Python

МОДУЛЬ tkinter

Графический интерфейс

Многие программы на сегодняшний день используют графический интерфейс, который более интуитивен и удобен для пользователя, чем консоль.

Модуль **Tkinter**, предназначен для работы с компонентами графического интерфейса пользователя (graphical user interface – **GUI**).

Tkinter доступен в виде отдельного встроенного модуля, который содержит все необходимые графические компоненты – кнопки, текстовые поля и т.д.

Как и любой модуль, tkinter в Python можно импортировать:

from tkinter import *

Создание окна

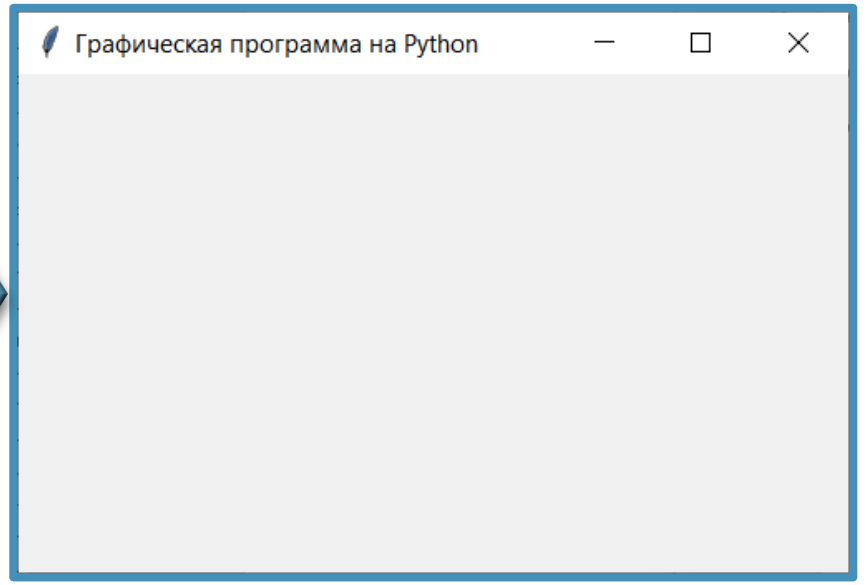
Базовым моментом в построении графических программ является создание окна. Затем в окно добавляются все остальные компоненты графического интерфейса.

root = Tk()	Для создания графического окна применяется конструктор Tk(). Переменную, связываемую с объектом, часто называют root (корень).
root.title(‘Пример 1’**)**	Устанавливается заголовок окна.
root.geometry(‘600x400’**)**	Устанавливается размер окна (передается строка в формате "Ширина x Высота").
root.geometry(‘400x300+100+100’**)**	Устанавливается размер окна и его положение относительно левого верхнего угла экрана (передается строка в формате "Ширина x Высота + координатаX + координатаY").
root.mainloop()	Запускает цикл обработки событий окна для взаимодействия с пользователем. Данная строчка кода должна быть всегда в конце скрипта!

Создание окна

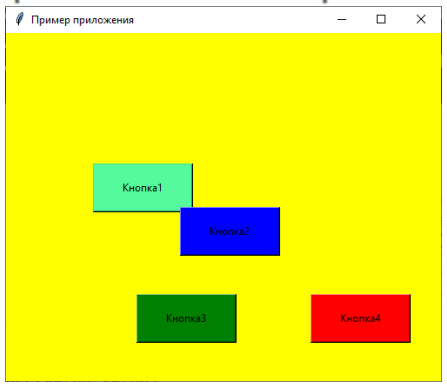
Для создания простейшего окна надо написать следующий скрипт. В результате при запуске скрипта мы увидим пустое окошко.

```
1 from tkinter import*
2 root = Tk()
3 root.title('Графическая программа на Python')
4 root.geometry('500x300+300+250')
5 root.mainloop()
```



