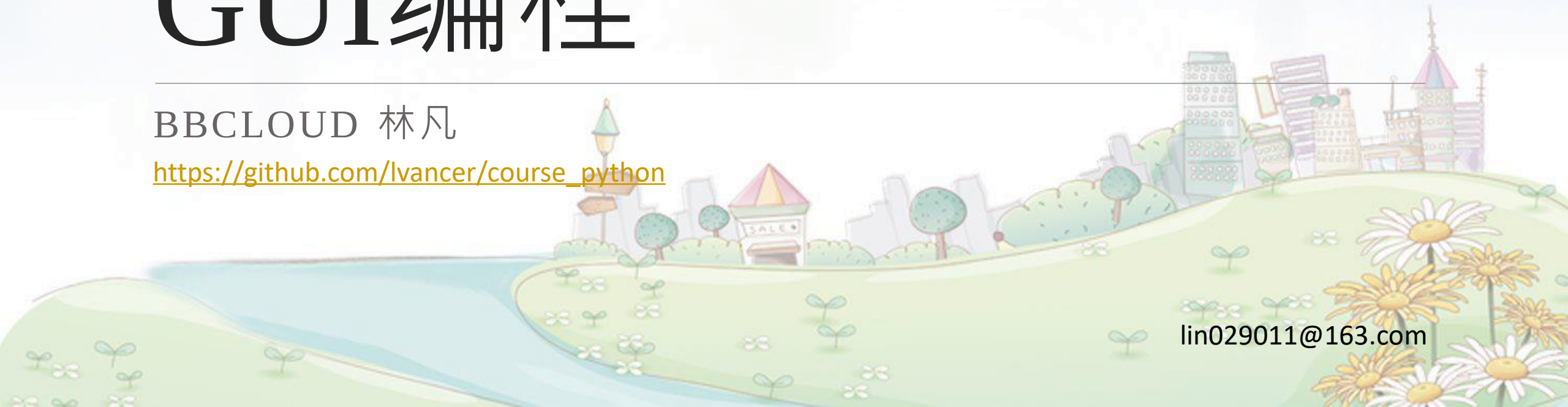
LV5

GUI编程

BBCLOUD 林凡

https://github.com/lvancer/course_python

lin029011@163.com



LV5

大纲

GUI

基础控件

容器布局

选择控件

列表控件

对话框

菜单控件

画布

事件绑定

练习

lin029011@163.com

LV5

GUI

GUI全称是图像用户界面，相对于命令行，GUI使用起来就很方便了。

主流的GUI有：

1、Tkinter：Python自带，轻量级，一些简单的界面可以快速开发。

2、wxPython： 中级，专业的小工具可以使用。

3、PyQt5：比较专业的开发工具，借助C++语言开发的Qt5框架，加了一层Python壳。

有大量的工具简化开发，如界面可以直接通过拖动完成。

下面主要介绍Tkinter，了解基本的GUI编写方法。

lin029011@163.com

LV5

基础控件

第一个GUI界面

```
import tkinter as tk          # Tk库

root = tk.Tk()                # 创建Tk对象
root.title("My First GUI")    # 界面标题
root.geometry('400x300')      # 界面大小
root.mainloop()               # 启动界面
```

这就是一个空白的界面，下面我们开始添加控件。

所有的界面代码都在mainloop方法前。

lin029011@163.com

LV5

基础控件

标签：Label。

```
label = tk.Label(root, text="输入",
                 bg="white", font=("Arial", 12),
                 width=5, height=1)
label.pack()     # 添加到界面
```

第一个参数是根节点root，所有控件通用。

text：现实的文本，Label核心参数。

bg：背景颜色【可不填，控件通用】。

font：字体【可不填，控件通用】。

width，height：宽度和高度，不是像素为单位【可不填，控件

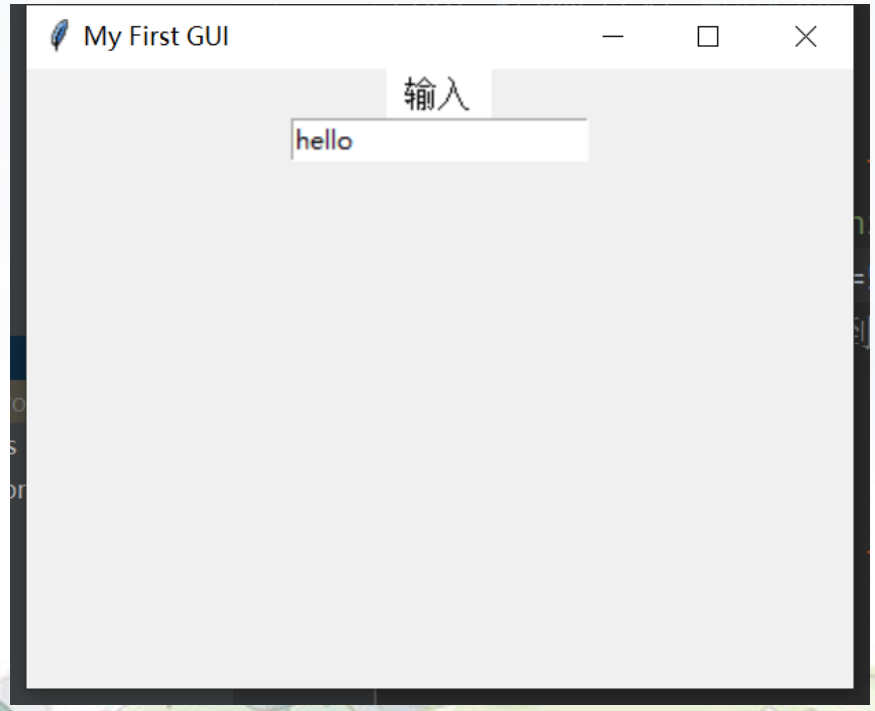
lin029011@163.com

LV5

基础控件

单行文本输入框：Entry。

```
var = tk.StringVar()     # 用于存储内容的变量
entry = tk.Entry(root, textvariable=var)
var.set("hello")         # 设置输入框内容
print(var.get())         # 获取输入框内容
entry.pack()
```



