

Grafika v tkinteri

Maturitná príprava – poznámky

Modul **tkinter** je súčasťou Pythonu a umožňuje vytvárať programy s oknami: kresliť na plátno, reagovať na klikanie myšou a klávesy, robiť animácie a používať tlačidlá či textové polia. Grafické úlohy patria k typickým úlohám praktickej maturity z informatiky. Na Mente beží tkinter v simulácii priamo v prehliadači; obrázky okien v týchto poznámkach vznikli spustením ukážok v tej istej simulácii.

Čo sa naučíš

- vytvoriť okno s plátnom a rozumieť súradniciam plátna,
- kresliť čiary, obdĺžniky, elipsy, mnohoúhelníky, oblúky a text,
- meniť nakreslené objekty (presun, farba, zmazanie),
- reagovať na kliknutie myšou a stlačenie klávesu,
- animovať pomocou `after`,
- používať tlačidlo (`Button`), textové pole (`Entry`) a nápis (`Label`).

1 Okno a plátno

Každý program s grafikou má rovnakú kosť:

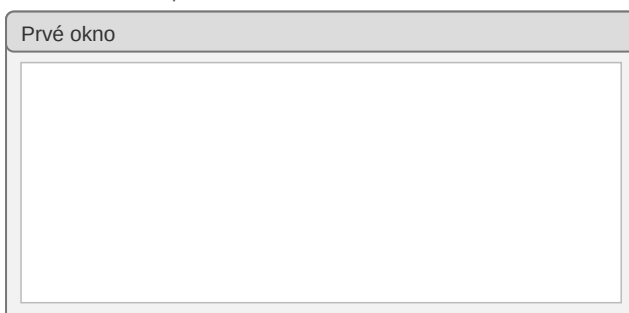
```
import tkinter

okno = tkinter.Tk()                                # hlavné okno
okno.title("Prvé okno")
platno = tkinter.Canvas(okno, width=300, height=120, bg="white")
platno.pack()                                     # umiestni plátno do okna

# sem príde kreslenie

okno.mainloop()                                  # okno čaká na udalosti
```

Prázdne okno s plátnom:

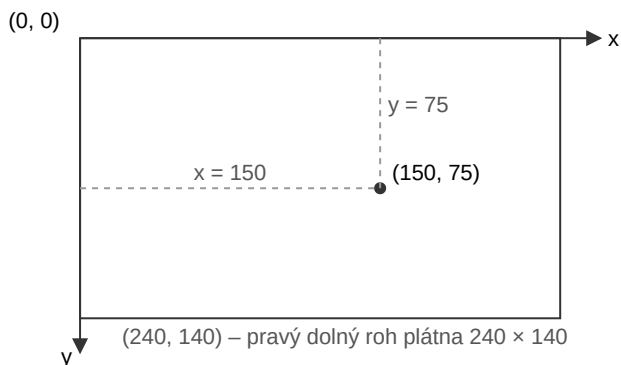


- `tkinter.Tk()` vytvorí **okno**.
- `tkinter.Canvas(...)` vytvorí **plátno** (*canvas*) zadanej šírky a výšky, `bg` je farba pozadia.
- `pack()` plátno do okna **umiestni** – bez neho by sa nezobrazilo.
- `mainloop()` spustí čakanie na udalosti (kliky, klávesy, časovače). Bez neho by sa okno v bežnom Pythone hneď zavrelo. Patrí na **koniec** programu.

Často sa používa aj skrátенý zápis bez premennej `okno`: `platno = tkinter.Canvas(width=300, height=120)` a na konci `platno.mainloop()` – okno sa vtedy vytvorí automaticky.

1.1 Súradnice

Poloha na plátne sa zadáva súradnicami `x` a `y` v **pixeloch**. Počiatok `(0, 0)` je v **ľavom hornom rohu**; `x` rastie doprava a `y` **rastie nadol** – opačne ako v matematike.



Bod `(150, 75)` je teda 150 pixelov od ľavého okraja a 75 pixelov od **horného** okraja. Stred plátna šírky `w` a výšky `h` má súradnice `(w / 2, h / 2)`.

2 Kreslenie

Kresliace metódy plátna sa začínajú `create_`. Ich prvé argumenty sú súradnice, za nimi nasledujú pomenované nastavenia (farba, hrúbka...).

