14. Programování - Tkinter - speciální komponenty

Události - princip, obsluha událostí, typy událostí

Tkinter poskytuje mechanismus, který nám umožňuje zachytávat události jako posun myši nebo stisk tlačítka na klávesnici.

U každé komponenty můžeme svázat (**bind**) jeho události s funkcemi. Pokud na komponentě nastane příslušná událost, daný ovladač je zavolán s parametrem popisujícím tuto událost.

Zachytávání myši

<Button-1>, <B1-Motion>, <ButtonRelease-1>, <Enter>, <Leave>, ...

Button 1 je vlevo, button 2 uprostřed a button 3 vpravo na myši.

```
def volanafunkce(udalost):
    print ("kliknuto na pozici:", udalost.x, udalost.y)
ramec = Frame(hlavni, width=100, height=100)
ramec.bind("<Button-1>", volanafunkce)
ramec.pack()
```

Zachytávání klávesnice

<Key> - jakákoli klávesa, <Return>, <F1>, …, klasická písmena bez ostrých závorek

Události klávesnice jsou posílány komponentě, která aktuálně má focus. Pokud tedy chceme, aby klávesnice fungovala v celé aplikaci, svážeme příslušnou událost s hlavním oknem nebo nejvyšším Framem.

```
def klavesa(udalost):
    print ("stisknuto", udalost.char)
ramec = Frame(hlavni, width=100, height=100)
ramec.bind("<Key>", klavesa)
ramec.pack()
ramec.focus_set()
```

Atributy události:

- widget - komponenta, která vyvolala tuto událost
- x, y - současná pozice myši, v pixelech
- x_root, y_root - současná pozice myši relativně k levému hornímu rohu obrazovky, v pixelech
- char - znak, jen u klávesnicových událostí
- keysym - název klávesy, jen u klávesnicových událostí
- keycode - kód klávesy, jen u klávesnicových událostí
- num - číslo tlačítka, jen u myších událostí
- width, height - nová velikost komponenty, v pixelech jen u Configure událostí
- type - typ události

proměnné knihovny Tkinter

tkProměnné se dají použít u většiny komponent (parametr **textvariable, variable**) a mít tak možnost sledovat změny zadávaných hodnot a dynamicky je měnit. **Checkbutton**, **Radiobutton** a **OptionMenu** dokonce použití těchto proměnných vyžadují.

Vytvoření:

```
pr = StringVar() # BooleanVar, IntVar, DoubleVar
```

Metoda .**get()** vrací aktuální hodnotu proměnné.

Metoda .**set(**hodnota**)** aktualizuje proměnnou a vyrozumívá všechny sledovatele. Parametr musí mít správný typ podle typu proměnné.

Metoda .**trace(**mód, funkce**)** se používá na spojení proměnné s funkcí. Funkce je volána vždy, když dojde např. ke změně proměnné (při módu “w”):

```
def funkce(*args):
    print ("změna!")
pr = StringVar()
pr.trace("w", funkce)
pr.set("ahoj")
```

standardní dialogy a jejich použití

Tkinter poskytuje rozhraní na standardní dialogy. Modul **messagebox** je nutné naimportovat:

```
from tkinter import messagebox
```

Metody:

1. **Pro prezentaci informací**
   - showinfo
   - showwarning
   - showerror
2. **Pro kladení otázek**
   - askquestion - vrací “yes”/“no”
   - askokcancel - ostatní vrací True/False
   - askyesno
   - askretrycancel
   - askyesnocancel - True/False/None

Syntax: messagebox.method(název, zpráva)

Název je zobrazen v titulku okna, a zpráva v dialogovém těle. Můžete použít \n ve zprávě pro text přes více řádků.

barevný dialog

Pro zobrazení barevného dialogu, který umožňuje uživateli vybrat si libovolnou barvu, slouží modul **colorchooser.** Je třeba ho importovat:

```
from tkinter import colorchooser
```

Jeho metoda **askcolor** vrací 2-prvkovou n-tici ve tvaru **((R, G, B), “hex”)** vybrané barvy. Je-li dialog zrušen uživatelem, je vrácena hodnota **None**.

```
colorchooser.askcolor(title=”Výběr barvy”, initialcolor=”#000000”)
```

Jaká barva se po zapnutí dialogu zobrazí se nastavuje parametrem **initialcolor**.