textvariable：要绑定的存储变量。

StringVar是tk的字符串变量，用于给控件进行变量存储。

绑定到控件后，使用set，get方法进行设置与获取。

lin029011@163.com

LV5

基础控件

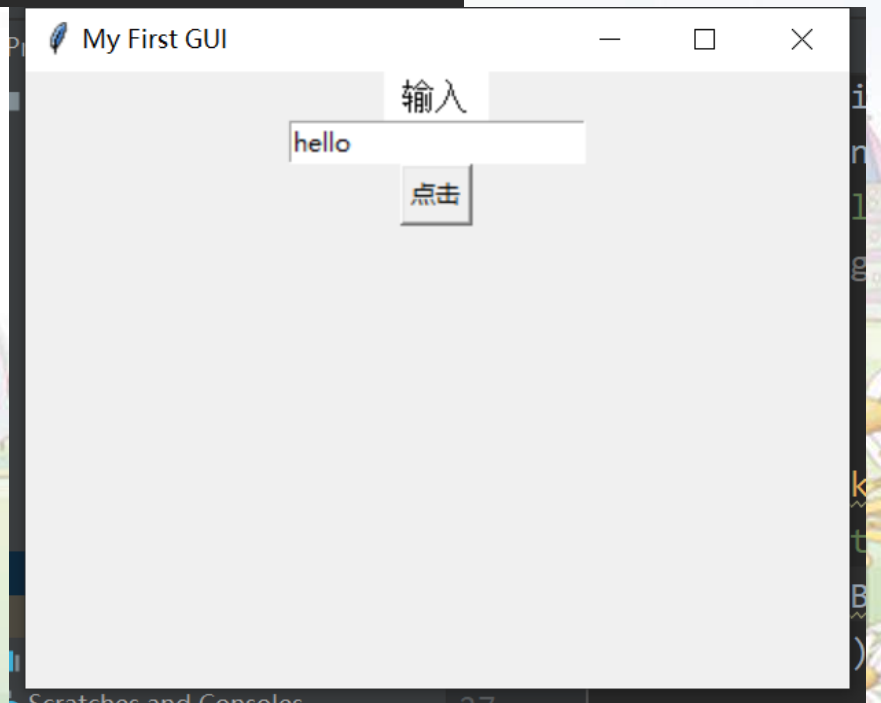
按钮：Button。

```
def btn_click():                    # 按钮点击触发方法
    print('btn_click')
button = tk.Button(root, text='点击', command=btn_click)
button.pack()
```

text：按钮文字。

command：按钮绑定的点击事件。

点击事件就是一个定义好的方法。

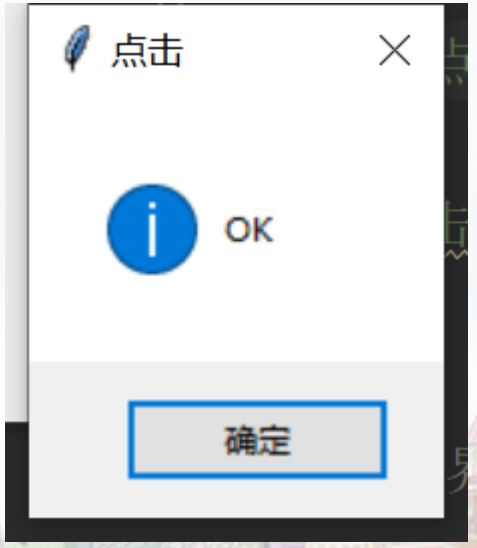


LV5

基础控件

消息弹窗：messagebox。

```
import tkinter.messagebox
tkinter.messagebox.showinfo('点击', 'OK')
```



参数1为消息标题。

参数2为消息内容。

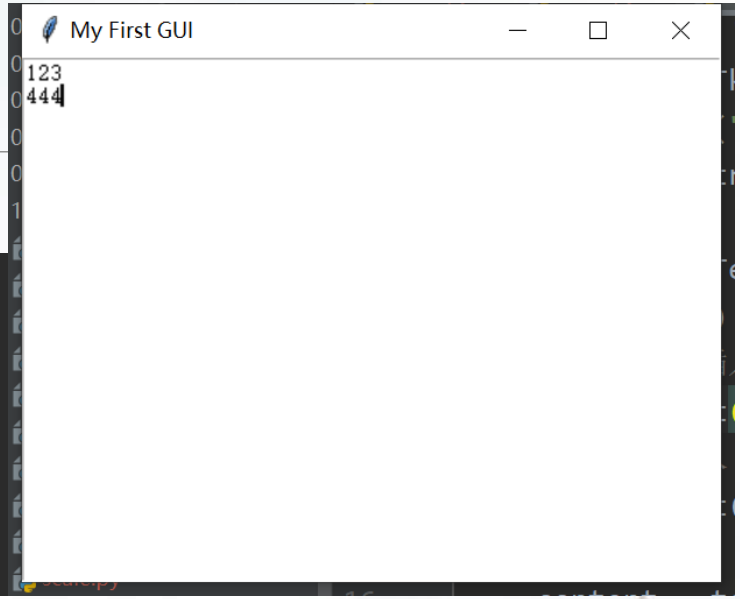
放入按钮点击事件中

```
def btn_click():
    tkinter.messagebox.showinfo('点击', 'OK')
```

lin029011@163.com

LV5

基础控件



多行文本框：Text、ScrolledText。

```
text = tk.Text(root)
text.pack()
text.insert(tk.INSERT, "123\n")      # 在光标出插入
text.insert(tk.END, "444")           # 在最后插入
content = text.get('0.0', tk.END)    # 获取内容
```

insert方法：插入内容。参数1是位置。INSERT代表在光标位置，END代表在最后。

get方法：获取内容。

ScrolledText与Text一样的用法，在右边多了一个拖动条。

```
import tkinter.scrolledtext
text = tkinter.scrolledtext.ScrolledText(root)
text.pack()
```

lin029011@163.com

LV5

容器布局

容器：Frame。

可以放入其他控件。

frame以root为根节点，

其他几个控件都以frame

为根节点。

这样就可以进行封装。

side：排列方式。

```
frame = tk.Frame(root)

label = tk.Label(frame, text="输入")
label.pack(side=tk.LEFT)

var = tk.StringVar()
entry = tk.Entry(frame, textvariable=var)
entry.pack(side=tk.LEFT)

def btn_click():
    tkinter.messagebox.showinfo('点击', 'OK')

button = tk.Button(frame, text='点击', command=btn_click)
button.pack(side=tk.RIGHT)

frame.pack(side=tk.TOP)
```