Metóda	Čo nakreslí	Súradnice
<code>create_line(x1, y1, x2, y2, ...)</code>	úsečku alebo lomenú čiaru	body, ktorými čiara prechádza
<code>create_rectangle(x1, y1, x2, y2)</code>	obdĺžnik	dva protiľahlé rohy
<code>create_oval(x1, y1, x2, y2)</code>	elipsu (kruh) vpísanú do obdĺžnika	rohy opísaného obdĺžnika
<code>create_polygon(x1, y1, x2, y2, ...)</code>	mnohouholník	vrcholy
<code>create_arc(x1, y1, x2, y2, start=..., extent=...)</code>	výsek / oblúk elipsy	ako pri <code>create_oval</code>
<code>create_text(x, y, text=...)</code>	text	bod, ku ktorému sa text ukotví

Nastavenie	Význam	Príklad
<code>fill</code>	farba výplne (pri čiare a texte farba čiar/písma)	<code>fill="red"</code> , <code>fill="#ff8800"</code>
<code>outline</code>	farba obrysu (obdĺžnik, elipsa, mnohouholník, oblúk)	<code>outline="black"</code> , <code>outline=""</code> (bez obrysu)
<code>width</code>	hrúbka čiar alebo obrysu	<code>width=3</code>
<code>font</code>	písmo textu: (meno, veľkosť, štýl)	<code>font=("Arial", 16, "bold")</code>
<code>anchor</code>	ktorá časť textu je v zadanom bode	<code>anchor="nw"</code> – ľavý horný roh

Farby sa zadávajú anglickým menom ("red", "gold", "lightblue", "darkgreen" ...) alebo kódom "#RRGGBB" – červená, zelená a modrá zložka v šesťnástkovej sústave ("ff0000" je červená, "#808080" sivá).

2.1 Semafor

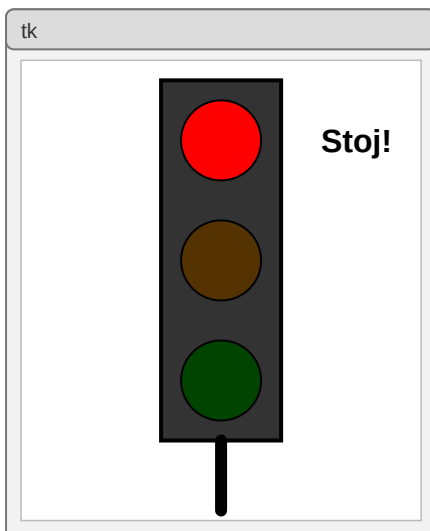
```
import tkinter

platno = tkinter.Canvas(width=200, height=230, bg="white")
platno.pack()

platno.create_rectangle(70, 10, 130, 190, fill="#333333", width=2)
platno.create_oval(80, 20, 120, 60, fill="red")
platno.create_oval(80, 80, 120, 120, fill="#553300")
platno.create_oval(80, 140, 120, 180, fill="#004400")
platno.create_line(100, 190, 100, 225, width=6)
platno.create_text(150, 40, text="Stoj!", anchor="w", font=("Arial", 12, "bold"))

platno.mainloop()
```

Okno programu:



Kruh nakreslíme ako elipsu vpísanú do **štvorca**: kruh so stredom (sx, sy) a polomerom r je `create_oval(sx - r, sy - r, sx + r, sy + r)`. Tento vzťah sa v úlohách používa neustále.

