Programowanie aplikacji graficznej w Python z modułem Tkinter

Opracowanie Piotr Kotarski

(na podstawie :https://realpython.com/python-gui-tkinter/#building-your-first-python-gui-application-with-tkinter)

# Budowanie pierwszej aplikacji z Tkinter

Podstawowym elementem GUI Tkinter jest okno . Okna to kontenery, które zawierają wszystkie inne elementy GUI. Te inne elementy GUI, takie jak pola tekstowe, etykiety i przyciski, są nazywane widżetami. Widgety znajdują się wewnątrz okien.

## Utworzenie okna aplikacji:

* zaimportowanie modułu Python GUI Tkinter:

import tkinter as tk

Okno jest instancją tkinter w klasie Tk.

* Utworzenie okna i przypisanie do zmiennej (tu okno):

okno = tk.Tk()

Kiedy wykonasz powyższy kod, na ekranie pojawi się nowe okno. Wygląd zależy od systemu operacyjnego:



## Dodawanie widżetu Label

* utworzenie widżetu

napis = tk.Label(text="Witaj, człowieku")

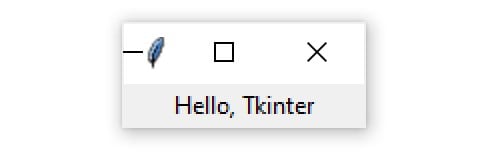
Utworzone wcześniej okno nie zmienia się.

Utworzony widżet należy dodać do okna. Istnieją trzy sposoby dodawania widżetów do okna.

* Dodanie do okna z wykorzystaniem metody pack():

napis.pack()

Okno wygląda teraz tak:



Po wstawieniu widżetu do okna za pomocą pack(), Tkinter dopasowuje to okno rozmiaru widżetu.

## uruchomienie pętli zdarzeń Tkinter

okno.mainloop()

window.mainloop() nakazuje Pythonowi uruchomienie pętli zdarzeń Tkinter.   
Ta metoda nasłuchuje zdarzeń, takich jak kliknięcia przycisków lub naciśnięcia klawiszy, i blokuje uruchamianie dowolnego kodu, który następuje po niej, aż do zamknięcia okna, w którym jest wywoływana.

# Praca z widżetami

Widżety są to elementy, przez które użytkownicy wchodzą w interakcję z programem. Każdy widżet w Tkinter jest zdefiniowany przez klasę. Oto niektóre z dostępnych widżetów:

|  |  |
| --- | --- |
| **Klasa** | **Opis** |
| Label | Widget używany do wyświetlania tekstu i obrazów na ekranie |
| Button | Przycisk, który może zawierać tekst i po kliknięciu może wykonać akcję |
| Entry | Widżet do wprowadzania tekstu, który pozwala tylko na jedną linię tekstu |
| Text | Widżet do wprowadzania tekstu, który umożliwia wprowadzanie tekstu w wielu wierszach |
| Frame | Prostokątny region używany do grupowania powiązanych widżetów lub zapewniania wypełnienia między widżetami |

## Wyświetlanie tekstu i obrazów (Label)

Widżety Label służą do wyświetlania tekstu lub obrazów.  
Tekst wyświetlany przez Label nie może być edytowany przez użytkownika. Służy wyłącznie do wyświetlania.

* Utworzenie widżetu:

label = tk.Label(text="Hello, Tkinter")

Label wyświetla tekst z domyślnym kolorem tekstu systemowego i domyślnym kolorem tła tekstu systemowego. Zazwyczaj są to odpowiednio czarno-białe, ale możesz zobaczyć różne kolory, jeśli zmieniłeś te ustawienia w swoim systemie operacyjnym.

* Można kontrolować w/w kolory (parametry foreground i background):

label = tk.Label(

text="Hello, Tkinter",

foreground="white", # Set the text color to white

background="black" # Set the background color to black

)

* lub nazwy skrócone bg i fg:

label = tk.Label(text="Hello, Tkinter", fg="white", bg="black")

Istnieje wiele prawidłowych nazw kolorów, w tym: "red" "orange" "yellow" "green" "blue" "purple"

Wiele nazw kolorów HTML współpracuje z Tkinter.