lin029011@163.com

LV5

容器布局

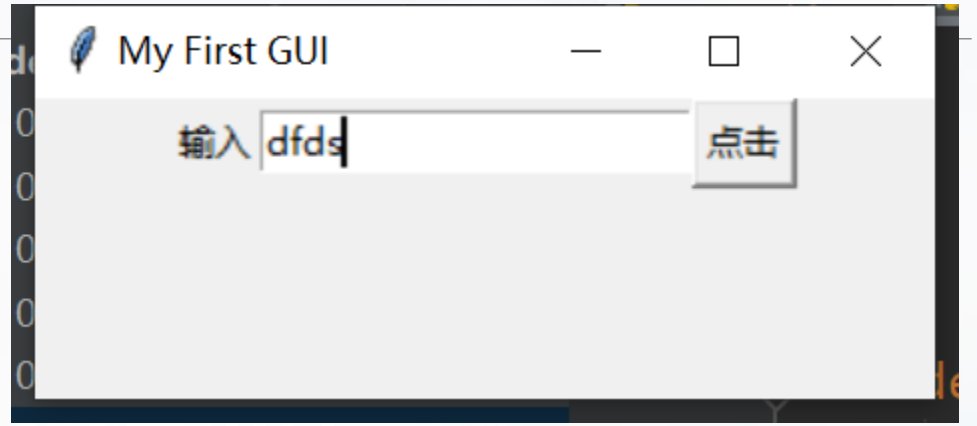
```
class MyFrame(tk.Frame):

    def __init__(self, master):
        tk.Frame.__init__(self, master)

        self.label = tk.Label(self, text="输入")
        self.label.pack(side=tk.LEFT)
        self.var = tk.StringVar()
        self.entry = tk.Entry(self, textvariable=self.var)
        self.entry.pack(side=tk.LEFT)
        self.button = tk.Button(self, text='点击', command=self.btn_click)
        self.button.pack(side=tk.RIGHT)

    def btn_click(self):
        tkinter.messagebox.showinfo('输入内容', self.var.get())
```

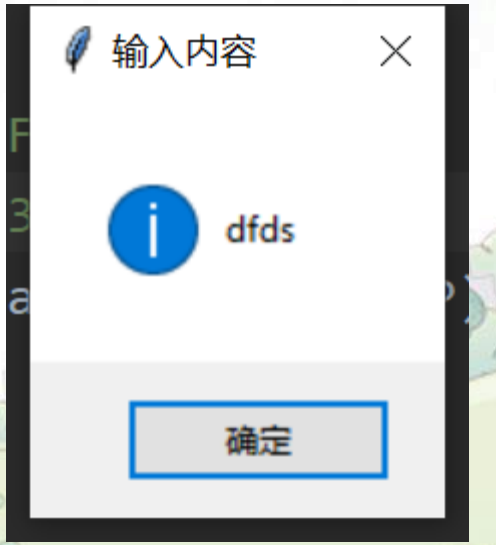
LV5

容器布局

嵌入root中，执行。

```
if __name__ == '__main__':
    root = tk.Tk()
    root.title("My First GUI")
    root.geometry('300x100')
    MyFrame(root).pack(side=tk.TOP)
    root.mainloop()
```





推荐封装类进行界面编写，方便扩展。

PyQt也是通过类进行界面编写的。

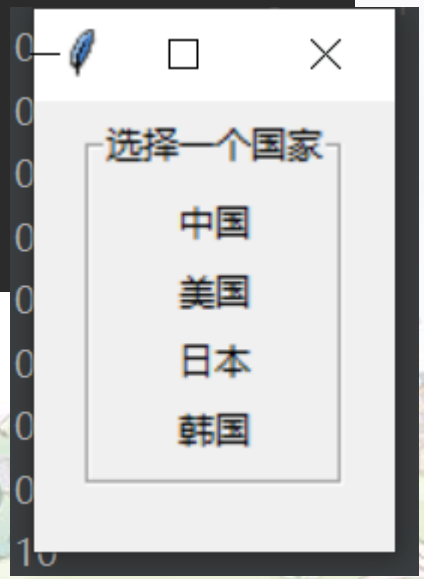
lin029011@163.com

LV5

容器布局

一种简单容器：LabelFrame。

```
frame = tk.LabelFrame(self, text="选择一个国家", padx=15, pady=15)
tk.Label(frame, text="中国").pack()
tk.Label(frame, text="美国").pack()
tk.Label(frame, text="日本").pack()
tk.Label(frame, text="韩国").pack()
frame.pack(padx=15, pady=15)
```



text：标签名称。

padx：布局参数，在横行方向左右各空出的长度。

pady：布局参数，在垂直方向上下各空出的长度。

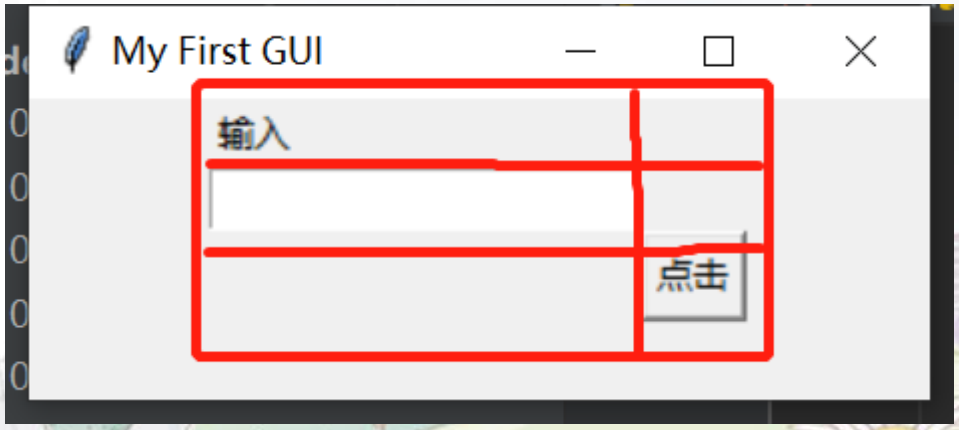
lin029011@163.com

LV5

容器布局

网格布局：grid，与pack对应。

```
self.label.grid(column=0, row=0, sticky=tk.W)
self.entry.grid(column=0, row=1, sticky=tk.W)
self.button.grid(column=1, row=2, sticky=tk.W)
```



