Лекции. Практические занятия

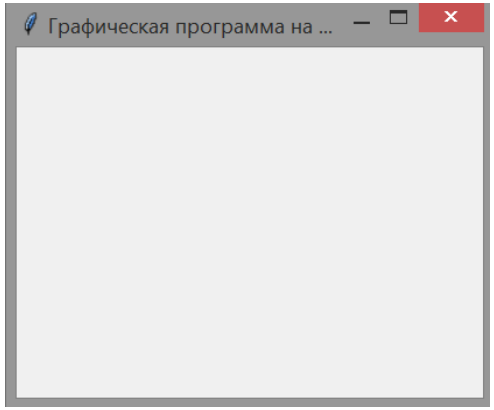
Солдатов Е.Ю.

2024 г.

НЕМНОГО ГРАФИКИ

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

В Python также можно создавать графические программы. Для этого используется модуль **tkinter**.



Сначала создадим окно:

```
import tkinter
window = tkinter.Tk()
window.title("Графическая программа на Python")
window.geometry("320x240")
window.mainloop()
```

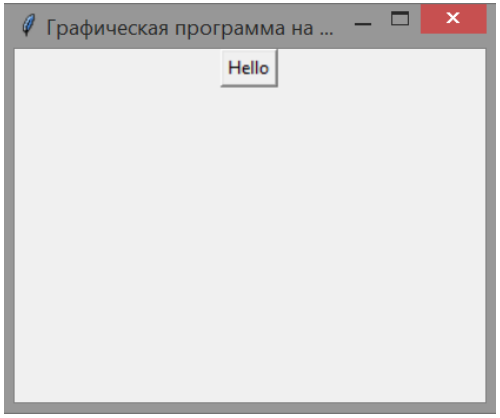
Метод *title()* – заголовок окна, *geometry()* – размеры окна.
Чтобы вывести окно, используется метод *mainloop()*.

Если мы хотим сдвинуть начальную позицию окна, то можно использовать метод **geometry()** так:

```
window.geometry("320x480+300+200")
```

Теперь строка в методе geometry имеет следующий формат: "Ширина x Высота + координатаX + координатаY".
То есть при запуске окно будет находиться на 300 пикселей вправо и на 200 пикселей вниз от верхнего левого угла экрана.

НЕМНОГО ГРАФИКИ



Теперь добавим кнопку:

```
btn = tkinter.Button(text="Hello")
btn.pack()
```

Метод **pack()** делает её видимой.

Использован только атрибут **text.**

Попробуем другие атрибуты:

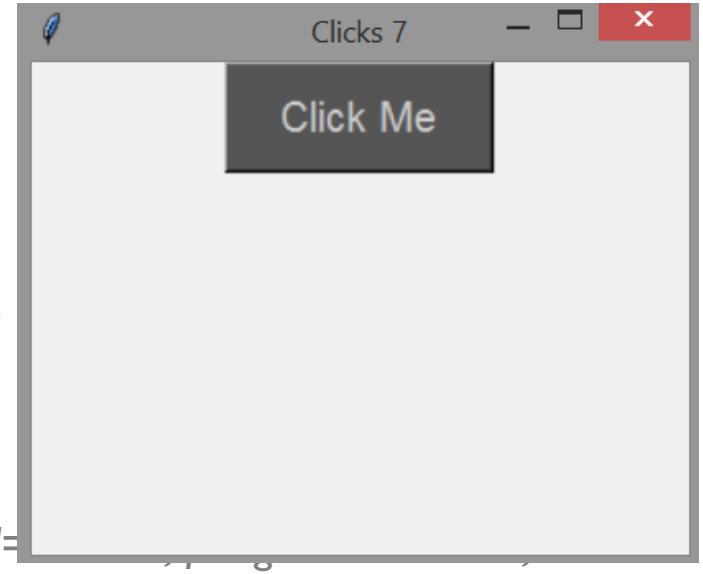
```
btn = tkinter. Button(text="Hello",          # текст кнопки
        background="#555",     # фоновый цвет кнопки
        foreground="#ccc",     # цвет текста
        padx="20",             # отступ от границ до содержимого по
горизонтали
        pady="8",              # отступ от границ до содержимого по вертикали
        font="16"              # высота шрифта
        )
```