2.2 Mnohouholník a text

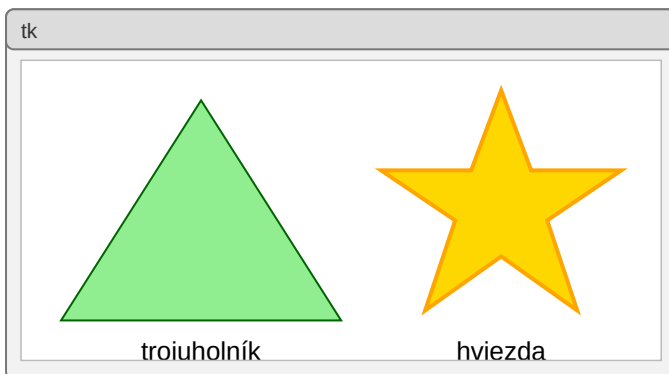
```
import tkinter

platno = tkinter.Canvas(width=320, height=150, bg="white")
platno.pack()

# trojuholník: tri vrcholy
platno.create_polygon(20, 130, 90, 20, 160, 130,
                     fill="lightgreen", outline="darkgreen")
# päťcípá hviezda: 10 vrcholov
platno.create_polygon(240, 15, 255, 55, 300, 55, 263, 80, 278, 125,
                     240, 98, 202, 125, 217, 80, 180, 55, 225, 55,
                     fill="gold", outline="orange", width=2)
platno.create_text(90, 145, text="trojuholník", font=("Arial", 10))
platno.create_text(240, 145, text="hviezda", font=("Arial", 10))

platno.mainloop()
```

Okno programu:



`create_polygon` dostane súradnice všetkých vrcholov za sebou a posledný vrchol sám spojí s prvým. Predvolená výplň mnohouholníka je čierna, preto `fill` zadávame vždy.

2.3 Kreslenie v cykle

Sila programovania sa ukáže, keď kreslíme cyklom. **Terč** – sústredné kruhy so spoločným stredom a zmenšujúcim sa polomerom:

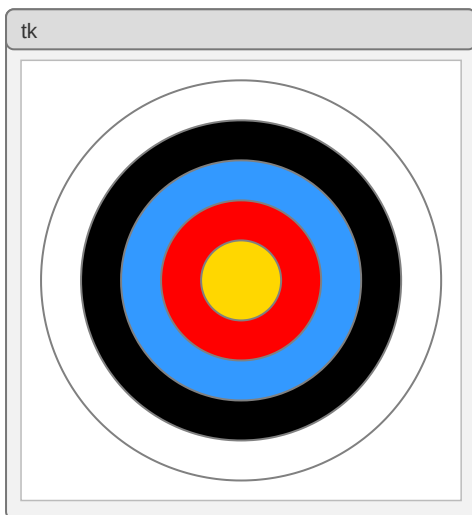
```
import tkinter

platno = tkinter.Canvas(width=220, height=220, bg="white")
platno.pack()

sx, sy = 110, 110
farby = ["white", "black", "#3399ff", "red", "gold"]
for i in range(5):
    r = 100 - i * 20
    platno.create_oval(sx - r, sy - r, sx + r, sy + r,
                       fill=farby[i], outline="gray")

platno.mainloop()
```

Okno programu:



Šachovnica – dva vnorené cykly cez riadky a stĺpce; farba políčka závisí od toho, či je súčet riadku a stĺpca párny:

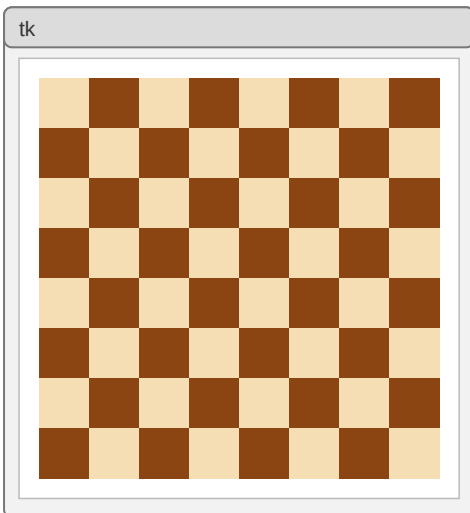
```
import tkinter

platno = tkinter.Canvas(width=220, height=220, bg="white")
platno.pack()

d = 25 # veľkosť políčka
for riadok in range(8):
    for stlpec in range(8):
        x = 10 + stlpec * d
        y = 10 + riadok * d
        farba = "wheat" if (riadok + stlpec) % 2 == 0 else "saddlebrown"
        platno.create_rectangle(x, y, x + d, y + d, fill=farba, outline="")

platno.mainloop()
```

Okno programu:



