Создание графического интерфейса на языке Python.

Введение в tkinter

Полезные ссылки

- Документация по tkinter – https://docs.python.org/3/library/tkinter.html
- Хороший онлайн-учебник с примерами – https://metanit.com/python/tkinter/

Импортирование tkinter

Рассмотрим создание графического интерфейса пользователя средствами Python. Для этого воспользуемся библиотекой tkinter. Первое, что нам понадобится — импортируем tkinter, а потом создадим корневое окно приложения, вызвав конструктор Tk().

```
from tkinter import *
root = Tk()
```

Именно в корневом окне root далее будут располагаться все оконные элементы — кнопки, поля, метки, списки и т. д.

Если запустить программу сейчас, то никакого окна вы пока что и не увидите. Добавим в конце строку

```
root.mainloop()
```

Теперь после запуска вы увидите окно, пока что ничего не содержащее.

Кнопки и обработчики событий

Добавим в это окно виджет-кнопку и определим для нее обработчики события. Пусть щелчок по кнопке левой клавишей мыши выводит в консоль сообщение о нажатии кнопки, а щелчок по правой клавише — печатает в консоль приветствие. Для этого сначала запишем обработчики событий в виде функций, потом займемся описанием кнопки.

```
from tkinter import *
root = Tk()

def but_command(event):
    print("Hey!")

def clicked_command(event):
    print("I'm clicked")

but = Button(root, text="Нажми на меня")
but.bind("<Button-1>", clicked_command)
but.bind("<Button-3>", but_command)
but.pack()


root.mainloop()
```

Поясним кратко, что здесь записано. Функции объявляются при помощи ключевого слова def. В качестве единственного параметра функция для обработки события должна принимать именно объект-событие. Далее создаем объект-кнопку, для которой указываем родительское окно (root), и свойство-текст, что будет на ней написан. Далее назначаем кнопке обработчики события через метод bind: первым параметром указываем для какого события назначаем обработчик, вторым параметром назначаем сам обработчик (здесь <Button-1> означает щелчок левой клавишей мыши, <Button-3> - правой, а <Button-2> зарезервирован для средней кнопки, если она есть, либо для одновременного нажатия двух кнопок). Далее кнопку надо разместить в окне — для этого применяем метод pack(). И потом запускаем сценарий окна через root.mainloop(). Если теперь вы запустите программу и понажимаете на кнопку левой-правой клавишами мыши, в консоли должны появиться различные сообщения.