Для обработки нажатия на кнопку необходимо установить в конструкторе параметр command, присвоив ему ссылку на функцию, которая будет срабатывать при нажатии:

```
def click_button():
    pass
...
btn = Button(text="Click Me", background="#555", foreground="#ccc",
            padx="20", pady="8", font="16", command=click_button)
```

НЕМНОГО ГРАФИКИ

```
import tkinter
num_of_clicks = 0

def click_button():
    global num_of_clicks
    num_of_clicks += 1
    window.title("Clicks {}".format(num_of_clicks))
window = tkinter.Tk()
window.title("Графика на Python")
window.geometry("320x240+300+200")
btn = tkinter.Button(text="Click Me", background=
            padx="20", pady="8", font="16", command=click_button)
btn.pack()
window.mainloop()
```



Можно менять и атрибуты кнопки, для этого используется метод **set()** в функции *click_button()*:

```
buttonText.set("Clicks {}".format(clicks))
```

В атрибутах кнопки в основной программе тогда должно быть *text=buttonText* (инициализация: *buttonText = tkinter.StringVar()*)

Есть и другой метод переопределения:

```
btn.config(text="Clicks {}".format(clicks))
```

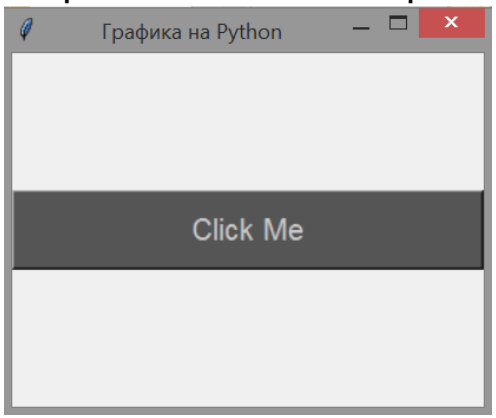
НЕМНОГО ГРАФИКИ

Самый простой способ позиционирования элементов представляет вызов у элемента метода ***pack()***. Этот метод принимает следующие параметры:

expand: если равно True, то виджет заполняет все пространство контейнера.

fill: определяет, будет ли виджет растягиваться, чтобы заполнить свободное пространство вокруг. Значения: NONE (по умолчанию, элемент не растягивается), **x** (растягивание по горизонтали), **y** (растягивание по вертикали) и **both** (растягивается и по X и по Y).

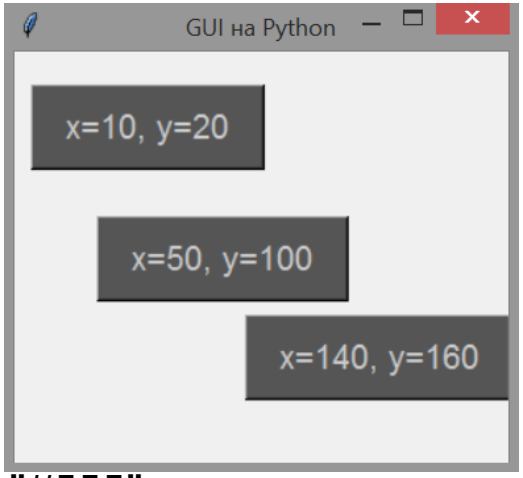
side: выравнивает виджет по одной из сторон контейнера. Значения: TOP (по умолчанию, выравнивается по верху), BOTTOM (выравнивание по низу), LEFT (выравнивание по левой стороне), RIGHT (выравнивание по правой стороне).




```
import tkinter
window = tkinter.Tk()
buttonText = tkinter.StringVar()
window.title("Графика на Python")
window.geometry("320x240+300+200")
btn = tkinter.Button(text="Click Me", background="#555", foreground="#ccc",
            padx="20", pady="8", font="16")
btn.pack(expand = True, fill="x")
btn.pack()
window.mainloop()
```

НЕМНОГО ГРАФИКИ

Метод **place()** позволяет более точно настроить параметры позиционирования.