label 被放在了第0列，第0行。

entry被放在了第0列，第1行。

button被放在了第1列，第2行。

column：列

row：行

sticky：内容对齐方式。N【上对齐】，S【下对齐】，W【左对齐】，E【右对齐】

lin029011@163.com

LV5

选择控件

选择框：Checkbutton。

```
def checkbutton_select(x):    #
    print(x.get())
check = tk.IntVar()
checkbutton = tk.Checkbutton(root, text="选择框", variable=check,
    command=lambda : checkbutton_select(check))
checkbutton.pack()
```

text：文字。

variable：绑定的变量。

state：设定为disabled时，不可操作。

command：绑定的命令。这里使用了lambda表达式来传递参数。

将check变量传递到checkbutton_select中去。

lin029011@163.com

LV5

选择控件

制作一个多选框：

```
self.check1 = tk.IntVar()
self.checkbutton1 = tk.Checkbutton(self, text="安卓", variable=self.check1)
self.checkbutton1.grid(column=0, row=0, sticky=tk.W)
self.check2 = tk.IntVar()
self.checkbutton2 = tk.Checkbutton(self, text="苹果", variable=self.check2)
self.checkbutton2.grid(column=1, row=0, sticky=tk.W)
self.check3 = tk.IntVar()
self.checkbutton3 = tk.Checkbutton(self, text="PC", variable=self.check3,
                                   state='disabled')
self.checkbutton3.select()  # 选择
self.checkbutton3.grid(column=2, row=0, sticky=tk.W)
```

尝试做一个全选按钮。

lin029011@163.com

LV5

选择控件

单选框：Radiobutton。经常与LabelFrame一起使用。

```
frame = tk.LabelFrame(root, text="选择一个国家", padx=5, pady=5)
self.value = tk.IntVar()
options = [('中国', 0), ('美国', 1), ('日本', 2), ('韩国', 3)]  # 选项
for t, v in options:
    tk.Radiobutton(frame, text=t, value=v, variable=self.value,
                   command=self.radio_select).pack()
frame.pack(padx=5, pady=5)
```

text：显示文字。

value：值。通过循环options生成所有的选项。

variable：绑定的变量。所有radiobutton都绑定同一个变量。

command：绑定的选择方法。所有radiobutton都绑定同一个事件方法。

LV5

选择控件

数值选择：Scale。

```
scale = tk.Scale(root, from_=0, to=5, tickinterval=1)
scale.pack()
scale_h = tk.Scale(root, from_=0, to=10, resolution=2, orient=tk.HORIZONTAL)
scale_h.pack()
```

from_：最小值。

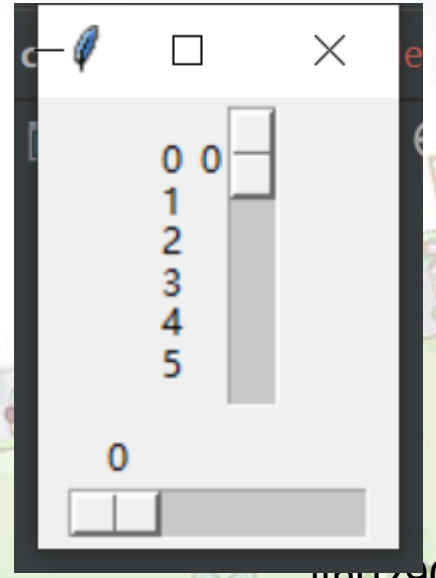
to：最大值。

tickinterval：刻度的步进。

resolution：每格的步进。

orient：方向。HORIZONTAL代表横向。

通过控件的get方法获得当前值，scale.get()。



lh029011@163.com

LV5

列表控件

列表：Listbox。

```
self.listbox = tk.Listbox(self, selectmode=tk.SINGLE)
for item in range(10):
    self.listbox.insert(tk.END, item)
self.listbox.pack(side=tk.LEFT)
```

selectmode：选择模式。SIGNGLE、BROWSE【单选】，MULTIPLE、EXTENDED【多选】

insert方法用于添加项，第一个参数是添加位置。

添加一个删除按钮，点击后删除选中项。

```
def delete(self):
    self.listbox.delete('active')    # 删除选中项
```

lin029011@163.com

LV5

列表控件

滚动条：Scrollbar。常用列表，多行文本框等控件配合使用。

```
scrollbar = tk.Scrollbar(self)
scrollbar.pack(side=tk.RIGHT, fill=tk.Y)

self.listbox = tk.Listbox(self, selectmode=tk.MULTIPLE,
                          yscrollcommand=scrollbar.set)
for item in range(10):
    self.listbox.insert(tk.END, item)
self.listbox.pack(side=tk.LEFT)

scrollbar.config(command=self.listbox.yview)
```