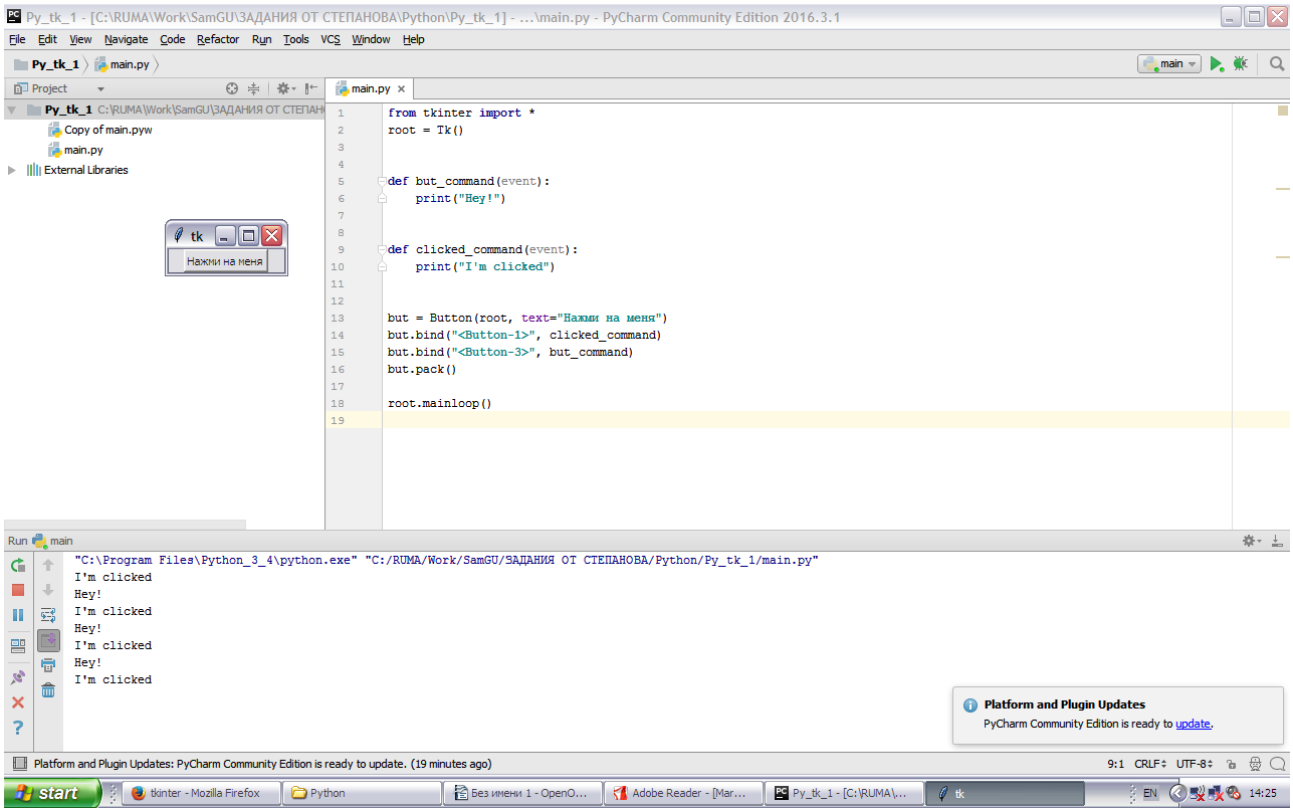

Рис. 1. Обработчики события для кнопки

Примечание 1. При помощи метода bind можно делать привязку обработчиков самым различным событиям. Более подробную справку рекомендую смотреть в книге М. Лутца «Программирование на Python. Ч.1».

Примечание 2. Кроме метода pack существуют и другие способы размещения виджетов в окне приложения, с ними мы ознакомимся немного позже.

2. Метка. Добавим в наше приложение виджет-метку и будем по щелчку левой клавиши мыши с зажатой клавишей Ctrl выводить некоторое случайное число в эту метку.

Допишем перед строкой root.mainloop() следующий фрагмент кода:

```
import random #для генератора случайных чисел
```

```
lb = Label(text='Йа метко') #собственно создаем саму метку

def lbl_command(event):  # определяем обработчик события
    lb['text'] = str(random.random())
                          #к свойствам виджетов можно обращаться
                          #по именам, используя их в виде индексов

lb.bind("<Control-Button-1>", lbl_command)
                                        #делаем привязку события
lb.pack() # и размещаем метку
```

Теперь, запустив проект, и нажимая на метку мышью (с зажатым Ctrl), можно убедиться, что каждый раз в нее записывается случайное число в диапазоне от 0 до 1.

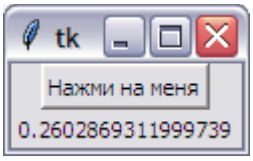

Рис. 2. Вывод случайного числа в метку

3. Текстовое поле ввода. Для добавления текстового поля ввода (аналог TEdit в Delphi) нам понадобится виджет Entry. Пусть текст из текстового поля копируется в нашу метку после нажатия кнопки (с зажатым Shift). Допишем перед последней строкой следующий фрагмент кода:

```
ent = Entry() #просто пустое текстовое поле ввода стандартного размера
ent.pack() #размещаем его

but1 = Button(text="В метку") #заводим кнопку

def printer(event):          #заводим обработчик для кнопки
```

```
    lb['text']=ent.get()

but1.bind("<Shift-Button-1>", printer) # привязываем событие
but1.pack() #и размещаем кнопку
```

Теперь посмотрим, что получилось:

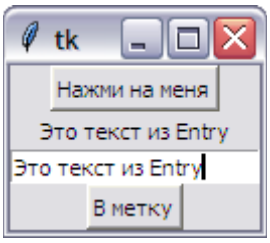

Рис. 3. Копирование текста из текстового поля в метку

4. Текстовая область. Аналогом компонента TMemo в Delphi в Python является компонент Text. Разместим небольшую текстовую область в нашем окне, и пусть щелчок по еще одной кнопке копирует содержимое области в консоль. Допишем код (как обычно, перед root.mainloop())

```
field = Text(width=20, height=6) #наша текстовая область, 20 символов длиной,
                                 # 5 строк высотой
field.pack()  # размещаем область

but2 = Button(text="В консоль")

# в обработчике мы заберем текст из области, первый параметр показывает
индексы,
# начиная с которых надо делать выборку (строка-столбец, при этом строки
# нумеруются с 1, столбцы — с нуля), второй параметр — до какого элемента
```

```
def printer2(event):
    print("start", field.get('1.0', END), "end")

but2.bind("<Button-1>", printer2)
but2.pack()
```