Создание и размещение кнопок

```
1  from tkinter import *
2  # Создаем окно приложения
3  root=Tk()
4  root.title('Пример приложения')
5  root.geometry('500x400+300+200') # ширина=500, высота=400, x=300, y=200
6  root.configure(background='yellow')# задаем цвет фона
7  btn1 = Button(text="Кнопка1", width=15, height=3,bg="#54FA9B")
8  btn2 = Button(text="Кнопка2", width=15, height=3,bg="blue")
9  btn3 = Button(text="Кнопка3", width=15, height=3,bg="green")
10 btn4 = Button(text="Кнопка4", width=15, height=3,bg="red")
11 # Метод place(x=a,y=b) попробовать варианты значений параметров x, y,
12 # учитываем размеры окна
13 btn1.place(x=100,y=150)
14 btn2.place(x=200,y=200)
15 btn3.place(x=150,y=300)
16 btn4.place(x=350,y=300)
17 root.mainloop()
```



Наберите программу. Задайте свои размеры окна и размеры и расположение 16 кнопок. Подберите для них цвета в шестнадцатеричном коде.

Объекты класса Canvas

Для рисования простейших рисунков создаются **объекты-холсты**, на которых можно "рисовать", размещая различные фигуры и объекты.

При создании экземпляра **Canvas** необходимо указать к какому окну он относится, его ширину и высоту:

```
c = Canvas(root, width=200, height=200, bg='white')
```

c – это имя (любой идентификатор) холста, под которым он в дальнейшем используется.

Чтобы объект был отображен в окне, надо использовать метод **pack()**. Если не вставить эту строчку кода, то объект в окне так и ***не появится***, хотя он есть в программе.

```
c.pack()
```

Управление цветом

Цвет в формате RGB

" #54FA9B "