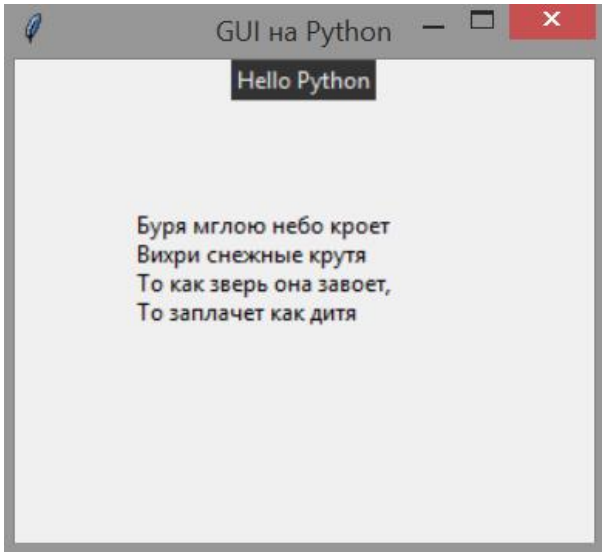
Пример:

```
import tkinter

window = tkinter.Tk()
window.title("GUI на Python")
window.geometry("300x250")

btn1 = tkinter.Button(text="x=10, y=20", background="#555",
foreground="#ccc", padx="14", pady="7", font="13")
btn1.place(x=10, y=20)

btn2 = tkinter.Button(text="x=50, y=100", background="#555",
foreground="#ccc", padx="14", pady="7", font="13")
btn2.place(x=50, y=100)

btn3 = tkinter.Button(text="x=140, y=160", background="#555",
foreground="#ccc", padx="14", pady="7", font="13")
btn3.place(x=140, y=160)

window.mainloop()
```

НЕМНОГО ГРАФИКИ

Текстовые метки в Python представлены элементом Label. Этот элемент позволяет выводить статический текст без возможности редактирования.

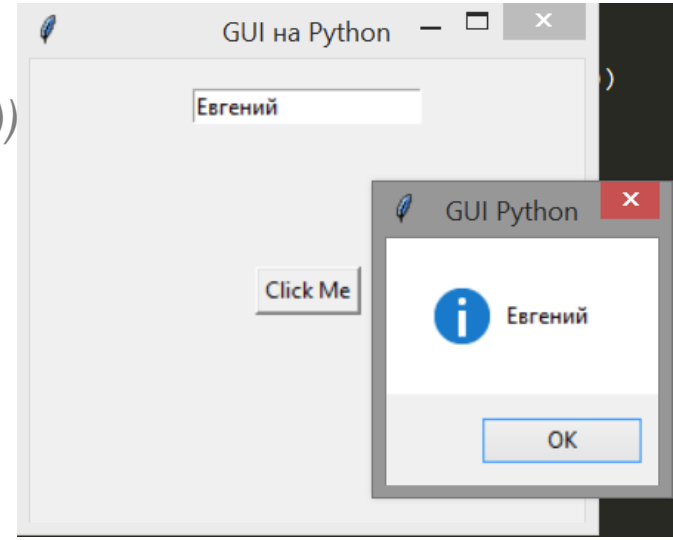
```
import tkinter

window = tkinter.Tk()
window.title("GUI на Python")
window.geometry("300x250")

label1 = tkinter.Label(text="Hello Python", fg="#eee", bg="#333")
label1.pack()

poetry = "Буря мглою небо кроет\nВихри снежные крутя\nТо как зверь она завоет,\nТо заплачет как дитя"
label2 = tkinter.Label(text=poetry, justify="left")
label2.place(relx=.2, rely=.3)

window.mainloop()
```