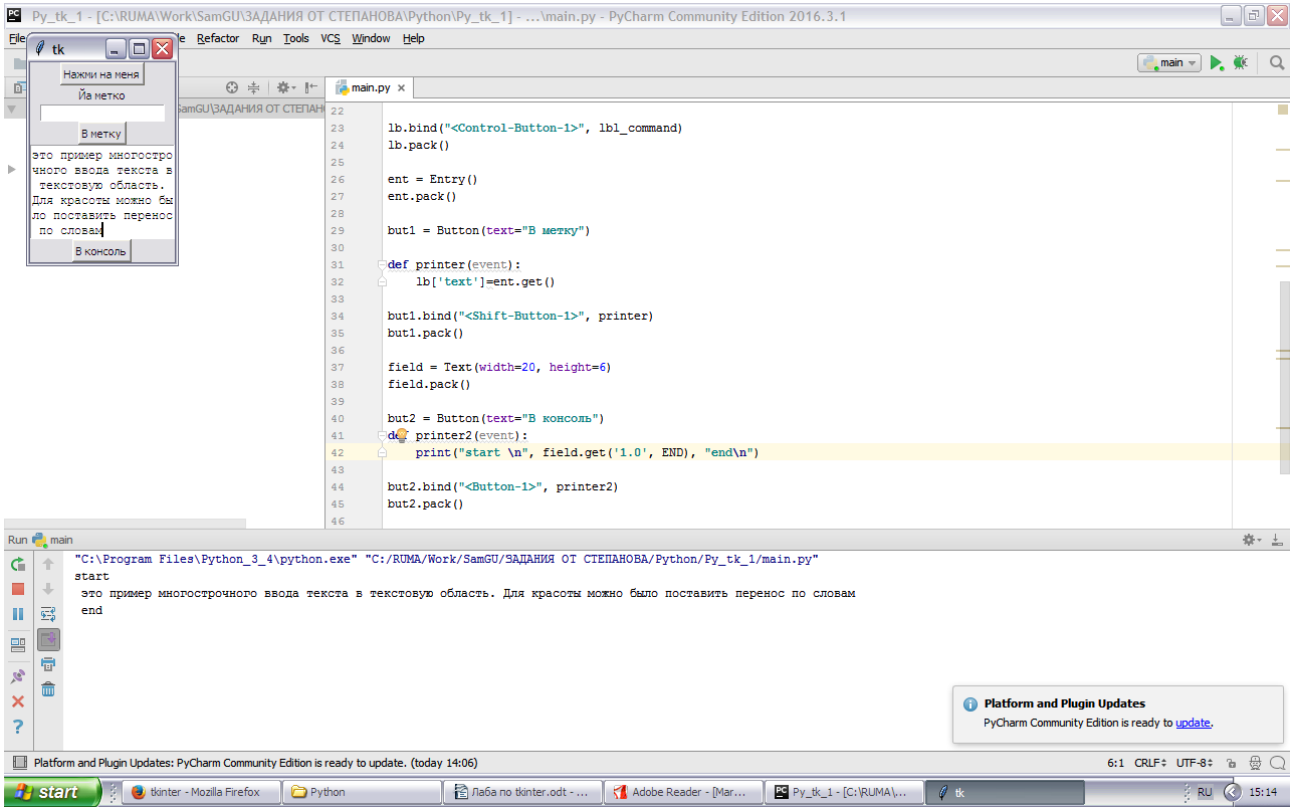

Рис. 4. Текстовая область

Сохраним этот пример в модуле с названием But-n-Text.py

5. Радиокнопки. Теперь научимся делать радиокнопки и связывать значения некоторой переменной и номер выбранного переключателя. Откроем новый python-файл с именем Check-n-Radio.py и запишем там следующий код:

```
from tkinter import *
root = Tk()

def change(): # это наш обработчик выбора радиокнопки
    print(var.get()) # печатаем в консоль содержимое ассоциированной
переменной

var=IntVar() #сама ассоциированная переменная, целочисленная
var.set(1) # установим ее равной 1

rad0 = Radiobutton(root,text="Первая",
        variable=var,value=0, command=change) # и при создании радиобаттона
                    # сопоставим им значение переменной
                    # мы таким образом включаем радиобаттоны в одну
радиогруппу
                    # и можем получать значения переменной для каждой
радиокнопки
rad1 = Radiobutton(root,text="Вторая",
        variable=var,value=1, command=change)
rad2 = Radiobutton(root,text="Третья",
        variable=var,value=2, command=change)
rad0.pack()
rad1.pack()
rad2.pack()
root.mainloop()
```