Red
00...FF

Green
0...FF

Blue
00... FF

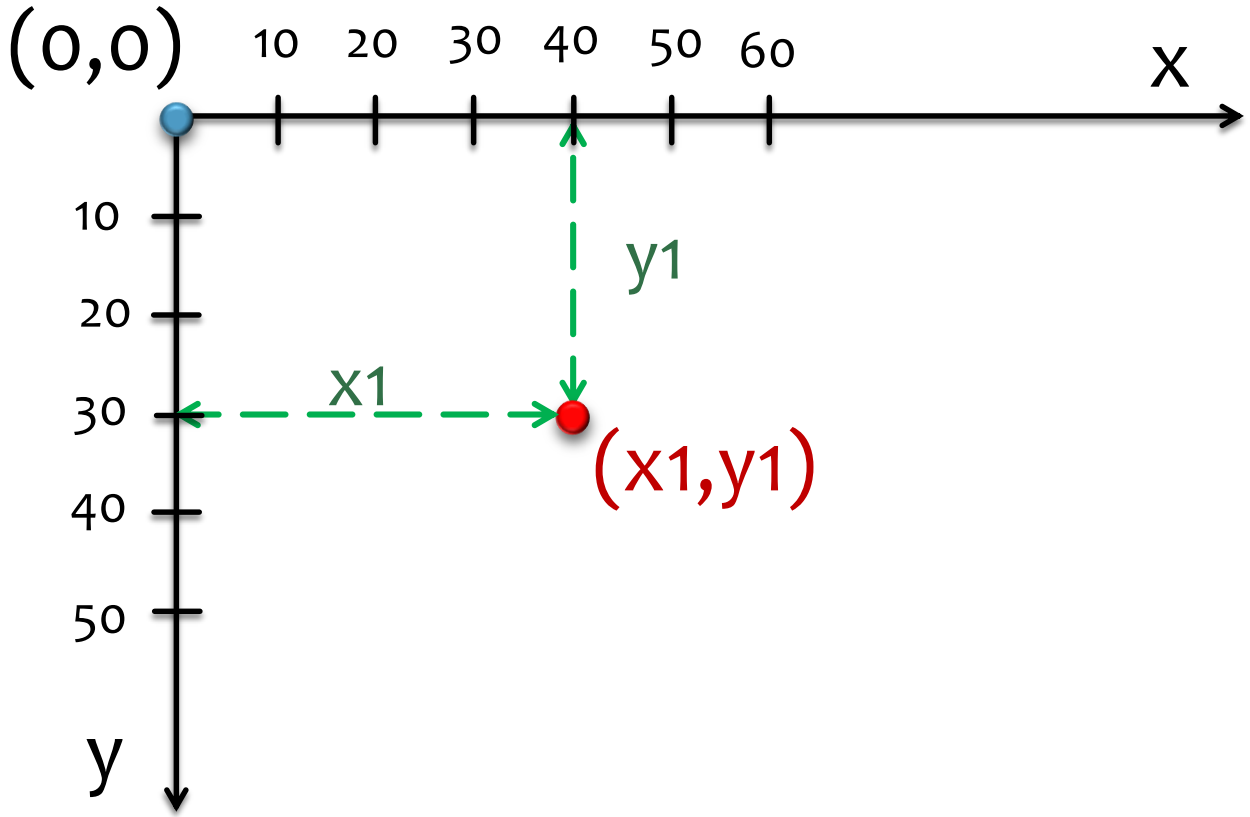
Цвет по названию

"red"

white, black, gray, navy, blue, cyan, green, yellow, red, orange, brown, maroon, violet, purple, ...

Создание простейших рисунков

При размещении геометрических примитивов и других объектов указываются их координаты на холсте в пикселах. Точкой отсчета является верхний левый угол (0,0).

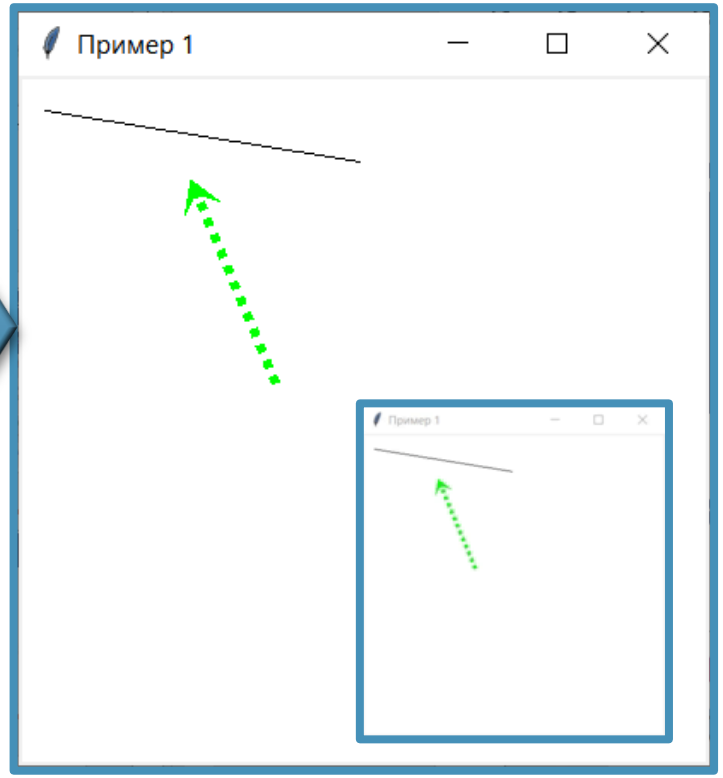


На холсте с помощью метода create рисуются различные объекты:

c.create_line(x,y,x1,y1)	Рисует линию между точками (x,y) и (x1,y1) на полотне c.
c.create_polygon(x,y, ... ,xn,yn)	Рисует ломанную по точкам (x,y) ... (xn,yn) на полотне c.
c.create_oval(x,y,x1,y1)	Рисует овал в границах прямоугольника с верхней левой точкой (x,y) и нижней правой (x1,y1) на полотне c.
c.create_rectangle(x,y,x1,y1)	Рисует прямоугольник с верхней левой точкой (x,y) и нижней правой (x1,y1) на полотне c.
c.create_text(x,y,x1,y1)	Рисует текст, центрируя его по точке (x,y) и (x1,y1), на полотне c.

