PRŮVODCE TEORIÍ

První program – Hello world

Editor *Mu*

Editor Mu stáhnete ze stránek https://codewith.mu .

Otevřete si editor M*u* a stiskněte tlačítko New. Měli byste vidět následující text:

```
from microbit import *

# Write your code here :-)
```

Na řádku 1 se zavádí systémová knihovna, která zavádí potřebné funkce a metody pro práci s micro:bitem. Tímto řádkem **musí** začínat všechny vaše programy.

Znak # na začátku třetího řádku znamená, že se jedná o komentář. Tento řádek můžete klidně smazat a kód programu psát místo něj. Je možné i psát hned na řádek 2.

Zkuste pro začátek následující kód:

```
from microbit import *
display.scroll("Ahoj svete")
```

Popis: řádek 2 znamená, že po displeji micro:bitu má běžet kód uvedený v uvozovkách. Stiskněte tlačítko Save a kód uložte. Programu bude automaticky přiřazena přípona py - rozpoznávací znamení pythonovských programů.

Nyní připojte svůj micro:bit pomocí USB kabelu k počítači. Počkejte asi pět vteřin a pak stiskněte tlačítko Flash. Vyčkejte až přestane blikat žlutá LED dioda na micro:bitu a pak byste měli vidět, jak váš text přeběhne přes displej micro:bitu.

Nyní můžete vyzkoušet následující modifikaci kódu:

```
from microbit import *
while True:
    display.scroll("Ahoj svete")
    sleep(1000)
```

Popis: Na řádku 2 je nyní zaveden tzv. nekonečný cyklus. Jeho příkazy jsou odsazené o čtyři mezerníky od začátku řádků. Pozor – je třeba dodržet stejný počet mezer (může být i vyšší, ale násobek čtyř) a nelze použít tabulátor. Některé pythonovské editoru mají v tomto syntaxi volnější, ale zde musíme dodržet tato pravidla. Na řádku 4 je pak příkaz `sleep` – čekej 1000 milisekund – 1 sekundu.

Program v nekonečné smyčce vypisuje text, pak čeká jednu sekundu a zase dokola.

Editor Thonny

Editor stáhneme ze stránek https://thonny.org/

Editor je původně určen pro práci s RaspBerry Pi Pico, ale snadno do něj nahrajeme rozšíření (plugin) pro práci s Micro:bitem. V nabídce Nástroje / Spravovat pluginy dáme do vyhledávače microbit a pak nainstalujeme plugin thonny-microbit. Editor nyní musíme vypnout a zapnout.

Nyní připojíme Micro:bit. Je třeba ještě v editoru vybrat jaký „kus“ hardware máme. Klikneme na nápis v pravém dolním rohu editoru a zvolíme možnost *configure interpreter*. Alternativně lze zvolit menu Nástroje / Volby / Interpretr. V horním rozbalovacím menu zvolíme MicroPython (Micro:bit) a v dolním port ke kterému je Micro:bit připojen.

Pokud tento Micro:bit připojujeme prvně doporučuji zvolit možnost Instal or update firmware (je nutné být připojen k internetu). Počkáme, než instalace proběhne a můžeme začít pracovat.

Okno editoru je vodorovně rozděleno na dvě poloviny. V horní se nachází prostor pro vlastní editaci programů a dolní je REPL (read–eval–print loop) – prostor pro zkoušení příkazů. Do něj lze psát příkazy a Micro:bit je rovnou vykonává.

Je zde ještě jeden velký rozdíl oproti editoru Mu. Pokud naeditujeme kód a stiskneme tlačítko play (zelený trojúhelník), kód se na micro:bitu spustí, ale neuloží se na něj. Po doběhnutí kódu, stisku tlačítka reset nebo odpojení micro:bitu, micro:bit na tento kód zapomene. Chceme-li náš kód uložit na Micro:bit, tak aby si jej pamatoval, je nutné z menu Soubor zvolit Uložit jako, vybrat micro:bit a soubor nazvat main.py. Takovýto soubor se automaticky spustí na Micro:bitu po připojení napájení nebo stisku tlačítka reset.

