1. GUI基础 (5,2,1)

*#将tkinter的全部内容引入到该程序中*

**from** tkinter **import** \*

*#创建根窗体 不需要在类名* *Tk前加模块名*

root = Tk()

*#根窗体标题*

root.title(**'simple GUI'**)

*#设置窗体大小 以字符'x'隔开 单位像素*

root.geometry(**'200x200'**)

*#在根窗体上创建一个框架,用来承载其他小部件*

*#必须将小部件的容器(master,用于承载这个小部件的东西)传给构造器*

*#这里将root传给这个框架的构造器,于是新框架被放到root窗体*

app = Frame(root)

*#grid()是所有小部件都有的方法,它关联一个布局控制器*

app.grid()

*#在框架中创建一个标签*

lb1 = Label(app,text = **'我是标签!'**)

*#调用grid()方法*

lb1.grid()

*#1.在框架中创建一个按钮*

bttn1 = Button(app,text = **'无用按钮'**)

bttn1.grid()

*#2.创建app框架中的第二个按钮*

bttn2 = Button(app)

bttn2.grid()

*#在小部件被创建出来后,仍可以对其进行修改configure()*

bttn2.configure(text = **'我也是'**)

*#3.创建app框架中的第三个按钮*

bttn3 = Button(app)

bttn3.grid()

*#使用类似于字典的方式来访问按钮的'text'选项,该选项的名字就是键(字符串形式)*

bttn3[**'text'**] = **'这里也是'**

*#启动事件循环*

root.mainloop()

2.类(3 2 5)

**class** Aplication(Frame):

**def** \_\_init\_\_(self,master):

super().\_\_init\_\_(master)

self.grid()

self.create\_widgets()

首先，\_\_init\_\_有两个参数。第一个是self，即Application对象本身。第二个是master，在Tkinter中，一个控件可能属于另一个控件，这时另一个控件就是这个控件的master。(默认一个窗口没有master，因此master有None的默认值。)

第二行tk.Frame.\_\_init\_\_(self, master)调用Application的父类Frame的\_\_init\_\_函数初始化Application类的Frame类部分。\_\_init\_\_函数的定义如下：

因为这里不是直接用对象调用函数，又不是一般意义的创建对象（例如后面的quitButton），而是初始化自身的一部分，因此需要传入self。我们还传入了Application.\_\_init\_\_的master参数，作为Frame.\_\_init\_\_的master。将容器传给框架构造器

**from** tkinter **import** \*

*#基于Frame的Application类*

**class** Aplication(Frame):

**def** \_\_init\_\_(self,master):

*# super(Aplication,self).\_\_init\_\_(master)*

super().\_\_init\_\_(master)

*# super().\_\_init\_\_()*

#super(Aplication,self).\_\_init\_\_(master)

*# 调用超类的构造器，使Application对象能得到超类中已经实现的方法*

*# 注意 这里传入的参数是当前Application的对象容器 之后调用这个对象*

*# 的create\_widget()方法*

self.grid()

self.create\_widgets()

**def** create\_widgets(self):

self.bttn1 = Button(self,text = **'无用按钮'**)

self.bttn1.grid()

self.bttn2 = Button(self)

self.bttn2.grid()

self.bttn2.configure(text = **'我也是'**)

self.bttn3 = Button(self)

self.bttn3.grid()

self.bttn3[**'text'**] = **'这里也是'**

root = Tk()

root.title(**'GUI'**)

root.geometry(**'200x85'**)

app = Aplication(root)

root.mainloop()

3.事件计数器

**from** tkinter **import** \*

*#基于Frame的Application类*

**class** Aplication(Frame):

**def** \_\_init\_\_(self,master):

super(Aplication,self).\_\_init\_\_(master)

self.grid()

self.bttn\_clicks = 0

self.create\_widgets()

*#创建小部件*

**def** create\_widgets(self):

self.bttn = Button(self)