Создание различных линий методом c.create_line()

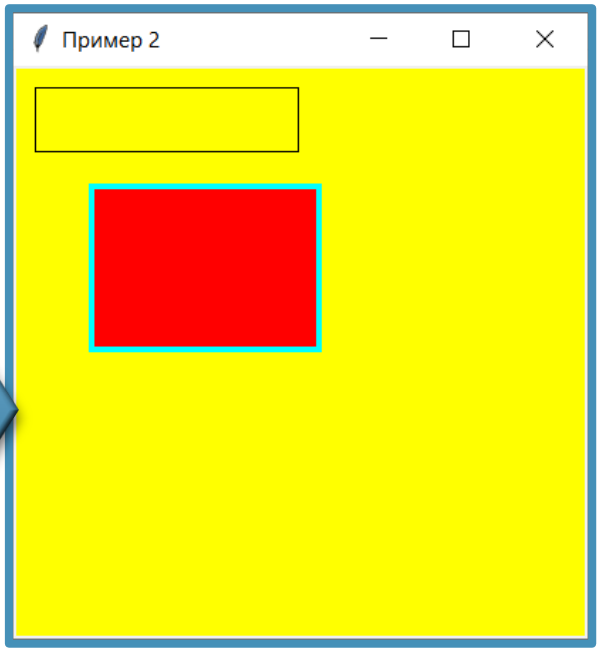
```
1  from tkinter import*
2  root = Tk()
3  root.title('Пример 1')
4  c = Canvas(root, width=400, height=400, bg='white')
5  c.pack()
6  c.create_line(15,20,200,50)
7  c.create_line(150,180,100,60, fill='lime',
8                width=5, arrow=LAST, dash=(10,2),
9                activefill='green',
10               arrowshape='10 20 10')
11 root.mainloop()
```



fill – цвет линии, arrow – стрелка (**FIRST**, **BOTH**, **LAST**),
dash – пунктир,
activefill определяет цвет отрезка при наведении на него курсора мыши,
arrowshape – вид стрелки.

Создание прямоугольников методом create_rectangle()

```
1 from tkinter import*
2 root = Tk()
3 root.title('Пример 2')
4 c= Canvas(root, width=400, height=400, bg='yellow')
5 c.pack()
6 c.create_rectangle(15,15,200,60)
7 c.create_rectangle(55,85,215,200, fill='red', outline='cyan',
8                     width=4, activedash=(5,4))
9 root.mainloop()
```



Первые координаты – верхний левый угол, вторые – правый нижний. В приведенном примере, когда на второй прямоугольник попадает курсор мыши, его рамка становится пунктирной, что определяется свойством **activedash.**

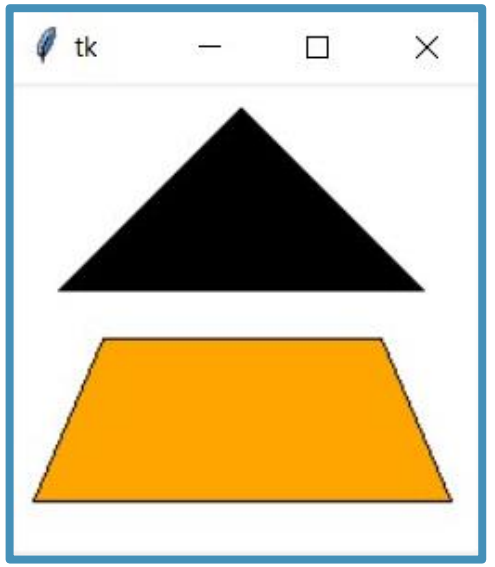
Создание многоугольников методом create_polygon()

Методом **create_polygon()** рисуется произвольный многоугольник путем задания координат каждой его точки.

Для удобства координаты точек можно заключать в скобки:

```
c.create_polygon((40, 110), (160, 110), (190, 180), (10, 180), fill='orange', outline='black')
```

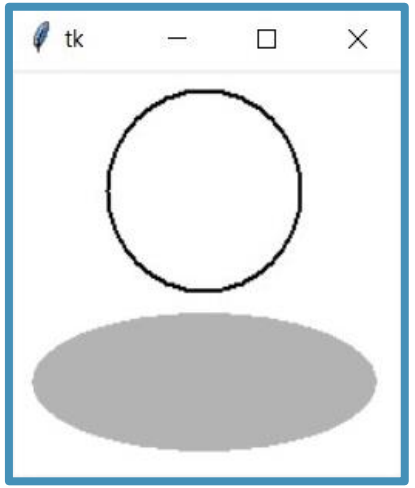
```
from tkinter import *
root = Tk()
c = Canvas(root, width=200, height=200, bg='white')
c.pack()
c.create_polygon(100, 10, 20, 90, 180, 90)
c.create_polygon(40, 110, 160, 110, 190, 180, 10, 180,
                 fill='orange', outline='black')
root.mainloop()
```