yscroolcommand：定义为scrollbar控件的set方法。

最后设置scrollbar的command为列表控件的yview方法。

lin029011@163.com

LV5

对话框

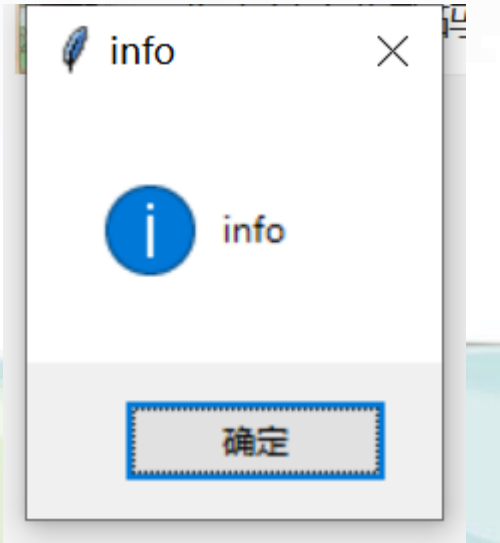
```
import tkinter.messagebox
```

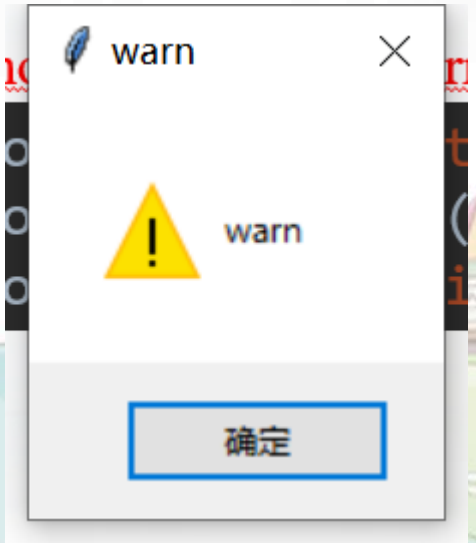
显示对话框：showinfo、showerror、showwarning。

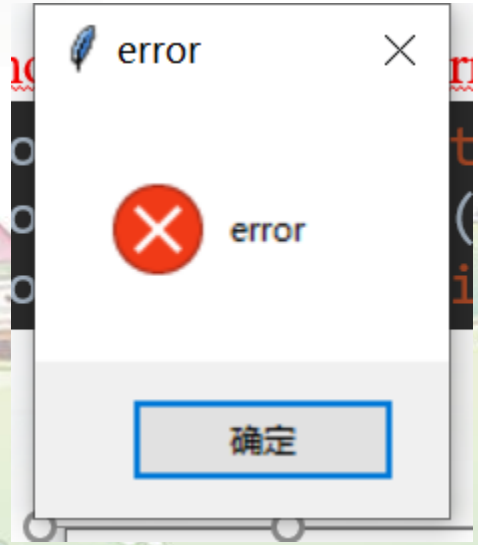
```
tkinter.messagebox.showinfo(title='info', message='info')
tkinter.messagebox.showwarning(title='warn', message='warn')
tkinter.messagebox.showerror(title='error', message='error')
```

title：标题。

message：信息。







lin029011@163.com

LV5

对话框

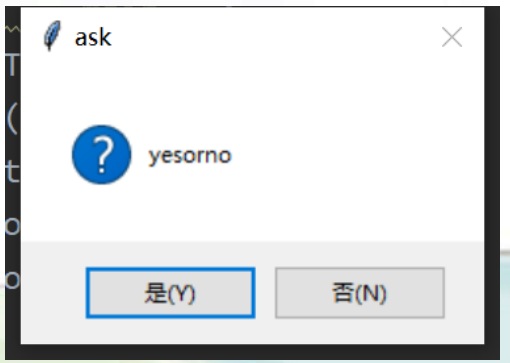
```
import tkinter.messagebox
```

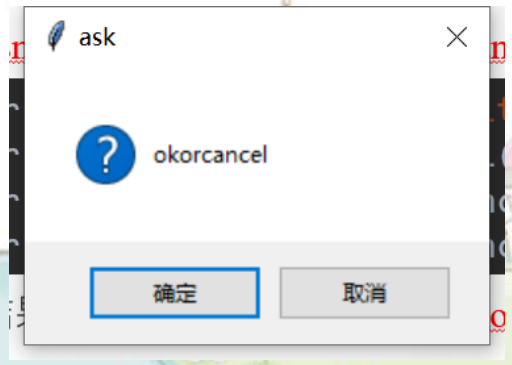
提问对话框： askyesno 、 askokcancel、 askretrycancel、 askyesnocancel。

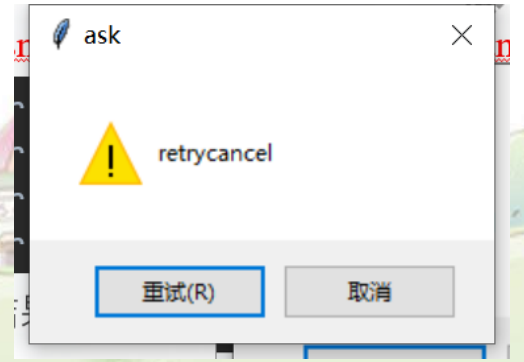
```
ask = tkinter.messagebox.askyesno(title='ask', message='yesorno')
ask = tkinter.messagebox.askokcancel(title='ask', message='okorcancel')
ask = tkinter.messagebox.askretrycancel(title='ask', message='retrycancel')
ask = tkinter.messagebox.askyesnocancel(title='ask', message='yesnocancel')
```

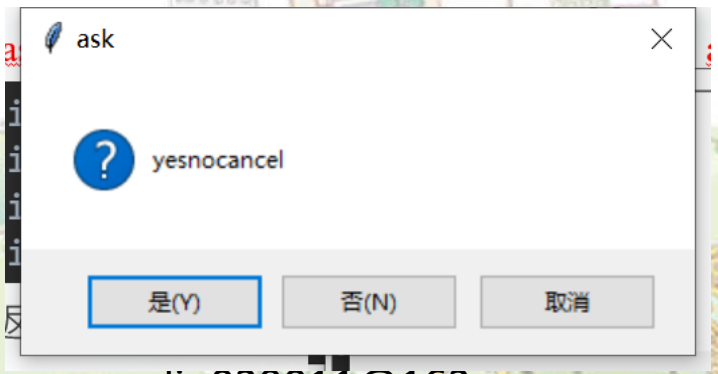
提问类会返回结果，True 或者 False。askyesnocancel多一个当选择取消时返回的是None。