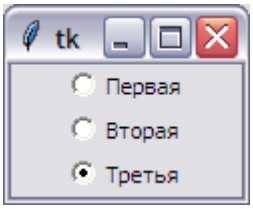

Рис. 5. Переключатели (радиокнопки)

6. Флажки. Добавим теперь флажки в наш модуль и ассоциируем с ними переменные. Для включенного (поднятого) и выключенного флажка переменная может обладать различными значениями, и это при необходимости можно использовать в программе.

```
def change1():      # обработчик события изменения состояния флажка 1
    print(c1.get())

def change2():      # обработчик события изменения состояния флажка 2
    print(c2.get())

c1 = IntVar() # ассоциированные переменные
c2 = IntVar()

# создадим сами флажки, привяжем к ним переменные
# при включении флажка переменная получает значение 1 (2)
# при выключении - 0
che1 = Checkbutton(root, text="Первый флажок",
                   variable=c1, onvalue=1, offvalue=0, command=change1)
che2 = Checkbutton(root, text="Второй флажок",
                   variable=c2, onvalue=2, offvalue=0, command=change2)
che1.pack()
che2.pack()
```

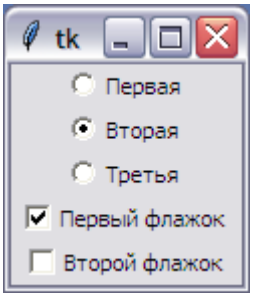

Рис. 6. Флажки

Понажимайте на флажки, убедитесь, что в консоль верно выводится значение ассоциированной переменной при работе с чекбоксом. Сохраним этот модуль, не будем более в нем ничего модифицировать.

7. Списки. Заведем новый python-файл и назовем его List-n-Scale.py. Рассмотрим кратко, как создать выпадающий список элементов. Для этого нам понадобится виджет Listbox. Разместим в документе следующий код:

```
from tkinter import *
root = Tk()

def confirm():       # обработчик события «выбор элемента списка»
    print(lst.curselection())

r = ['Delphi', 'Python', 'C++', 'Java'] # это будет содержание нашего списка

lst = Listbox(root, selectmode=MULTIPLE, height=4) # сам список
                # на 4 строки высотой, можно выбирать несколько пунктов

# заполнение содержимого списка
for i in r:
    lst.insert(END, i)
```

```
lst.pack()

but = Button(text=['Confirm'], command=confirm)
but.pack()

root.mainloop()
```

Поэкспериментируйте с программой: попробуйте изменить размеры списка, сделать одиночный выбор элемента, попробуйте выводить не индексы элементов списка, а сам текст (содержимое).