2.4 Oblúky: koláčový graf

`create_arc` kreslí časť elipsy. Uhol `start` sa meria v stupňoch od smeru „na tri hodiny“ **proti smeru hodinových ručičiek**, `extent` je veľkosť výseku v stupňoch. Výseky koláčového grafu teda začínajú tam, kde predchádzajúci skončil:

```
import tkinter

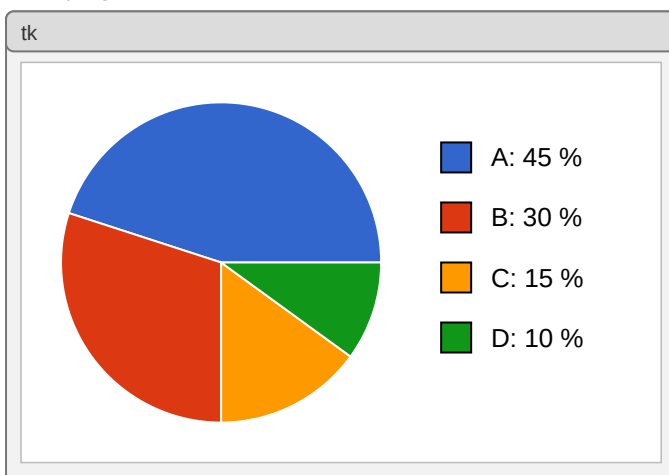
hlasy = {"A": 45, "B": 30, "C": 15, "D": 10}      # percentá
farby = ["#3366cc", "#dc3912", "#ff9900", "#109618"]

platno = tkinter.Canvas(width=320, height=200, bg="white")
platno.pack()

start = 0
for i, (strana, p) in enumerate(hlasy.items()):
    uhol = p * 360 / 100
    platno.create_arc(20, 20, 180, 180, start=start, extent=uhol,
                      fill=farby[i], outline="white")
    platno.create_rectangle(210, 40 + i * 30, 225, 55 + i * 30, fill=farby[i])
    platno.create_text(235, 47 + i * 30, text=f"{strana}: {p} %", anchor="w")
    start += uhol

platno.mainloop()
```

Okno programu:



2.5 Náhodné kreslenie

S modulom `random` sa dajú kresliť náhodné obrázky – napríklad hviezdna obloha z bodiek s náhodnou polohou a veľkosťou:

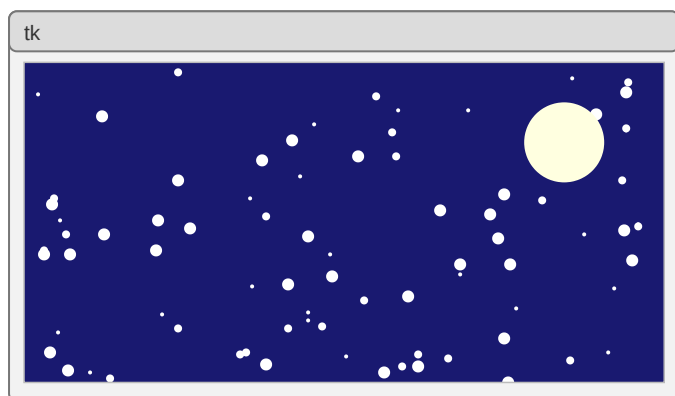
```
import tkinter
import random

platno = tkinter.Canvas(width=320, height=160, bg="midnightblue")
platno.pack()

for _ in range(80):
    x = random.randint(0, 320)
    y = random.randint(0, 160)
    r = random.randint(1, 3)
    platno.create_oval(x - r, y - r, x + r, y + r, fill="white", outline="")
platno.create_oval(250, 20, 290, 60, fill="lightyellow", outline="")

platno.mainloop()
```

Okno programu:



Náhodnú farbu zo zoznamu vyberie `random.choice(["red", "blue", "green"])`. Úplne náhodnú farbu vytvoríš z troch náhodných zložiek: `f"#{random.randint(0, 255):02x}{random.randint(0, 255):02x}{random.randint(0, 255):02x}"` (formát `02x` zapíše číslo dvoma šesťnástkovými ciframi).