之后可以根据返回的结果，继续编写逻辑。









lin029011@163.com

LV5

对话框

```
import tkinter.filedialog
```

文件对话框： askopenfile、 askopenfiles、 askopenfilename、 askopenfilenames 。

```
filename = tkinter.filedialog.askopenfilename(initialdir='C:\\')
filenames = tkinter.filedialog.askopenfilenames()
file = tkinter.filedialog.askopenfile(mode='r', filetypes=[('Python', '.py')])
files = tkinter.filedialog.askopenfiles(title='选择多个文件')
```

title：标题。

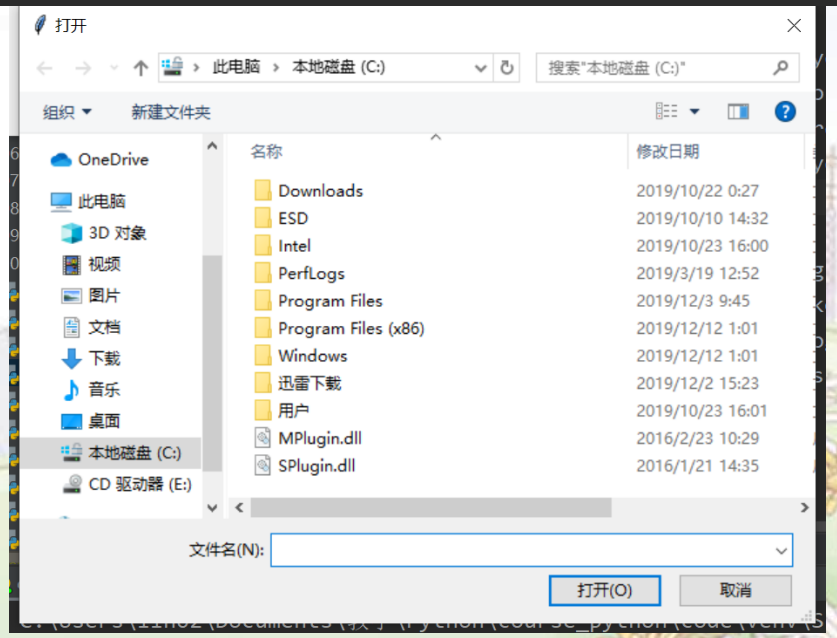
initialdir：初始文件夹。

filetypes：限定文件类型，列表中包含名称和后缀名。

filename方法返回选择的文件路径。

file方法返回打开的文件对象，mode是模式，默认为 r。

选择多个文件使用加 s 的方法。



LV5

对话框

```
import tkinter.filedialog
```

文件夹选择框：askdirectory。

```
dir = tkinter.filedialog.askdirectory(initialdir='C:\\')
```

与文件选择框类似，返回的是文件夹路径。

保存文件对话框： asksaveasfilename、 asksaveasfile。

```
filename = tkinter.filedialog.askopenfilename(filetypes=[('Python', '.py')])
file = tkinter.filedialog.asksaveasfile(defaultextension='py', initialdir='C:\\')
```

defaultextension：默认文件后缀名，当用户没有输入后缀时自动加上。

filename返回的只是文件路径。

file返回的则是文件对象，默认的模式是 w，当文件不存在时会自动创建。

lin029011@163.com

LV5

对话框

```
import tkinter.colorchooser
```

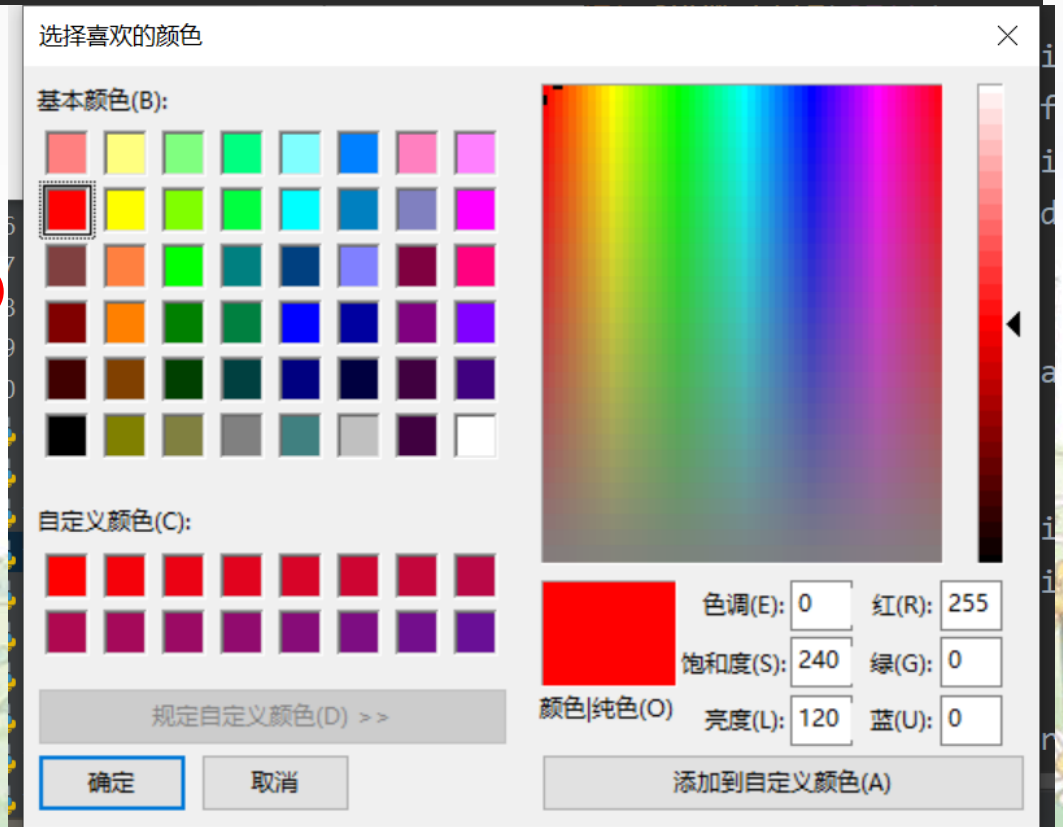
颜色选择对话框： askcolor。

```
color = tkinter.colorchooser.askcolor(color='red', title='选择喜欢的颜色')
print(color)
```

color：初始颜色。

返回的颜色数据

((47.18359375, 79.30859375, 208.8125), '#2f4fd0')



LV5

对话框

```
import tkinter.simpledialog
```

输入对话框： askinteger、 askfloat 、 askstring 。

```
i = tkinter.simpledialog.askinteger(title='simple', prompt='整数: ',
                                    minvalue=1, maxvalue=10)
f = tkinter.simpledialog.askfloat(title='simple', prompt='小数: ',
                                  initialvalue='1.1')
s = tkinter.simpledialog.askstring(title='simple', prompt='字符串: ')
```