* Można także określić kolor za pomocą szesnastkowych wartości RGB :

label = tk.Label(text="Hello, Tkinter", bg="#34A2FE")

* Określenie rozmiarów widżetu:

label = tk.Label(

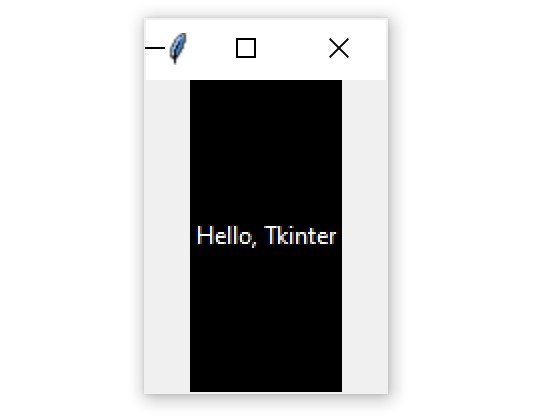
text="Hello, Tkinter",

fg="white", bg="black",

width=10, height=10

)

Oto jak ta etykieta wygląda w oknie:



* Uwaga:

Szerokość i wysokość są mierzone w jednostkach tekstowych:  
Jedna pozioma jednostka tekstowa = szerokość znaku "0"  
Jedna pionowa jednostka tekstowa = wysokość znaku "0".

Ten sposób wymiarowania zapewnia spójne działanie aplikacji na różnych platformach.

* Wyświetlanie obrazów

Etykiety mogą również wyświetlać obraz zamiast tekstu.

Dwa etapy:  
Utworzenie obiektu obrazu.  
Ustawienie parametru image:

obraz = PhotoImage(file='myimage.gif')

label['image'] = obraz

* Etykiety mogą również wyświetlać jednocześnie obraz i tekst.

O sposobie wyświetlania decyduje parametr compound:

|  |  |
| --- | --- |
| none | obraz, jeżeli jest obecny, w przeciwnym wypadku tekst |
| text | tylko tekst |
| image | tylko obraz |
| center | tekst na środku obrazu |
| top | obraz nad tekstem |
| left | obraz z lewej |
| bottom | obraz pod tekstem |
| right | obraz z prawej |

## Wyświetlanie "klikalnych" przycisków (Button)

Widżety Button służą do wyświetlania klikalnych przycisków. Efektem kliknięcia jest wywołanie wskazanej funkcji.

* Utworzenie przycisku:

button = tk.Button(

text="Click me!",

width=25, height=5,

bg="blue", fg="yellow",

)

Oto jak wygląda przycisk w oknie:



## Uzyskiwanie danych wejściowych użytkownika (Entry)

Widżet Entry wyświetla małe pole tekstowe, w które użytkownik może wpisać jakiś tekst. Może być on odczytany w programie.

* Tworzenie widżetu:

entry = tk.Entry(fg="yellow", bg="blue", width=50)

Podstawowe metody widżetu:

|  |  |
| --- | --- |
| get() | Pobieranie tekstu |
| delete()  delete(0)  delete(0,4)  delete(0,tk.END) | Usuwanie tekstu  znak 0  znaki 0-3  wszystkie |
| insert(0,"tekst") | Wstawianie tekstu od podanej pozycji |

* przykład:

import tkinter as tk

window = tk.Tk()

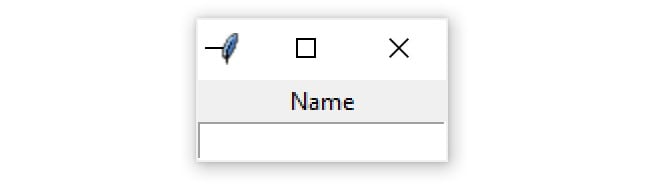
label = tk.Label(text="Name")

entry = tk.Entry()

label.pack()

entry.pack()

Oto jak to wygląda:



Tkinter automatycznie wyśrodkowuje widżety oknie. Jest to domyślna funkcja programu .pack().

## Pobieranie wielowierszowych tekstów (Text)

Widżet Text działa podobnie jak Entry, z tym, że teksty mogą być wielowierszowe.

