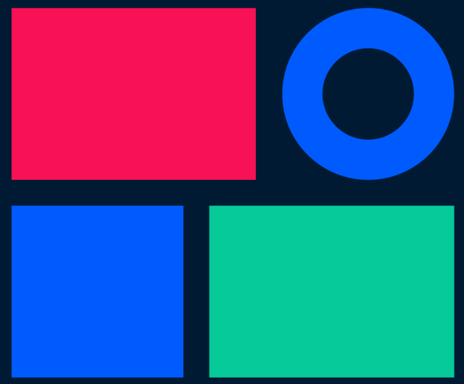


Ozon
New Skills

Ozon
New Skills

Python 18+

Tkinter && OOP/ Практика

GUI

Графический интерфейс пользователя, графический пользовательский интерфейс — система средств для взаимодействия пользователя с компьютером, основанная на представлении всех доступных пользователю системных объектов и функций в виде графических компонентов экрана.

GUI

Что такое GUI?

GUI

Под графическим интерфейсом пользователя (GUI) подразумеваются все те окна, кнопки, текстовые поля для ввода, скроллеры, списки, радиокнопки, флажки и др., которые вы видите на экране, открывая то или иное приложение.

Создание Gui

Для создание Gui в Python в основном используются две библиотеки — PyQT(4 и 5 версии), и встроенная библиотека Tkinter. О ней дальше и пойдет речь

Tkinter

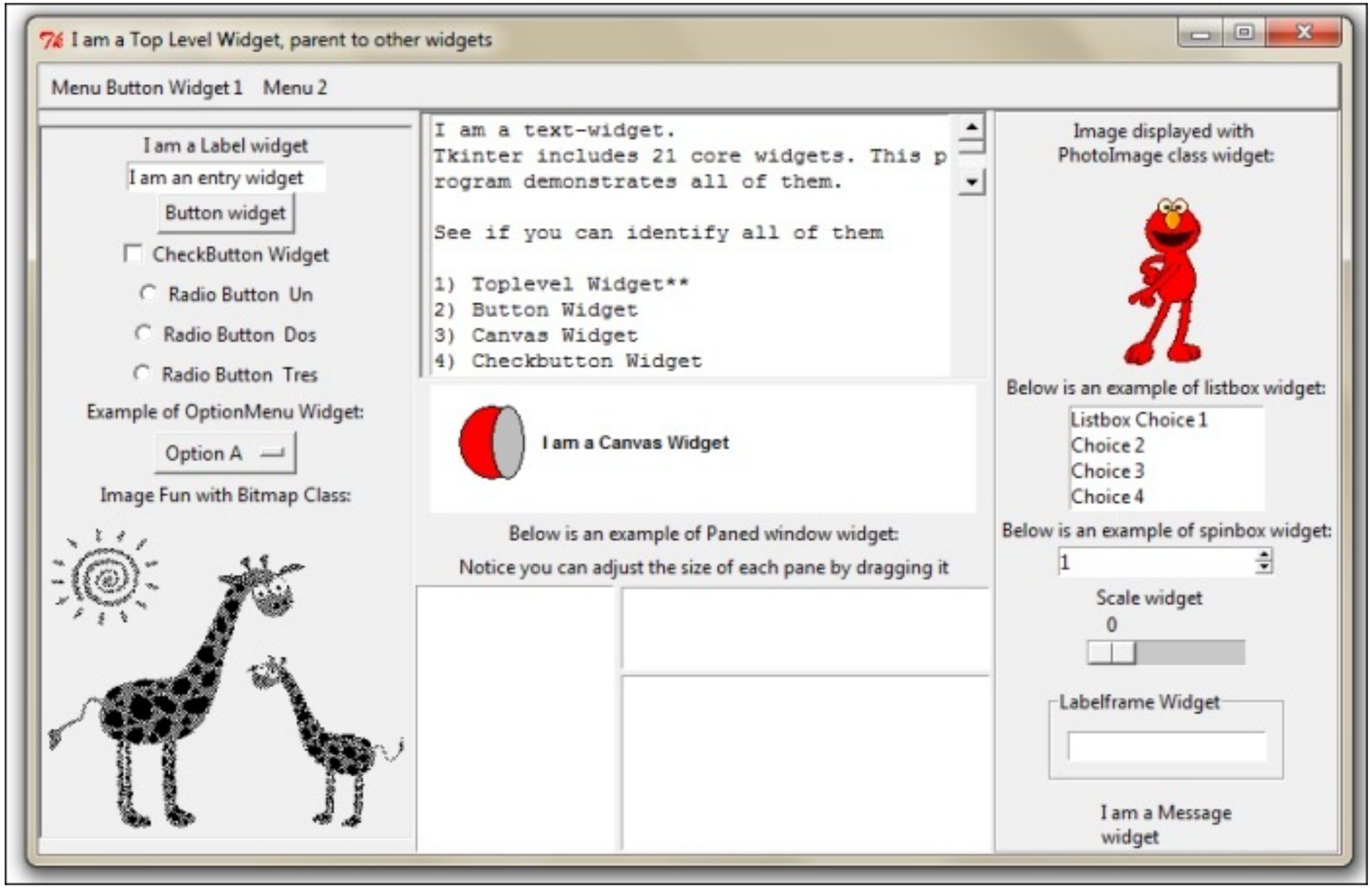
Tkinter — кросс-платформенная событийно-ориентированная графическая библиотека на основе средств Tk, написанная Стином Лумхольтом и Гвидо ван Россумом