НЕМНОГО ГРАФИКИ

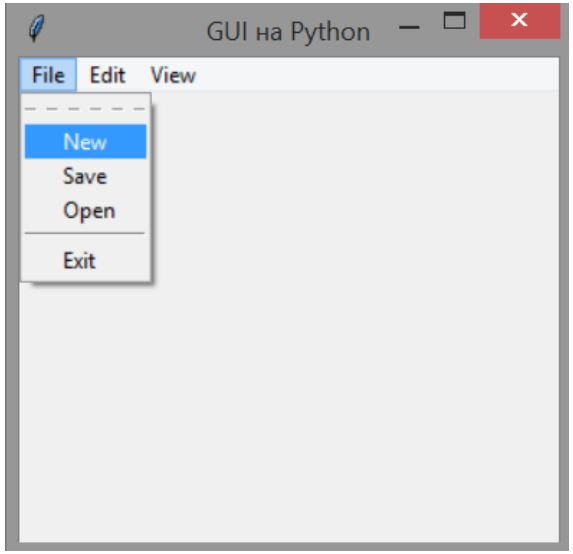
Элемент Entry представляет поле для ввода текста.

```
import tkinter
from tkinter import messagebox

def show_message():
    messagebox.showinfo("GUI Python", message.get())

window = tkinter.Tk()
window.title("GUI на Python")
window.geometry("300x250")

message = tkinter.StringVar()

message_entry = tkinter.Entry(textvariable=message)
message_entry.place(relx=.5, rely=.1, anchor="c")

message_button = tkinter.Button(text="Click Me", command=show_message)
message_button.place(relx=.5, rely=.5, anchor="c")

window.mainloop()
```

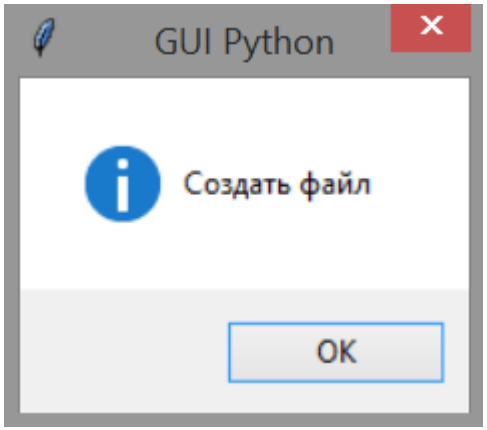


НЕМНОГО ГРАФИКИ

Для создания иерархического меню в tkinter и Python применяется виджет **Menu**.




```
import tkinter
from tkinter import messagebox
def show_message():
    messagebox.showinfo("GUI Python", "Создать файл")
window = tkinter.Tk()
window.title("GUI на Python")
window.geometry("300x250")
main_menu = tkinter.Menu()
file_menu = tkinter.Menu()
file_menu.add_command(label="New", command=show_message)
file_menu.add_command(label="Save")
file_menu.add_command(label="Open")
file_menu.add_separator()
file_menu.add_command(label="Exit")
main_menu.add_cascade(label="File", menu=file_menu)
main_menu.add_cascade(label="Edit")
main_menu.add_cascade(label="View")
window.config(menu=main_menu)
window.mainloop()
```



ПРИМЕР СКРИПТА НА PyROOT

Настройки и инструкции по установке для **pyROOT**: https://root.cern.ch/pyroot
Попробуем предложенный скрипт:

```
from ROOT import gROOT, TCanvas, TF1
gROOT.Reset()
c1 = TCanvas( 'c1', 'Example with Formula', 200, 10, 700, 500 )
fun1 = TF1( 'fun1', 'abs(sin(x)/x)', 0, 10 )
c1.SetGridx()
c1.SetGridy()
fun1.Draw()
c1.Update()
```