Podstawowe metody widżetu:

|  |  |
| --- | --- |
| get("linia.znak")  get("lp.zp","lk.zk")  get("1.0", tk.END) | Pobieranie 1 znaku (linia od 1, znak od 0)  Pobieranie znaków od lp.zp do lk.zk-1  Pobieranie całego tekstu |
| delete()  delete("1.0")  delete("1.0","2.4")  delete("1.0",tk.END) | Usuwanie tekstu  znak 0  znaki z przedziału  wszystkie |
| insert("1.0","tekst")  insert(tk.END,"tekst") | Wstawianie tekstu od podanej pozycji  Wstawianie tekstu na końcu |

## Przypisywanie widżetów do ramek (Frame)

W tym samouczku będziesz pracować tylko z pięcioma widżetami. To są cztery, które widziałeś do tej pory, plus Framewidget. Framewidżety są ważne dla organizowania układu widżetów w aplikacji.

Zanim przejdziesz do szczegółów dotyczących tworzenia wizualnej prezentacji widżetów, przyjrzyj się bliżej działaniu Framewidżetów i sposobom przypisywania do nich innych widżetów. Poniższy skrypt tworzy pusty Framewidżet i przypisuje go do głównego okna aplikacji:

import tkinter as tk

window = tk.Tk()

frame = tk.Frame()

frame.pack()

window.mainloop()

frame.pack()pakuje ramę do okna, tak aby rozmiar okna był tak mały, jak to możliwe, aby objąć ramę. Po uruchomieniu powyższego skryptu otrzymujesz naprawdę nieciekawe dane wyjściowe:

## Adjusting Frame Appearance With Reliefs

## Understanding Widget Naming Conventions

## Check Your Understanding

# Kontrolowanie układu widżetów w oknie aplikacji

Istnieją trzy tzw. menedżery geomertii:

.pack()

.place()

.grid()

Każde okno lub ramka może używać tylko jednego menedżera geometrii. Jednak różne ramki mogą używać różnych menedżerów geometrii, nawet jeśli są przypisane do ramki lub okna za pomocą innego menedżera geometrii.

## Menedżer .pack()

Widżety wstawiane są w określonej kolejności.   
Domyślnie:

* Wstaw widżet pod poprzednim
* Dostosuj wymiary okna do wymiarów widżetu,
* Wyśrodkuj widżet

Przykład:

import tkinter as tk

window = tk.Tk()

frame1 = tk.Frame(master=window, width=100, height=100, bg="red")

frame1.pack()

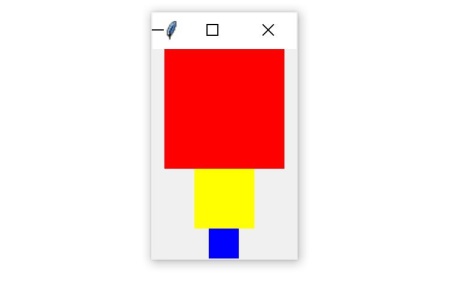
frame2 = tk.Frame(master=window, width=50, height=50, bg="yellow")

frame2.pack()

frame3 = tk.Frame(master=window, width=25, height=25, bg="blue")

frame3.pack()

window.mainloop()



* Parametry:

|  |  |
| --- | --- |
| fill=tk.X | wypełnienie w kierunku poziomym |
| fill=tk.Y | wypełnienie w kierunku pionowym |
| fill=tk.BOTH | wypełnienie w obu kierunkach |
| side=tk.TOP  side=tk.BOTTOM  side=tk.LEFT  side=tk.RIGHT | po której stronie okna widget powinien być umieszczony.  (domyślnie tk.TOP) |

* Odstępy między widżetami:

Można "oddalić" widżety od siebie ustalając parametry padx i pady:

button.pack(padx=5, pady=5)

Wymiary w pikselach

## Menedżer .place()

Menedżer place() umożliwia wstawienie widżetu w konkretne miejsce okna.

Podawane są dwa argumenty x i y oznaczające współrzędne lewego górnego rogu podane w pikselach.

import tkinter as tk

window = tk.Tk()

frame = tk.Frame(master=window, width=150, height=150).pack()