3 Práca s nakreslenými objektmi

Každá metóda `create_...` vráti **číslo objektu** (*id*). Pomocou neho môžeme objekt neskôr zmeniť:

Metóda	Čo urobí
<code>platno.move(id, dx, dy)</code>	posunie objekt o <code>dx</code> doprava a <code>dy</code> nadol
<code>platno.coords(id, x1, y1, x2, y2)</code>	nastaví nové súradnice; <code>platno.coords(id)</code> ich vráti
<code>platno.itemconfig(id, fill="red")</code>	zmení nastavenia (farbu, text, hrúbku...)
<code>platno.delete(id)</code>	zmaže objekt; <code>platno.delete("all")</code> zmaže všetko

```
import tkinter

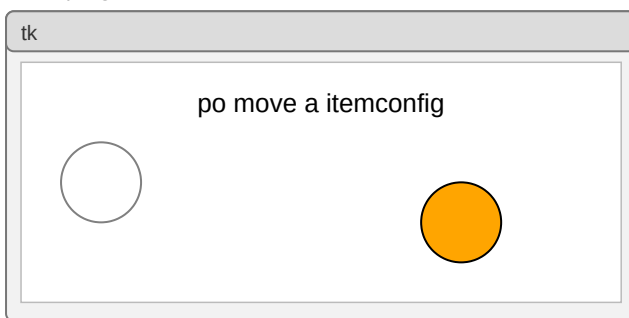
platno = tkinter.Canvas(width=300, height=120, bg="white")
platno.pack()

lopta = platno.create_oval(20, 40, 60, 80, fill="red")
napis = platno.create_text(150, 20, text="pôvodne")
tien = platno.create_oval(20, 40, 60, 80, outline="gray") # pôvodná poloha

platno.move(lopta, 180, 20) # posun doprava a nadol
platno.itemconfig(lopta, fill="orange")
platno.itemconfig(napis, text="po move a itemconfig")
print("nové súradnice lopty:", platno.coords(lopta))

platno.mainloop()
```

Okno programu:



Výstup v konzole:

```
nové súradnice lopty: [200.0, 60.0, 240.0, 100.0]
```

4 Udalosti: myš a klávesnica

Grafický program väčšinu času **čaká na udalosť** – kliknutie, stlačenie klávesu. Na udalosť zareaguje funkcia, ktorú si k udalosti „priviažeme“ metódou `bind`. Takejto funkcii sa hovorí **obsluha udalosti**; Python ju zavolá a ako argument jej odovzdá objekt `event` s informáciami o udalosti.

4.1 Kliknutie myšou

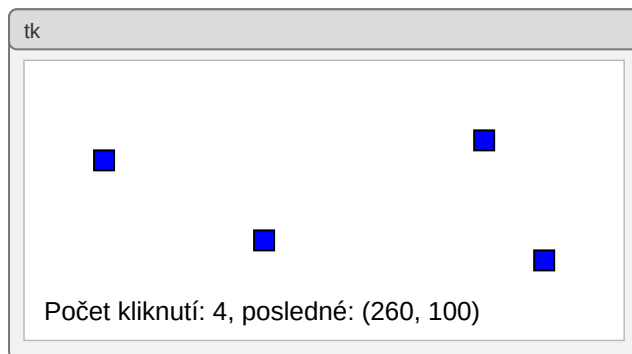
```
import tkinter

platno = tkinter.Canvas(width=300, height=140, bg="white")
platno.pack()
pocet = 0
info = platno.create_text(10, 125, text="Klikni do plátna.", anchor="w")

def klik(event):
    global pocet
    pocet += 1
    x, y = event.x, event.y
    platno.create_rectangle(x - 5, y - 5, x + 5, y + 5, fill="blue")
    platno.itemconfig(info, text=f"Počet kliknutí: {pocet}, posledné: ({x}, {y})")

platno.bind("<Button-1>", klik)
platno.mainloop()
```

Okno programu:



- "<Button-1>" je kliknutie **ľavým** tlačidlom ("<Button-3>" pravým).
- `event.x` a `event.y` sú súradnice miesta kliknutia na plátno.
- Pri `bind` píšeme meno funkcie **bez zátvoriek** – funkciu nevoláme my, zavolá ju tkinter pri každom kliknutí.
- Obsluha udalosti musí mať parameter `event`, aj keď ho nepoužije.
- Počítadlo `pocet` je globálna premenná, ktorú funkcia mení – preto `global pocet`. (V jednoduchých grafických programoch je to bežné riešenie.)