*# self.bttn.configure(text = 'Total Click:0')*

self.bttn[**'text'** ]= **'Total Click:0'**

*#绑定能够事件计数器 设置command*

self.bttn[**'command'**] = self.update\_count

self.bttn.grid()

*#创建事件计数器*

**def** update\_count(self):

self.bttn\_clicks += 1

*# self.bttn.configure(text = 'Total Click:'+str(self.bttn\_clicks))*

self.bttn[**'text'**] = **'Total Click:'**+str(self.bttn\_clicks)

*#程序主体*

root = Tk()

root.title(**'GUI'**)

root.geometry(**'200x85'**)

app = Aplication(root)

root.mainloop()

4.布局管理器

*#使用Text（多行文本）和Entry(单行文本)小部件以及Grid布局管理器*

**from** tkinter **import** \*

*#基于Frame的Application类*

**class** Aplication(Frame):

**def** \_\_init\_\_(self,master):

super(Aplication,self).\_\_init\_\_(master)

self.grid()

self.create\_widgets()

*#创建小部件*

**def** create\_widgets(self):

self.inst\_lb1 = Label(self,text=**'输入正确密码就可以得到答案了'**)

*#Grid布局管理器指定这个标签的具体位置*

self.inst\_lb1.grid(row =0,column = 0,columnspan = 2,sticky = W)

*#row column定义对象在容器中的具体位置,将容器想象为单元格。*

*#columnspan 参数用于横跨多列防止小部件*

*#rowspan 参数用于横跨多行防止小部件*

*#sticky用于调整小部件在这个单元格中的位置* *N S E W*

*#创建表示密码的标签*

self.pw\_lb1 = Label(self,text=**'password:'**)

self.pw\_lb1.grid(row=1,column=0,sticky=W)

*#创建用于接受密码的Entry小部件 单行文本框*

self.pw\_ent = Entry(self)

self.pw\_ent.grid(row = 1,column = 1,sticky = W)

*#创建提交按钮 绑定到reveal()方法*

self.submit\_bttn =Button(self,text=**'submit'**,command=self.reveal)

self.submit\_bttn.grid(row=2,column=0,sticky=W)

*#创建Text小部件*

*#wrap决定文本的换行方式，WORD在文本框右边缘时自动换行*

*# CHAR只将下一个字符放到下一行,NONE不换行*

self.secret\_txt = Text(self,width=35,height=5,wrap=WORD)

self.secret\_txt.grid(row =3,column = 0,columnspan = 2,sticky = W)

**def** reveal(self):

*#判断密码是否正确*

*#调用Entry小部件的get()方法得到文本,#下述方法entry和text都有*

contents=self.pw\_ent.get()

**if** contents == **'secret'**:

message = **'这里有可以让你长生的秘诀，万古长青'**

**else**:

message = **'密码错误，你离长生的秘密还差的很远'**

*#将其插入到text小部件，首先删除text小部件的所有文本*

self.secret\_txt.delete(0.0,END)

*#表示从第0行第0列开始删除文本一直到文本末尾*

*#insert接收插入位置和字符串*

self.secret\_txt.insert(0.0,message)

*#程序主体*

root = Tk()

root.title(**'GUI'**)

root.geometry(**'300x150'**)

app = Aplication(root)

root.mainloop()

5.复选框

*#复选框允许用户从一组选项中选取任意数量的的选项*

**from** tkinter **import** \*

*#基于Frame的Application类*

**class** Aplication(Frame):

**def** \_\_init\_\_(self,master):

super(Aplication,self).\_\_init\_\_(master)

self.grid()

self.create\_widgets()

*#创建小部件*

**def** create\_widgets(self):

*#创建描述标签 在tkinter中所有GUI元素都通过容器跟程序保持联系*

*#如果不需要直接访问某个小部件,就不需要将其赋值给变量*

Label(self,text = **'请输入你喜欢的电影种类'**)\

.grid(row = 0,column = 0,sticky=W)