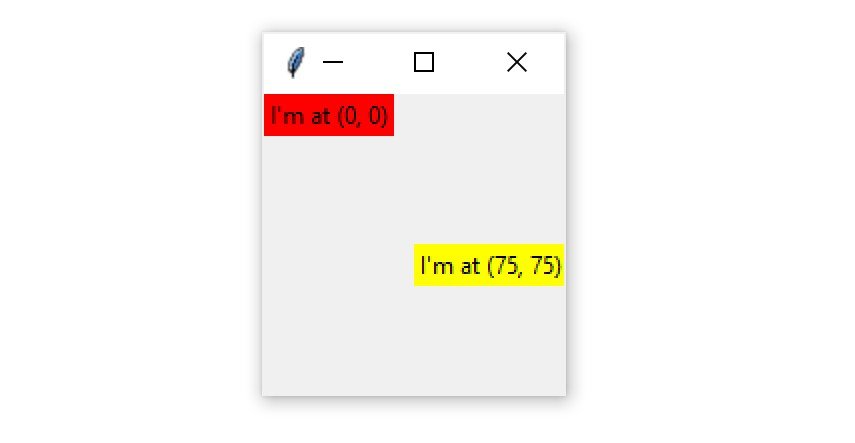
label1 = tk.Label(master=frame, text="I'm at (0, 0)", bg="red")

label1.place(x=0, y=0)

label2 = tk.Label(master=frame, text="I'm at (75, 75)", bg="yellow")

label2.place(x=75, y=75)

window.mainloop()



Uwaga: Układy utworzone za pomocą .place()nie są responsywne. Nie zmieniają się wraz ze zmianą rozmiaru okna.

## Menedżer .grid()

.grid() działa, dzieląc okno na wiersze i kolumny (podobnie do tabeli). Lokalizację widżetu określa się, wywołując np.: xxx.grid(row=2, column=3) i przekazując indeksy wierszy i kolumn. Indeksy zaczynają się od 0.

Przykład:

import tkinter as tk

window = tk.Tk()

for i in range(3):

for j in range(3):

frame = tk.Frame(

master=window,

relief=tk.RAISED,

borderwidth=1

)

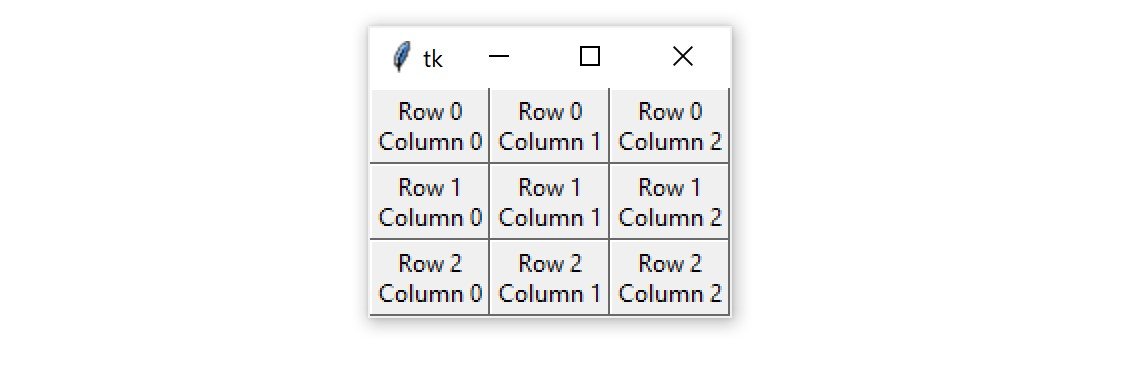
frame.grid(row=i, column=j)

label = tk.Label(master=frame, text=f"Row {i}\nColumn {j}")

label.pack()

window.mainloop()

Efekt:



* Parametry

|  |  |
| --- | --- |
| **Parametr** | **Opis** |
| column | numer kolumny (od 0) |
| row | numer wiersza (od 0) |
| rowspan | liczba połączonych komórek w pionie (łączone wiersze) |
| columnspan | liczba połączonych komórek w poziomie (łączone kolumny) |
| sticky | Jeśli komórka jest większa niż widżet, opcja określa położenie widżetu w komórce.Wartości:  N 🡑 E 🡒 S 🡓 W 🡐  NW 🡔 NE 🡕 SW 🡗 SE 🡖  EW 🡘 NS 🡙 (rozciągnięcie) |
| padx | margines zewnętrzny |
| pady | margines zewnętrzny |
| ipadx | margines wewnętrzny |
| ipady | margines wewnętrzny |

## Check Your Understanding

# Making Your Applications Interactive

## Using Events and Event Handlers

## Using .bind()

## Using command

## Check Your Understanding

# Building a Temperature Converter (Example App)

# Building a Text Editor (Example App)

# Conclusion

# Additional Resources