Ďalšie udalosti myši: "<B1-Motion>" – ťahanie so stlačeným tlačidlom (hodí sa na kreslenie „perom“), "<ButtonRelease-1>" – pustenie tlačidla, "<Double-Button-1>" – dvojklik.

4.2 Klávesy

Klávesy zachytávame pomocou `bind_all` (udalosť príde bez ohľadu na to, ktorá časť okna je aktívna). Šípky majú mená "`<Left>`", "`<Right>`", "`<Up>`", "`<Down>`"; ľubovoľný kláves zachytí "`<Key>`" a jeho meno je v `event.keysym`:

```
import tkinter

platno = tkinter.Canvas(width=300, height=150, bg="white")
platno.pack()
lopta = platno.create_oval(20, 20, 50, 50, fill="green")
popis = platno.create_text(150, 135, text="Ovládaj šípkami.")
KROK = 30

def vpravo(event):
    platno.move(lopta, KROK, 0)

def vlavo(event):
    platno.move(lopta, -KROK, 0)

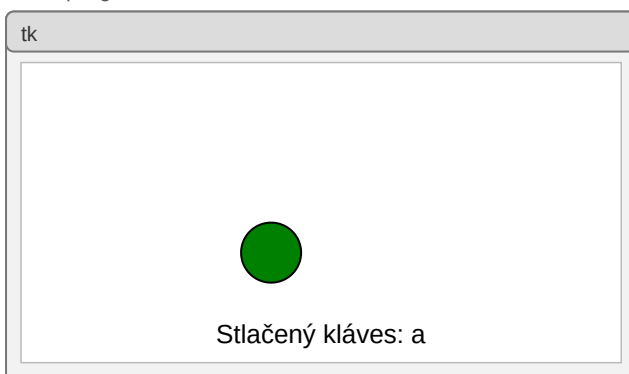
def hore(event):
    platno.move(lopta, 0, -KROK)

def dole(event):
    platno.move(lopta, 0, KROK)

def iny_klaves(event):
    platno.itemconfig(popis, text=f"Stlačený kláves: {event.keysym}")

platno.bind_all("<Right>", vpravo)
platno.bind_all("<Left>", vlavo)
platno.bind_all("<Up>", hore)
platno.bind_all("<Down>", dole)
platno.bind_all("<Key-a>", iny_klaves)
platno.mainloop()
```

Okno programu:



(Obrázok ukazuje stav po stlačení klávesov →, →, →, ↓, ↓ a a.)

5 Animácia: after

Animácia je séria malých zmien v krátkych časových odstupoch. V tkinteri na to slúži metóda `after(ms, funkcia)`: **naplánuje** zavolanie funkcie o zadaný počet milisekúnd. Animačná funkcia urobí jeden krok a na konci si naplánuje **sama seba** znova – vznikne „cyklus“, ktorý však okno nezablokuje:

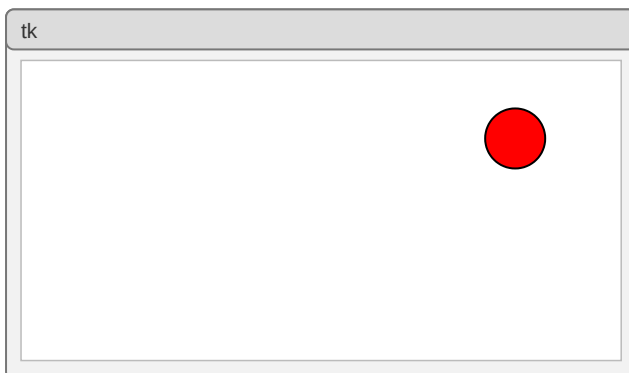
```
import tkinter

platno = tkinter.Canvas(width=300, height=150, bg="white")
platno.pack()
lopta = platno.create_oval(10, 60, 40, 90, fill="red")
dx, dy = 6, 4                                # rýchlosť: o koľko sa posunie za jeden krok

def krok():
    global dx, dy
    platno.move(lopta, dx, dy)
    x1, y1, x2, y2 = platno.coords(lopta)
    if x1 <= 0 or x2 >= 300:                  # náraz do ľavej alebo pravej steny
        dx = -dx
    if y1 <= 0 or y2 >= 150:                  # náraz hore alebo dole
        dy = -dy
    platno.after(30, krok)                    # ďalší krok o 30 ms

krok()
platno.mainloop()
```

Okno programu po 1,5 s:



Loptička sa každých 30 ms posunie o `(dx, dy)`. Keď narazí na okraj, zmení sa znamienko príslušnej rýchlosti – odrazí sa. Obrázok ukazuje stav po 1,5 sekunde.