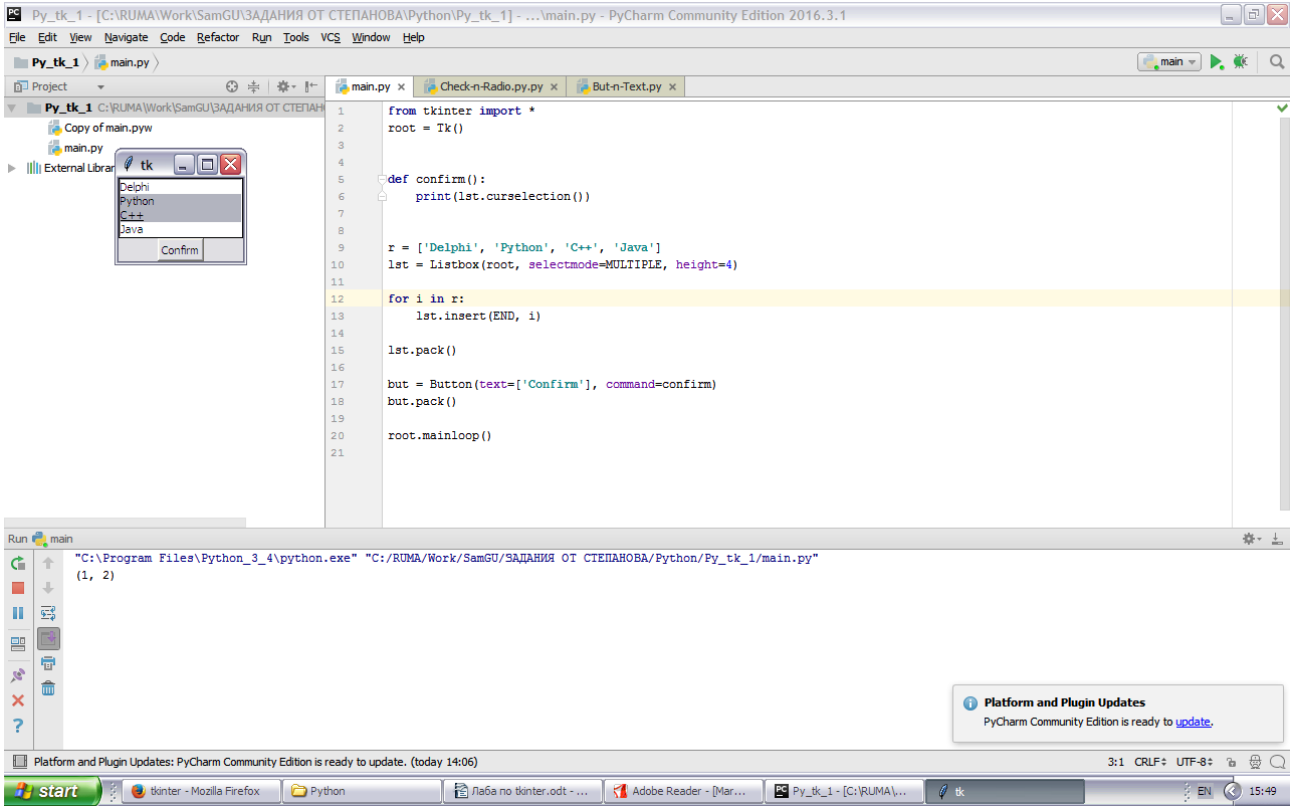

Рис. 7. Список

8. Шкала. Очень часто бывает нужно разместить шкалу, которая, например, управляла бы масштабом какого-либо элемента в окне. Научимся работать с

горизонтальной и вертикальной шкалой. Поместим следующий код в документ:

```
def f1(a):  # это обработчик события для первой шкалы
    print(a)

def f2(b): # обработчик событий для второй шкалы
    print(b)

# теперь займемся шкалами. Сначала горизонтальная, можно управлять ее
размерами
# и ее диапазоном значений, указать интервал делений, разрешение (то число,
# на которое можно дискретно двигать ползунок), назначить обработчик

sca1 = Scale(root, orient=HORIZONTAL, length=300,
            from_=0, to=100, tickinterval=10, resolution=5, command=f1)
# и вертикальная шкала

sca2 = Scale(root, orient=VERTICAL, length=400,
            from_=1, to=2, tickinterval=0.1, resolution=0.1, command=f2)
sca1.pack()
sca2.pack()
```