*#创建操作说明标签*

Label(self,text = **'请选择:'**).grid(row = 1,column = 0,sticky=W)

*#创建复选框* *BooleanVar类*

*#任何复选框都要有一个特殊对象与之关联,以便自动反应该复选框的状态*

*#1.创建喜剧复选框*

self.likes\_comedy = BooleanVar()

Checkbutton(self,

text = **'comedy'**,

variable = self.likes\_comedy,

command = self.update\_text

).grid(row = 2,column = 0,sticky = W)

*# 2.创建戏剧复选框*

self.likes\_drama = BooleanVar()

Checkbutton(self,

text=**'drama'**,

variable=self.likes\_drama,

command=self.update\_text

).grid(row=3, column=0, sticky=W)

*# 3.创建浪漫复选框*

self.likes\_romance= BooleanVar()

Checkbutton(self,

text=**'romance'**,

variable=self.likes\_romance,

command=self.update\_text

).grid(row=4, column=0, sticky=W)

*#创建文本框*

self.result\_text = Text(self,width=40,height = 5,wrap=WORD)

self.result\_text.grid(row=5,column=0,columnspan=3)

*#根据复选框选取状态更新文本内容*

**def** update\_text(self):

likes = **''**

**if** self.likes\_comedy.get():

likes += **'你喜欢喜剧电影'**

**if** self.likes\_drama.get():

likes += **'你喜欢戏剧电影'**

**if** self.likes\_romance.get():

likes += **'你喜欢浪漫电影'**

self.result\_text.delete(0.0,END)

self.result\_text.insert(0.0,likes)

*#程序主体*

root = Tk()

root.title(**'GUI'**)

*# root.geometry('300x150')*

app = Aplication(root)

root.mainloop()

6.单选框

*#单选框 只允许在一组选项中选取一个*

**from** tkinter **import** \*

*#基于Frame的Application类 定义构造器初始化新的Application对象*

**class** Aplication(Frame):

**def** \_\_init\_\_(self,master):

super(Aplication,self).\_\_init\_\_(master)

self.grid()

self.create\_widgets()

*#创建小部件*

**def** create\_widgets(self):

Label(self,text = **'请输入你喜欢的电影种类'**)\

.grid(row = 0,column = 0,sticky=W)

*#创建操作说明标签*

Label(self,text = **'请选择1个:'**).grid(row = 1,column = 0,sticky=W)

*#创建单选框*

*#一组单选框只需共享一个用于说明'选中'的特殊对象*

*#该对象是StringVar类*

*#创建共享变量 用set()方法将其初始值设为None*

self.favorite = StringVar()

self.favorite.set(**None**)

*#1.创建comedy单选框*

*#当comedy被选中时,*

*# 由self.favorite所引用的StringVar存储字符串'comedy'*

Radiobutton(self,

text = **'Comedy'**,

variable = self.favorite,

value = **'comedy'**,

command = self.update\_text

).grid(row = 2,column = 0,sticky=W)

*# 2.创建drama单选框*

Radiobutton(self,

text=**'Drama'**,

variable=self.favorite,

value=**'drama'**,

command=self.update\_text

).grid(row=3, column=0, sticky=W)

*# 3.创建romance单选框*

Radiobutton(self,

text=**'Romance'**,

variable=self.favorite,

value=**'romance'**,

command=self.update\_text

).grid(row=4, column=0, sticky=W)

*#创建用于显示结果的文本框*

self.result\_text = Text(self,width = 40,height=5,wrap=WORD)

self.result\_text.grid(row = 5,column = 0,columnspan=3)

**def** update\_text(self):

message = **'你最喜欢的电影类型:'**

message += self.favorite.get()

self.result\_text.delete(0.0,END)

self.result\_text.insert(0.0,message)

*#程序主体*

root = Tk()

root.title(**'GUI'**)

*# root.geometry('300x150')*

app = Aplication(root)

root.mainloop()