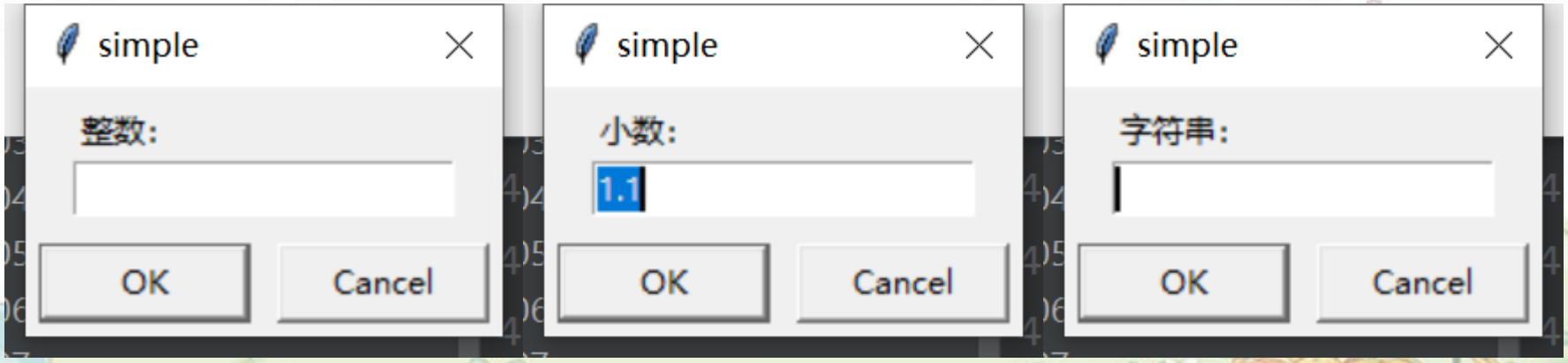
title：标题。

prompt：提示。

minvalue：最小值。

maxvalue：最大值。

initialvalue：初始值。



lin029011@163.com

LV5

对话框

自定义对话框：Toplevel 。继承该类，定义自己的对话框。

```
class MyDialog(tk.Toplevel):
    def __init__(self, parent):
        tk.Toplevel.__init__(self, parent)
```

parent：父节点。

我们直接套用raidobutton的代码进行布局，区别在于事件最后用destroy关闭对话框。

```
def radio_select(self):
    self.destroy()
```

对话框调用，直接创建就生成了对话框。

```
dialog = MyDialog(self)
self.wait_window(dialog)
print(dialog.value.get())
```

wait_window等待对话框返回。

返回后，可以直接获取对话框的成员变量作为返回值。

lin029011@163.com

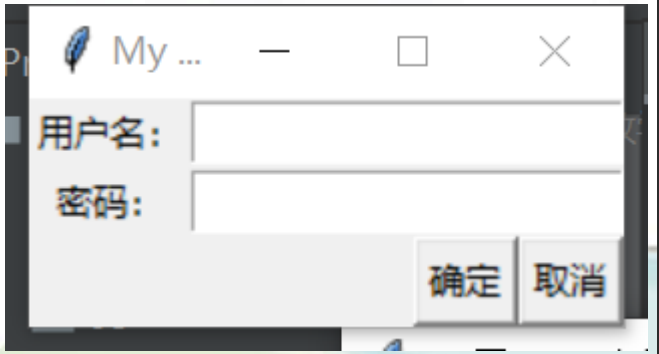
LV5

对话框

登录对话框：

调用获取登录数据

```
dialog = LoginDialog(self)
self.wait_window(dialog)
print(dialog.login_info)
```



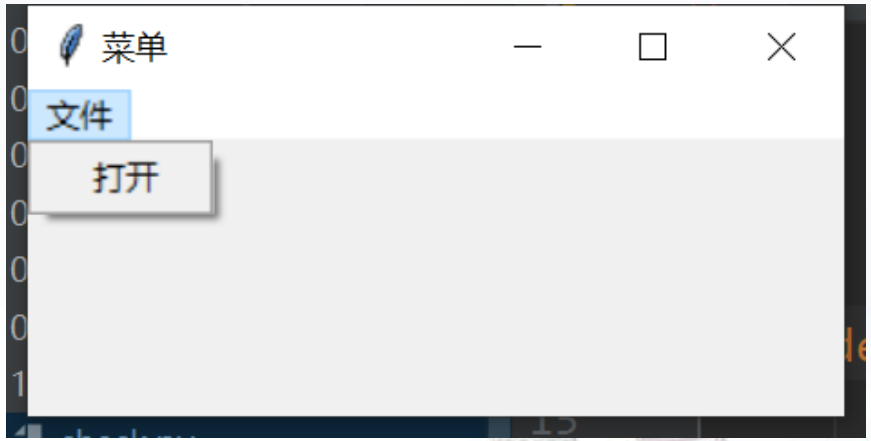

```
class LoginDialog(tk.Toplevel):

    def __init__(self, parent):
        tk.Toplevel.__init__(self, parent)

        tk.Label(self, text='用户名: ',).grid(column=0, row=0)
        tk.Label(self, text='密码: ').grid(column=0, row=1)

        self._username = tk.StringVar()
        self._password = tk.StringVar()
        tk.Entry(self, textvariable=self._username).grid(column=1, row=0)
        tk.Entry(self, textvariable=self._password).grid(column=1, row=1)

        btns = tk.Frame(self)
        tk.Button(btns, text="取消", command=self.cancel).pack(side=tk.RIGHT)
        tk.Button(btns, text="确定", command=self.ok).pack(side=tk.RIGHT)
        btns.grid(column=1, row=2, sticky=tk.E)

        self.login_info = {'username': None, 'password': None}

    def cancel(self):
        self.destroy()

    def ok(self):
        self.login_info['username'] = self._username.get()
        self.login_info['password'] = self._password.get()
        self.destroy()
```

LV5

菜单控件

顶部菜单：Menu。一般都直接属于最顶级对象，我们这里采用直接继承Tk对象的方式。

```
class Window(tk.Tk):

    def __init__(self):
        tk.Tk.__init__(self)
        self.title("菜单")
        self.geometry('300x100')
        # 添加菜单
        self.addMenu()

if __name__ == '__main__':
    Window().mainloop()
```



下面我们实现addMenu方法。

lin029011@163.com

LV5

菜单控件

```
def addMenu(self):
    menubar = tk.Menu(self)      # 菜单栏
    self.config(menu=menubar)    # 设置菜单

    menuFile = tk.Menu(menubar, tearoff=0)  # 一级菜单
    menubar.add_cascade(label="文件", menu=menuFile)  # 添加到菜单栏
    menuFile.add_command(label="打开", command=self.onMenuFileOpen)    # 添加子菜单
```

