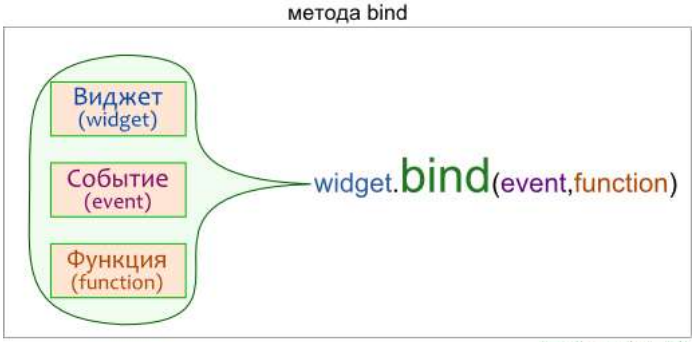
Урок 7. Метод bind()

В tkinter с помощью метода bind() между собой связываются виджет, событие и действие. Например, виджет – кнопка, событие – клик по ней левой кнопкой мыши, действие – отправка сообщения. Другой пример: виджет – текстовое поле, событие – нажатие Enter, действие – получение текста из поля методом get() для последующей обработки программой. Действие оформляют как функцию (или метод), которая вызываются при наступлении события.

Добавление функциональности графическому элементу с помощью метода bind



Один и тот же виджет можно связать с несколькими событиями. В примере ниже используется одна и та же функция-обработчик, однако могут быть и разные:

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

def change(event):
    b['fg'] = "red"
    b['activeforeground'] = "red"

b = Button(text='RED', width=10, height=3)
b.bind('<Button-1>', change)
b.bind('<Return>', change)

b.pack()

root.mainloop()
```

Здесь цвет текста на кнопке меняется как при клике по ней (событие <Button-1>), так и при нажатии клавиши Enter (событие <Return>). Однако Enter сработает, только если кнопка предварительно получила фокус. В данном случае для этого надо один раз нажать клавишу Tab. Иначе нажатие Enter будет относиться к окну, но не к кнопке.

У функций-обработчиков, которые вызываются через bind(), а не через свойство command, должен быть обязательный параметр event, через который передается событие. Имя event – соглашение, идентификатор может иметь другое имя, но обязательно должен стоять на первом месте в функции, или может быть вторым в методе:

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

class RedButton:
    def __init__(self):
        self.b = Button(text='RED', width=10, height=3)
```

```
        self.b.bind('<Button-1>', self.change)
        self.b.pack()
    def change(self, event):
        self.b['fg'] = "red"
        self.b['activeforeground'] = "red"

RedButton()
root.mainloop()
```

Что делать, если в функцию надо передать дополнительные аргументы? Например, клик левой кнопкой мыши по метке устанавливает для нее один шрифт, а клик правой кнопкой мыши – другой. Можно написать две разные функции:

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

def font1(event):
    l['font'] = "Verdana"
def font2(event):
    l['font'] = "Times"

l = Label(text="Hello World")

l.bind('<Button-1>', font1) # ЛКМ
l.bind('<Button-3>', font2) # ПКМ
l.pack()

root.mainloop()
```

Но это не совсем правильно, так как код тела функций фактически идентичен, а имя шрифта можно передавать как аргумент. Лучше определить одну функцию:

```
…
def changeFont(event, font):
    l['font'] = font
…
```

Однако возникает проблема, как передать дополнительный аргумент функции в метод bind()? Ведь в этот метод мы передаем объект-функцию, но не вызываем ее. Нельзя написать `l.bind('<Button-1>', changeFont(event, "Verdana"))`. Потому что как только вы поставили после имени функции скобки, то значит вызвали ее, то есть заставили тело функции выполниться. Если в функции нет оператора return, то она возвращает None. Поэтому получается, что даже если правильно передать аргументы, то в метод bind() попадет None, но не объект-функция.

На помощь приходят так называемые анонимные объекты-функции Python, которые создаются инструкцией lambda. Применительно к нашей программе выглядеть это будет так:

```
…
l.bind('<Button-1>', lambda event, f="Verdana": changeFont(event, f))
l.bind('<Button-3>', lambda event, f="Times": changeFont(event, f))
…
```

Лямбда-функции можно использовать не только с методом bind(), но и опцией command, имеющейся у ряда виджет. Если функция передается через command, ей не нужен параметр event. Здесь обрабатывается только одно основное событие для виджета – клик левой кнопкой мыши.

У меток нет command, однако это свойство есть у кнопок:

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

def changeFont(font):
    l['font'] = font

l = Label(text="Hello World")
l.pack()
Button(command=lambda f="Verdana": changeFont(f)).pack()
Button(command=lambda f="Times": changeFont(f)).pack()

root.mainloop()
```

Практическая работа

Напишите программу по следующему описанию. Нажатие Enter в однострочном текстовом поле приводит к перемещению текста из него в список (экземпляр Listbox). При двойном клике (<Double-Button-1>) по элементу-строке списка, она должна копироваться в текстовое поле.