7.GUI项目

*#mad lib 在用户帮助下编写故事程序 需要用户提供人名 名词 动词*

**from** tkinter **import** \*

*#基于Frame的Application类 定义构造器初始化新的Application对象*

**class** Aplication(Frame):

**def** \_\_init\_\_(self,master):

super(Aplication,self).\_\_init\_\_(master)

self.grid()

self.create\_widgets()

*#创建小部件*

**def** create\_widgets(self):

Label(self,text=**'请输入故事的基本要素.'**).\

grid(row=0,column=0,columnspan=2,sticky=W)

*#创建 人名 名词 动词 的标签和文本框*

Label(self, text=**'人名:'**). \

grid(row=1, column=0,sticky=W)

self.person\_txt = Entry(self)

self.person\_txt.grid(row=1, column=1,sticky=W)

Label(self, text=**'名词:'**). \

grid(row=2, column=0,sticky=W)

self.noun\_txt = Entry(self)

self.noun\_txt.grid(row=2, column=1, sticky=W)

Label(self, text=**'动词:'**). \

grid(row=3, column=0,sticky=W)

self.verb\_txt = Entry(self)

self.verb\_txt.grid(row=3, column=1, sticky=W)

*#创建用于形容词的复选框*

Label(self, text=**'形容词:'**). \

grid(row=4, column=0, sticky=W)

self.is\_itchy = BooleanVar()

Checkbutton(self,

text = **'渴望的'**,

variable = self.is\_itchy,

).grid(row=4, column=1, sticky=W)

self.is\_joyous = BooleanVar()

Checkbutton(self,

text=**'快乐的'**,

variable=self.is\_joyous,

).grid(row=4, column=2, sticky=W)

self.is\_electric = BooleanVar()

Checkbutton(self,

text=**'激动的'**,

variable=self.is\_electric,

).grid(row=4, column=3, sticky=W)

*#创建用于身体部位的标签 变量以及单选框*

Label(self,text = **'身体部分:'**).grid(row=5, column=0, sticky=W)

self.body\_part = StringVar()

self.body\_part.set(**None**)

body\_parts = [**'肚脐'**,**'大脚趾'**,**'太阳穴'**]

column = 1

**for** part **in** body\_parts:

Radiobutton(self,

text = part,

variable = self.body\_part,

value = part).grid(row=5, column=column, sticky=W)

column += 1

*#创建提交按钮*

Button(self,text = **'单击生成故事'**,command = self.tell\_story).\

grid(row=6, column=0, sticky=W)

*#创建用于显示结果的文本框*

self.story\_txt = Text(self,width=75,height=10,wrap=WORD)

self.story\_txt.grid(row=7, column=0, columnspan=3)

**def** tell\_story(self):

person = self.person\_txt.get()

noun = self.noun\_txt.get()

verb = self.verb\_txt.get()

adjectives = **''**

**if** self.is\_itchy.get():

adjectives += **'渴望的'**

**if** self.is\_joyous.get():

adjectives += **'快乐的'**

**if** self.is\_electric.get():

adjectives += **'激动的'**

body\_part = self.body\_part.get()

*#创建故事*

story = **'有一个著名的探险者'**

story += person

story += **'.最近他在极地地区发现了一座只存在与神话中的城市'**

story += noun.title()

story += **'根据神话所述'**

story += noun

story += **'这个城市是由一个'**

story += adjectives

story += **'勇者建立的,他'**

story += verb

story += **'存在于城市周边的恶魔.'**

story += person

story += **'指着自己的'**

story += body\_part

story += **',决定一定发掘出勇者与恶魔最后的故事.'**

*#显示故事*

self.story\_txt.delete(0.0,END)

self.story\_txt.insert(0.0,story)

*#程序主体*

root = Tk()

root.title(**'GUI'**)

*# root.geometry('300x150')*

app = Aplication(root)

root.mainloop()