Создание овалов и кругов методом create_oval()

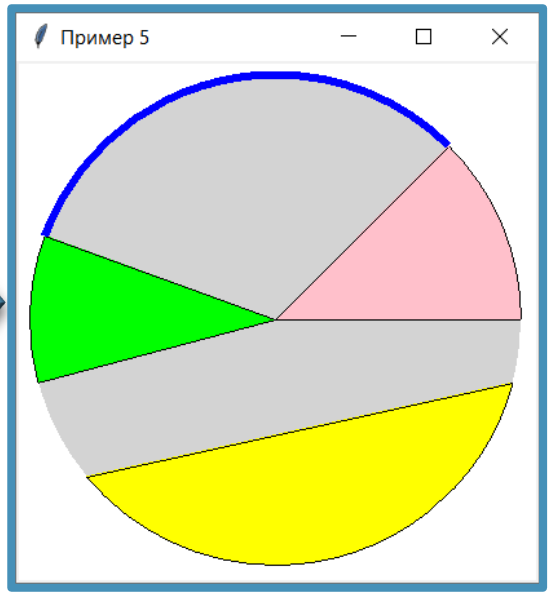
Метод **create_oval()** создает эллипсы. При этом задаются координаты гипотетического прямоугольника, описывающего эллипс. Если нужно получить круг, то соответственно описываемый прямоугольник должен быть квадратом.

```
from tkinter import *
root = Tk()
c = Canvas(root, width=200, height=200, bg='white')
c.pack()
c.create_oval(50, 10, 150, 110, width=2)
c.create_oval(10, 120, 190, 190, fill='grey70', outline='white')
root.mainloop()
```



Создание фигур методом create_arc()

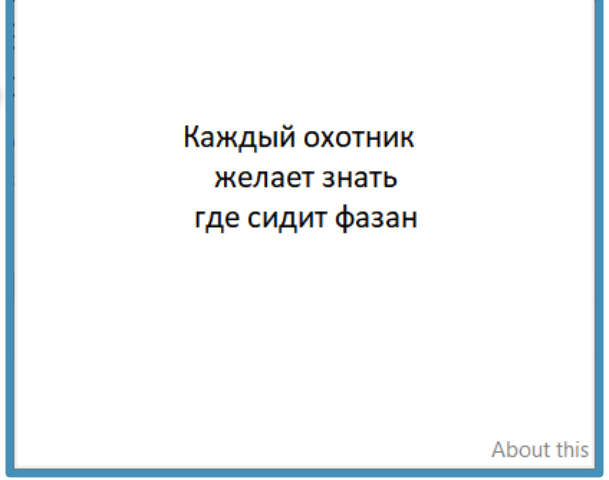
```
1  from tkinter import*
2  root = Tk()
3  root.title('Пример 5')
4  c= Canvas(root, width=400, height=400, bg='white')
5  c.pack()
6  c.create_oval(10,10,390,390, fill='lightgrey', outline='white')
7  c.create_arc(10,10,390,390, start=0, extent=45, fill='pink')
8  c.create_arc(10,10,390,390, start=160, extent=35, fill='lime')
9  c.create_arc(10,10,390,390, start=220, extent=125,
10                 style=CHORD, fill='yellow')
11 c.create_arc(10,10,390,390, start=160, extent=-115,
12                 style=ARC, outline='blue', width=6)
13 root.mainloop()
```



start	градус начала фигуры
extent	угол поворота
style	сектор (по умолчанию), сегмент (CHORD), дуга (ARC)

Размещение текста на форме методом create_text()

```
from tkinter import*
root = Tk()
root.title('Пример 6')
c= Canvas(root, width=400, height=400, bg='white')
c.pack()
c.create_text(200,200, text='Каждый охотник \n желает
                justify=CENTER, font='Calibri 14')
c.create_text(400,400, text='About this',
                anchor=SE, fill='grey')
```



координаты	в заданной точке располагается центр текстовой надписи
anchor	якорь (привязка) со значениями N, NE, E, SE, S, SW, W, NW (стороны света)
justify	выравнивание текста относительно себя самого

Выполните задание