Pozor: Na animáciu nepoužívaj `while True` s `time.sleep`. Okno by počas cyklu nereagovalo na nič (ani na zatvorenie), kým by cyklus neskončil. `after` naplánuje ďalší krok a medzi krokmi nechá tkinter obsluhovať udalosti.

5.1 Zmena farby v čase

Časovač nemusí iba posúvať – môže meniť ľubovoľnú vlastnosť. Blikajúci nápis:

```
import tkinter

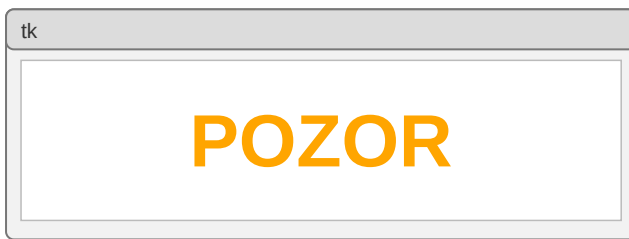
platno = tkinter.Canvas(width=300, height=80, bg="white")
platno.pack()
napis = platno.create_text(150, 40, text="POZOR", fill="red",
                           font=("Arial", 28, "bold"))

farby = ["red", "orange"]
index = 0

def blikaj():
    global index
    index = (index + 1) % len(farby)
    platno.itemconfig(napis, fill=farby[index])
    platno.after(500, blikaj)

blikaj()
platno.mainloop()
```

Okno programu po 1,2 s:



Zvyšok po delení (`% len(farby)`) zabezpečí, že sa index točí dookola: 0, 1, 0, 1...

6 Tlačidlá, textové polia a nápisy

Okrem plátna má tkinter aj ďalšie prvky okna (*widgety*). Najpoužívanejšie:

Prvok	Vytvorenie	Dôležité
tlačidlo	<code>tkinter.Button(okno, text="OK", command=funkcia)</code>	<code>command</code> – funkcia (bez zátvoriek a bez parametra)
textové pole	<code>pole = tkinter.Entry(okno)</code>	<code>pole.get()</code> vráti napísaný text (reťazec)
nápis	<code>napis = tkinter.Label(okno, text="...")</code>	<code>napis.config(text="nový text")</code>

Každý prvok treba umiestniť – `pack()` ich ukladá pod seba v poradí, v akom sa zavolá.

```
import tkinter

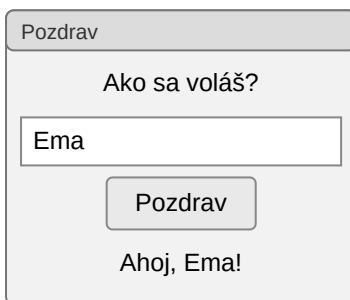
okno = tkinter.Tk()
okno.title("Pozdrav")

tkinter.Label(okno, text="Ako sa voláš?").pack()
pole = tkinter.Entry(okno)
pole.pack()
vysledok = tkinter.Label(okno, text="")

def pozdrav():
    meno = pole.get().strip()
    if meno:
        vysledok.config(text=f"Ahoj, {meno}!")
    else:
        vysledok.config(text="Najprv napíš meno.")

tkinter.Button(okno, text="Pozdrav", command=pozdrav).pack()
vysledok.pack()
okno.mainloop()
```

Okno programu:



(Obrázok: do poľa sa napísalo „Ema“ a stlačilo sa tlačidlo Pozdrav.)

Text z `Entry` je vždy **retazec** – ak s ním chceš počítať, preved' ho (`int(pole.get())`) a ošetri neplatný vstup (`try – except`, téma Výnimky).