komponenty Menu, Spinbox, Text, Canvas - vlastnosti a použití

Hlavní menu je zobrazeno hned pod titulkovým pruhem okna. K vytvoření hlavního menu vytvoříme nejprve novou instanci Menu, položky přidáváme pomocí metod **add_*** a zobrazíme přes .config.

```
hlavniMenu = Menu(hlavni) # vytvoření hlavního menu
hlavniMenu.add_command(label="Konec", command=root.destroy)
hlavni.config(menu=hlavniMenu) # zobrazení menu
```

Podmenu se vytvářejí podobným způsobem. Hlavní rozdíl je, že k rodičovskému menu se musí připojit pomocí **add_cascade**.

```
hlavniMenu = Menu(hlavni) # vytvoření hlavního menu
```

```
# vytvořit rozbalovací menu a přidat ho k hlavnímu menu
menuSoubor = Menu(hlavniMenu, tearoff=0)
menuSoubor.add_command(label="Otevřít", command=hello)
menuSoubor.add_separator()
menuSoubor.add_command(label="Pryč", command=root.quit)
hlavniMenu.add_cascade(label="Soubor", menu=menuSoubor)
hlavni.config(menu=hlavniMenu) # zobrazení hlavního menu
```

Spinbox (číselník) je varianta komponenty Entry umožňující uživateli zvolit číselnou hodnotu v určitém rozsahu šipkami a vpisování lib. textu. Hodnoty pro šipky specifikují parametry **from_**, **to, increment** anebo **values**. Užitečné je použití stavu READONLY pro ovládání jen šipkami.

```
hodnota=StringVar()
hodnota.set(0)
def Nastav():
   print(hodnota.get())
Spinbox(hlavni,from_=0, to=10, increment=2,textvariable=hodnota,command=Nastav).pack()
Spinbox(hlavni,values=(1,2,4,8)).pack()
```

Text nabízí zobrazení a editaci textu s různými styly. Podporuje také vkládání obrázků a oken. Indexuje se ve formátu řádek.pozice, řádek začíná 1, pozice 0. Speciální indexy (INSERT, END) a stavy (DISABLED, NORMAL…) fungují jako u Entry.

Metody:

.tag_config(název, font=, foreground=, underline=) - definice stylu

```
text.tag_config("modry", font="Arial 20 italic", foreground="blue")
```

.insert(řádek.pozice, řetězec, styl) - vkládá do textového pole řetězec (formátovaný zvoleným stylem) na zadaný řádek od dané pozice, pokud daná pozice neexistuje, vkládá na konec textu

```
text.insert(1.0,"Ahoj studenti!\n","modry")
text.insert(END, "To je dnes hezky...")
```

.get(od, do) - získání řetězce

```
retezec=text.get(1.0, END)
```

.delete(od, do) - smázání vymezené části textu

Canvas (plátno) nám umožňuje vykreslovat vektorové objekty. Chceme-li přidat (nakreslit) na plátno nové prvky, použijeme metody **create_***.

```
w = Canvas(hlavni, width=200, height=100)
w.pack()
w.create_line(0, 100, 200, 0, fill="red", dash=(4, 4))
w.create_rectangle(50, 25, 150, 75, fill="blue", outline=”green”)
```

Objekty (items) plátna:

- line (čára)

- rectangle (obdélník) - souřadnice úhlopříčky (levý horní a pravý dolní)
- oval (kruh nebo elipsa) - souřadnice zadáváme jako obdélníku
- arc (oblouk/tětiva/výseč kruhu) - parametr style=arc/chord/pieslice
- polygon (mnohoúhelník)
- bitmap (bitmapový obrázek *.xbm vestavěný nebo načtený ze souboru)
- image (ostatní obrázky BitmapImage nebo PhotoImage)
- text - souřadnice, text, anchor
- window (okno)

Prvky přidané na plátno tam zůstávají, dokud je neodstraníme - pokud přemaluji přes čtverec jiný čtverec, původní čtverec se překryje, ale nesmaže.

Objekty jsou uloženy někde uvnitř canvasu a každý má své číslo (celé). Tento **index** získáme jako návratovou hodnotu při vytvoření daného objektu. Potom podle něho můžeme u objektů měnit jejich vlastnosti.

Metody na změnu prvků:

- .**itemconfig**(id,**parametry)
- .**coords**(id,x1,y1,x2,y2) - změna souřadnic, pokud neuvedeme souřadnice, vrací aktuální
- .**move**(id, x, y) - posun o daný vektor
- .**delete**(id) - smazání

Jako identifikátor můžeme také použít **tagy**. Ty se definují u prvků plátna parametrem **tags** (řetězec jmen oddělených mezerou). Canvas poskytuje také dva přednastavené tagy:

- ALL - zahrnuje všechny prvky na plátně
- CURRENT - zahrnuje prvek pod kurzorem myši