Алгоритм создания приложения

Существует следующий порядок создания приложений с GUI:

1. Создать главное окно.
2. Создать виджеты и выполнить конфигурацию их свойств (опций).
3. Определить события, то есть то, на что будет реагировать программа.
4. Описать обработчики событий, то есть то, как будет реагировать программа.
5. Расположить виджеты в главном окне.
6. Запустить цикл обработки событий.

нет, он
не всегда
такой
страшный



Tkinter

Виджеты и возможности упаковки геометрии

root

Для создания базового окна вам потребуется вызов объекта верхнего уровня — **tk()**. Уже в него будут «насаживаться» все элементы, понимая его как главный объект окна.

Вариант создания приложения

```
from tkinter import *
root = Tk()
data = ['И что ты тут кликаешь', 'Тебе не с кем поговорить?',
'здесь могла быть ваша реклама']
i = 0
def change(event):
        global i
        if i == len(data):
                i = 0
        b['text'] = data[i]
        b['activeforeground'] = "red"
        i= i + 1


b = Button(text='Кликни для продолжения', width=50, height=15, font = ('Verdana', 30))
b.bind('<Button-1>', change)
b.pack()
root.mainloop()
```

Самые распространённые виджеты

Команды	Их назначение
`Label`	Для текста
`Button`	Просто кнопка
`Checkbutton`	Для подтверждения чего-либо
`Entry`	Поле пользовательского ввода
`Combobox`	Поле с выпадающих списком

Варианты верстки

Однако для того, чтобы ваши виджеты располагались в программе согласованно, их нужно как —то сверстать. Существуют два виды верстки в Tkinter (там правда они называются **упаковщики геометрии**)

Виды верстки

Команды	Их назначение
`a.pack(side=g,fill=c)`	side==LEFT (состыковывает виджет к предыдущему к его левому краю или в плотную к левому краю окна) side==RIGHT (состыковывает виджет к правому левому краю или) side==TOP (состыковывает виджет к нижнему краю прошлого виджета либо верхнему краю окна) side==BOTTOM (состыковывает виджет к верхнему краю прошлого виджета либо нижнему краю окна) side==MIDDLE (помещает виджет в центр окна)
`a.grid(row=c,column=m)`	Размещает виджет в колонке с координатами row (строка) и column (колонка)
`a.place(x=n,y=m)`	Размещает виджет так, что его левый правый угол будет в координатах, значений которых берутся из параметров x и y.

Диалоговые окна

С помощью `tkinter.messagebox` можно создавать диалоговые окна. Для этого нужно отдельно его импортировать:

```
import tkinter.messagebox
```

Работа с файлами

Команды	Их назначение
askopenfilename	skopenfilename - класс окна открытия файла. Общий синтаксис создания окна: **name = askopenfilename()**
asksavefilename	asksavefilename - класс окна сохранения файла. Общий синтаксис создания окна: **name = asksavefilename()**

Блокнот

Используя все полученные знания, вы можете попробовать создать свой клон блокнота! Для него вам придется использовать ваши знания по OOP и реализовать ваши познания в tkinter