先定义一个menubar，作为整个菜单栏，后设置到窗口中。

再定义一个一级菜单menuFile，以menubar为父节点。tearoff=0代表不可分窗口。

add_cascade方法：添加菜单目录【含有子菜单】。menu：要添加的Menu。

add_command方法：添加菜单命令。command：绑定的方法。

```
def onMenuFileOpen(self):
    file = tkinter.filedialog.askopenfilename()
    print(file)
```

lin029011@163.com

LV5

菜单控件

```
import tkinter.ttk as ttk
```

选择菜单：OptionMenu。

ttk是tkinter库的升级版，基本上tkinter有的，ttk都有。

```
options = ['gbk', 'utf-8']
self.code = tk.StringVar()
menuCode = ttk.OptionMenu(self, self.code,
                          options[0], *options,
                          command=self.codeChange)

menuCode.pack()
```

参数2：绑定的内容。

参数3，4：初始选项，全部选项。这里加了一个 * ，是不定参数。

command：绑定的命令。

传入一个参数，为当前值。

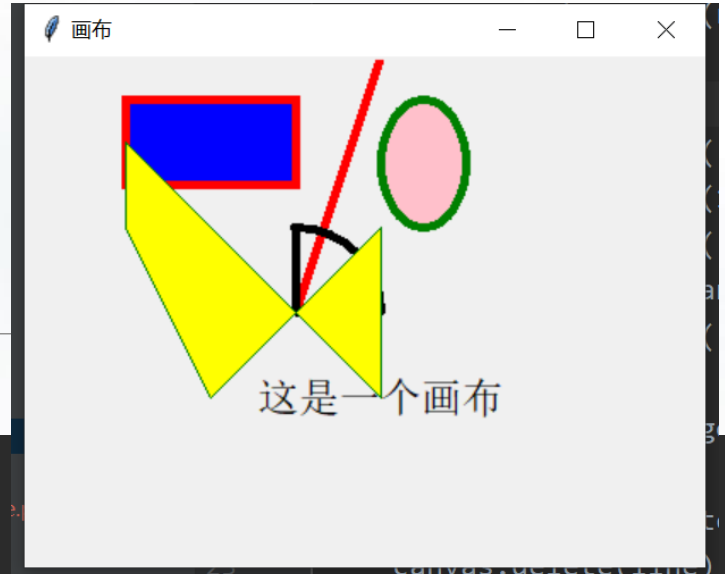
```
def codeChange(self, value):
    print(value)
```

lin029011@163.com

LV5

画布

画布：Canvas。用于画图。




```
canvas = tk.Canvas(root)
canvas.pack()

canvas.create_line((200, 0), (150, 150), width=5, fill="red")
canvas.create_arc((100, 100), (200, 200), width=5)
canvas.create_text((200, 200), text="这是一个画布", font=("宋体", 18))
canvas.create_rectangle((50, 25), (150, 75), fill='blue', outline='red', width=5)
canvas.create_oval((200, 25), (250, 100), fill='pink', outline='green', width=5)
point = [(50, 50), (50, 100), (100, 200), (200, 100), (200, 200)]
canvas.create_polygon(point, outline='green', fill='yellow')

line = canvas.create_line((0, 0), (100, 100), width=5, fill="red")
canvas.delete(line)
```

LV5

画布

create_line ：直线（起始坐标，终点坐标，width=线宽，fill=颜色，dash=虚线）

create_arc ：圆弧（起始坐标，终点坐标，width=线宽，fill=颜色，dash=虚线）

create_text ：文字（坐标，text=文字，font=字体）

create_rectangle ：矩形（左上点坐标，右下点坐标，fill=填充颜色，outline=边框颜色）

create_oval ：椭圆（左上点坐标，右下点坐标，fill=填充颜色，outline=边框颜色）

create_polygon ：多边形（所有顶点的坐标，fill=填充颜色，outline=边框颜色）

delete：删除一条图形。

lin029011@163.com

LV5

事件绑定

对于电脑软件来说，事件主要来自鼠标、键盘、以及系统事件。

前面我们已经接触过一些控件的基本事件通过command来绑定。

事件绑定：bind。所有控件都有这个方法。

参数1：事件名称。

参数2：绑定的方法。

方法中默认传入事件参数。

点击鼠标右键和输入内容时，就会输出。

```
<ButtonPress event num=1 x=213 y=202>
<KeyPress event keysym=Control_L keycode=17 x=158 y=404>
```

```
def key(event):
    print(event)

def callback(event):
    print(event)

text = tk.Text(root)
text.bind('<Key>', key)
text.bind('<Button-1>', callback)
text.pack(padx=5, pady=5)
```

lin029011@163.com

LV5

事件绑定

事件类型

<Button-1>：鼠标点击，1代表左键，2代表中键，3代表右键。

<B1-Motion>：鼠标拖动。

<ButtonRelease-1>：鼠标点击释放。

<Double-Button-1>：双击鼠标。

<Enter>、<Leave>：鼠标进入、离开控件。

<Key>：键盘点击。

<KeyPress-A> ：键盘点击某个按键。A可以替换称对应的按键名称。

<Control-A>：Ctrl+A。同理，Control可以替换称Alt、Shift等。

<Control-Shift-H>：三个按键事件。

lin029011@163.com

LV5

事件绑定

<KeyRelease-A>：键盘释放。

<Configure>：当控件的位置或大小改变时触发。

<Activate>、<Deactivate>：控件激活。

<Destroy>：控件被销毁。

<FocusIn>、<FocusOut>：控件获得焦点。

<Visibility>：控件可见。

lin029011@163.com

LV5

事件绑定

事件对象：event。

```
type = event.type                    # 事件类型
widget = event.widget                # 触发事件的控件
x, y = event.x, event.y              # 触发位置
char = event.char                    # 键盘事件内容
num = event.num                      # 鼠标事件内容
width, height = event.width, event.height    # 控件新大小
```

lin029011@163.com

LV5

练习

1、编写一个简易计算器。

2、编写一个简易的文本编辑器。

lin029011@163.com