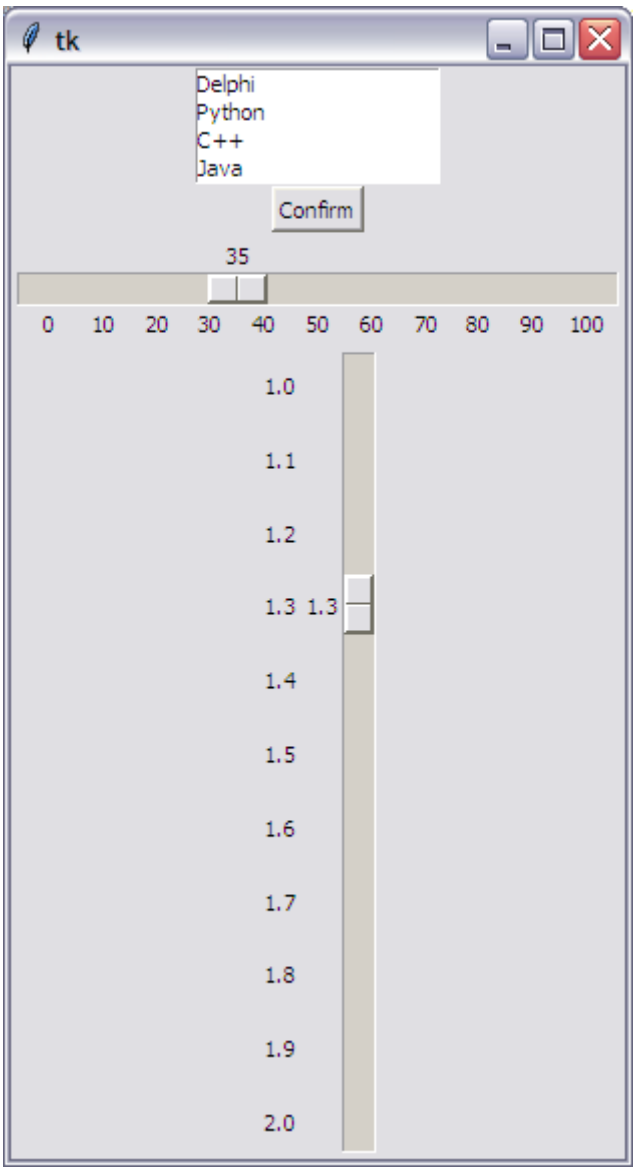

Рис. 8. Горизонтальная и вертикальная шкала

Подвигайте ползунки на шкалах и посмотрите, какие именно значения печатаются в консоли. Почему происходит именно так?

Сохраним этот файл и перейдем к следующему.

9. Скроллинг. Полосы прокрутки, как вы наверное уже обратили внимание, автоматически тут не добавляются. Придется их добавлять вручную. Давайте разбираться, как это сделать. Заведем новый python-скрипт с именем Scroll.py

```
from tkinter import *
```

```
root = Tk()

# заведем уже известную текстовую область. Она была ранее без полос
прокрутки
tx = Text(root, width=15, height=3, font='14')

# а теперь создадим полосу прокрутки, и свяжем 2 виджета вместе
# первым шагом в связке назначим команду скроллу tx.yview

scrV = Scrollbar(root, orient=VERTICAL, command=tx.yview)

# затем сконфигурируем текстовую область и назначим ей только что созданный
# скроллбар в качестве реакции на скроллинг по у
tx.configure(yscrollcommand=scrV.set)

# разместим все элементы в окне
# Здесь фигурирует новый для нас способ размещения — сетка. Для каждого
элемента # можно указать номер столбца и строки, куда его надо поместить
tx.grid(row=0, column=0)
scrV.grid(row=0, column=1)
root.mainloop()
```

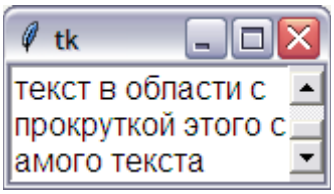

Рис. 10. Полосы прокрутки

Думаю, вы уже догадались, что аналогично можно назначить скроллинг и

другим компонентам, например, тому же выпадающему списку.
Сохраним наш файл, более в него изменения вносить не будем.

11. Рисование на форме. Канва. Чтобы немного порисовать, нам понадобится элемент (виджет) канва. Создать его с настройками по умолчанию максимально просто:

```
canvas = Canvas(root)
```

Чтобы нарисовать линию, воспользуемся методом create_line. Параметрами будут координаты начала и конца линии (или линий — можно одной командой рисовать мультилинии), также можно в качестве опции указать штрихование или цвет линии.

```
from tkinter import *
root = Tk()

canvas = Canvas(root)
canvas.create_line(15, 25, 200, 25)
canvas.create_line(300, 35, 300, 200, dash=(4, 2))
canvas.create_line(55, 85, 155, 85, 105, 180, 55, 85, fill='green')

canvas.pack(fill=BOTH, expand=1)
root.mainloop()
```

Запустите этот фрагмент кода и посмотрите, что именно у вас нарисовалось.

Немного усложним задачу. Нарисуем на канве разноцветные (закрашенные) прямоугольники:

```
canvas.create_rectangle(30, 10, 20, 180,
    outline="black", fill="#fb0")
canvas.create_rectangle(150, 10, 240, 80,
    outline="#f50", fill="#f50")
canvas.create_rectangle(270, 10, 370, 80,
    outline="#05f", fill="#05f")
```

Очевидно, что первые 2 числа в параметрах задают координаты левого верхнего угла прямоугольника, последние 2 - координаты правого нижнего. Цвета «границы» и внутренней заливки могут быть разными и могут задаваться как значениями-константами, так и шестнадцатеричными кодами цветов.