6.1 Tlačidlá a plátno spolu

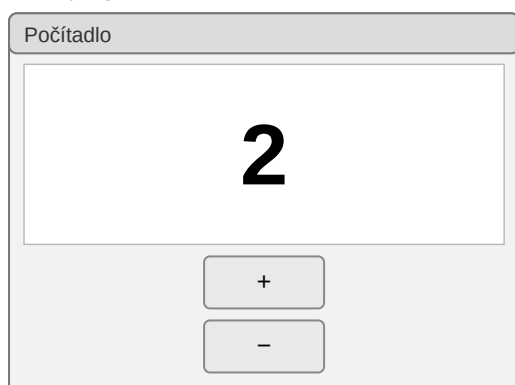
```
import tkinter

okno = tkinter.Tk()
okno.title("Počítadlo")
platno = tkinter.Canvas(okno, width=240, height=90, bg="white")
platno.pack()
hodnota = 0
text = platno.create_text(120, 45, text="0", font=("Arial", 32, "bold"))

def zmen(o_kolko):
    global hodnota
    hodnota += o_kolko
    platno.itemconfig(text, text=str(hodnota))

tkinter.Button(okno, text="+", command=lambda: zmen(1)).pack()
tkinter.Button(okno, text="-", command=lambda: zmen(-1)).pack()
okno.mainloop()
```

Okno programu:



`command` musí byť funkcia **bez parametrov**. Ak potrebujeme zavolať funkciu s argumentom, obalíme volanie do `lambda: zmen(1)` – to je krátka funkcia bez parametrov, ktorá pri stlačení tlačidla zavolá `zmen(1)`.

7 Časté chyby

- **Chýbajúce** `pack()` – plátno alebo tlačidlo sa nezobrazí.
- **Chýbajúce** `mainloop()` – v bežnom Pythone sa okno hneď zavrie (alebo sa vôbec neukáže).
- **Os y smeruje nadol** – „vyššie“ na obrazovke znamená **menšie** `y`.
- `command=funkcia()` **namiesto** `command=funkcia` – so zátvorkami sa funkcia zavolá hneď pri vytváraní tlačidla, nie pri stlačení.
- **Obsluha udalosti bez parametra:** `def klik():` pri `bind` skončí chybou – musí byť `def klik(event):`.
- **Zabudnuté** `global` – funkcia, ktorá mení globálnu premennú (počítadlo, rýchlosť), potrebuje `global` meno.
- **Animácia cez** `while` **a** `sleep` – okno zamrzne; použi `after`.
- **Text z Entry je reťazec** – na počítanie ho preved' na číslo.

Zhrnutie

- Kostra: `import tkinter, Canvas(width=..., height=...), pack(), kreslenie, mainloop()`.
- Súradnice: (0, 0) vľavo hore, x doprava, y nadol; kruh: `create_oval(sx - r, sy - r, sx + r, sy + r)`.
- `create_line`, `create_rectangle`, `create_oval`, `create_polygon`, `create_arc`, `create_text`; nastavenia `fill`, `outline`, `width`, `font`, `anchor`.
- `create_...` vráti id; `move`, `coords`, `itemconfig`, `delete` objekt menia.
- Udalosti: `platno.bind("<Button-1>", f)`, `platno.bind_all("<Left>", f)`; `def f(event): a event.x, event.y, event.keysym`.
- Animácia: funkcia urobí krok a zavolá `platno.after(ms, funkcia)`.
- `Button(..., command=f)`, `Entry + get()`, `Label + config(text=...)`.

Úlohy na precvičenie

1. Nakresli slovenskú vlajku bez znaku: tri vodorovné pruhy (biely, modrý, červený).
2. Nakresli rad 10 kruhov rovnakej veľkosti, ktorých farba sa strieda medzi červenou a modrou.
3. Po každom kliknutí nakresli na mieste kliknutia náhodne veľký kruh náhodnej farby.
4. Naprogramuj štvorec, ktorý sa pohybuje zľava doprava, a keď zmizne vpravo, objaví sa opäť vľavo.
5. Vytvor okno s textovým poľom a tlačidlom: po stlačení tlačidla nakresli na plátno toľko štvorcov vedľa seba, koľko je zadané v poli.
6. Pomocou "<B1-Motion>" naprogramuj jednoduché kreslenie: pri ťahaní myšou sa kreslia malé bodky.

Úlohy k tejto téme nájdeš na learnwithmente.com v časti Informatika → Python.