Более сложные фигуры, например, овалы, сектора и закрашенные многоугольники можно рисовать при помощи примитивов create_oval, create_arc и create_polygon.

```
canvas.create_oval(10, 10, 80, 80, outline="gray",
    fill="gray", width=2)
canvas.create_oval(110, 10, 210, 80, outline="gray",
    fill="gray", width=2)

canvas.create_arc(30, 200, 90, 100, start=0,
    extent=210, outline="gray", fill="gray", width=2)

points = [150, 100, 200, 120, 240, 180, 210,
    200, 150, 150, 100, 200]
canvas.create_polygon(points, outline='gray',
    fill='gray', width=2)
```

Ну и, наконец, надпись можно сделать так:

```
canvas.create_text(20, 220, anchor=W, font="Purisa",
   text="Вот такое рисование средствами Python")
```

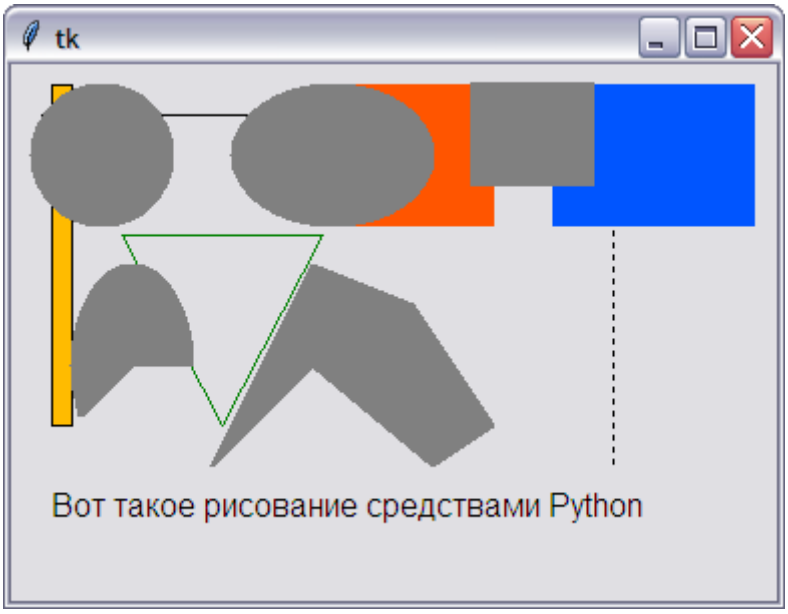

Рис. 11. Рисование средствами Python на канве

Сохраним и этот файл с именем Canvas.py.

Вопросы

1. Как можно изменить значения свойств виджетов? Например, название кнопки, размеры текстового поля и т. д.?
2. Для чего служат ассоциированные переменные? Каких типов они должны быть для радиокнопок, флажков, текстовых полей?
3. Как можно забрать и поместить текст из текстовой области? Текстового поля?
4. Какие вы знаете способы размещения элементов на форме? Чем они отличаются? Как можно позиционировать виджеты в окне?
5. Как добавить скроллинг к выпадающему списку?

6. Самостоятельно изучите, как можно размещать виджеты во фреймах. Ответьте на вопрос, как добавить горизонтальный скроллинг фрейму.
7. Как при помощи шкалы (ползунка) можно управлять размером текста в текстовой области?
8. Самостоятельно изучите способы создания и работы с диалоговыми окнами. Продемонстрируйте работу, например, с диалоговым окном выбора цвета.

Задания по вариантам.

1. На форме расположены три флажка и кнопка. Флажки задают red, green, blue компоненты цвета (установите их равными 255). Щелчок по кнопке устанавливает цвет кнопки в соответствии с выбранными флажками компонентами цвета.
2. Поместите на форму флажки и кнопку. Нажатие на кнопку включает все флажки. Повторное нажатие на кнопку снимает все флажки. Рядом с флажками располагается текст, нажатие на текст или флажок должно также включать соответствующий флажок.
3. На форму поместите строку, кнопку и список с размерами шрифта. Выбор размера шрифта в списке меняет на таковой размер текста в строке. Нажатие на кнопку убирает одну букву из начала строки и добавляет ее в конец строки, цвет текста строки при этом меняется случайным образом.
4. На форму поместите флажки и текстовые поля. Например, флажки будут отмечать список экзаменов в зимнюю сессию, а текстовые поля - оценки за экзамены. Поставьте еще один флажок "Сессия сдана". Как только вы проставите все оценки - автоматически появится галочка для флажка "Сессия сдана" и появится окошко с сообщением об итогах сессии (отлично, хорошо, удовлетворительно или неудовлетворительно). Если хотя бы по одному экзамену выставлена неудовлетворительная оценка - флажок "Сессия сдана" очищается и флажком отмечаем экзамен, сданный

на 2.

5. На форме располагается текстовая область, текстовое поле, кнопка и радоигруппа из 2 радиокнопок. При активной одной радиокнопке в поле вводится текст, после нажатия на кнопку он оказывается в текстовой области. В текстовую область непосредственно ничего впечатать нельзя, при попытке это сделать должно выдаваться сообщение-предупреждение. При активной другой радиокнопке в текстовую область вводится текст, после нажатия на кнопку он оказывается в текстовом поле. В текстовое поле непосредственно ничего впечатать нельзя, при попытке сделать это должо появляться предупреждение.
6. Поместите на форму флажки и 4 кнопки. Нажатие на первую кнопку поднимает все флажки, нажатие на вторую – снимает все флажки. Нажатие на 3-ю кнопку позволяет поставить все флажки, если они сняты, и снять все флажки, если они подняты. Нажатие на 4 кнопку делает все флажки неактивными, повторное нажатие на эту кнопку делает флажки активными.
7. Поместите на форму флажок, радиогруппу, список, и кнопку. Щелчок мышкой по тексту рядом с флажком должен включать-выключать его. Значения из выпадающего списка должны включать соответствующие радиокнопки. Щелчок по кнопке меняет местами надписи на кнопке и рядом с флажком.
8. Поместите на форму флажки и примечание, что выбрать можно ограниченное количество пунктов. Попытка выбора большего числа флажков вызывает предупреждающее сообщение. При этом лишние флажки снимаются. Повторная попытка выбрать лишний флажок должна привести к появлению метки-предупреждения, которая будет "убегать" от курсора мыши при попытке сделать по ней клик, не выходя за пределы окна.
9. Поместите на форму флажки, несколько радиокнопок, несколько

текстовых полей и кнопку. Флажки можно отмечать, а по нажатию на кнопку должно выводиться сообщение о количестве выбранных флажков, радиокнопок и заполненных текстовых полей. После этого заполненные поля становятся неактивными, поднятые флажки и радиокнопки становятся неактивными.

10. Заполняем регистрационную карточку (например, на конференцию). Необходимо отметить имя, фамилию, университет (из списка), должность (из списка), страну (последнюю можно выбрать из выпадающего списка) . Данные надо продублировать. Для этого поместите на форму все необходимые элементы и кнопку. Заполняется только одна карточка, нажатием на кнопку данные копируются во вторую карточку. Если какое-либо поле не заполнено, либо заполнено неверно (например, одни цифры в фамилии), об этом должно быть выдано сообщение и предложено данное поле заполнить еще раз. Приветствуется возможность сохранить данные в файл.
11. Поместите на форму радиокнопки и список. Выбор радиокнопки формирует значения в списке. Например, радионопки могут указывать страны, а в выпадающем списке будут присутствовать международные аэропорты данных стран.
12. Поместите на форму несколько радиокнопок с обозначением цветов и выпадающий список с обозначением цветов. Выбор радиокнопки делает этот цвет цветом фона формы, выбор элемента в списке делает цвет цветом текста на форме.
13. Установите на форму кнопку и список. Выбранное значение из списка должно отображаться на кнопке. Щелчок по кнопке запускает бегущую строку на кнопке. Повторный щелчок останавливает бегущую строку. Бегущая строка должна менять цвет в процессе своего движения.
14. Делаем «планировщик заданий». Поместите на форму компонент (или компоненты) для выборы даты, компоненты (возможно, списки) для

выбора часов и минут, текстовое поле, кнопку, текстовую область. В текстовое поле записываем задание, выбираем для него дату, устанавливаем время. Щелчок по кнопке добавляет выбранные данные в текстовую область.

15. Поместите на форму компонент (или компоненты) для выбора даты, текстовую область, метку и кнопку. После выбора даты в метку выводится, какой это день недели, а в текстовой области появляется расписание на этот день. Щелчок по кнопке очищает текстовую область, значение метки, выбранную дату.