Další příklady

Zadání: Napište program, který vypíše čísla od jedné do deseti pomocí příkazu for a pak skončí.

Řešení:

```
from microbit import *
for i in range(1, 11):
    display.scroll(i)
```

Popis: Na řádku 2 je zaveden cyklus s pevným počtem opakování. Hodnota proměnné i se mění dle rozsahu intervalu `range(a, b)` od a do b-1. Chcete-li tedy od 1 do 10 musíme psát takto. Za čárkou v intervalu musí být v editoru Mu mezera. Pozor na konci řádku je dvojtečka, tady se také často dělá chyba. Na řádku tři je pak výpis čísla.

Zadání: Řešte předchozí příklad pomocí příkazu while

Řešení:

```
from microbit import *
i = 1
while (i < 11):
    display.scroll(i)
    i = i + 1
```

Popis: Na řádku 2 do proměnné i přiřadíte hodnotu 1. Pozor okolo = jsou v Mu vyžadovány mezery. Na řádku 3 je cyklus, který se opakuje dokud je i menší než 11. Pozor kolem nerovnosti

musí být mezery a na konci řádku je dvojtečka. Na řádku 5 zvyšujeme hodnotu proměnné i o jedničku. Pozor opět na chybějící mezery. U všech nutných mezer se jedná o syntaktická pravidla Mu a v jiných editorech nemusí být vyžadována.

Zadání: Po dobu jedné vteřiny zobrazte na displeji písmeno X.

Řešení:

```
from microbit import *
display.show("X")
sleep(1000)
display.clear()
```

Popis: Na řádku 2 zobrazíte písmeno X. Na řádku 3 čeká program jednu sekundu a příkaz na řádku 4 smaže displej.

Přednastavené obrázky

MicroPython obsahuje asi padesát připravených obrázků. Ukázka jejich použití je v následujícím kódu:

```
1 from microbit import *
2 display.show(Image.SAD)
3 sleep(1000)
4 display.show(Image.SMILE)
5 sleep(1000)
6 display.show(Image.HAPPY)
7 sleep(1000)
8 display.clear()
```

Zobrazení obrázků je na řádcích 2, 4 a 6. Jak je vidět, jedná se o konstanty začínající slovem Image.

Seznam všech obrázků naleznete v příloze A anebo v dokumentaci MicroPythonu pro micro:bit.

Příklad: Pomocí konstant obrázků `Image.HEART` a `Image.HEART_SMALL`, simulujte údery srdce.

Řešení:

```
from microbit import *
while True:
    display.show(Image.HEART)
    sleep(400)
    display.show(Image.HEART_SMALL)
    sleep(400)
```

Pauza `sleep(400)` je zvolena, aby frekvence odpovídala zhruba 75 úderům za minutu.

Vlastní obrázky

Příklad: Zobrazte na displeji obrázek rakety

```
from microbit import *
raketa = Image("00900:"
               "05550:"
               "05550:"
               "09990:"
               "90909:")
display.show(raketa)
```

Popis: Struktura na řádcích 2 až 6 popisuje obrázek. Pětice čísel ukončených dvojtečkou uzavřená do apostrofů popisuje vždy jeden řádek displeje shora dolů. Číslo pak znamená intenzitu světla od 0 (dioda nesvítí) po 9 (dioda svítí naplno). Na řádku 8 je pak příkaz pro zobrazení obrázku.

Je možná i syntaxese zápisem Image do jednoho řádku:

```
from microbit import *
raketa = Image("00900:05550:05550:09990:90909:")
display.show(raketa)
```

Nyní si na základě tohoto příkladu sestrojíme pohyblivý obrázek startující rakety. Zdrojový kód je následující:

```
from microbit import *
raketa1 = Image("00900:"
                "05550:"
                "05550:"
                "09990:"
                "90909:")
raketa2 = Image("00900:"
                "05550:"
                "05550:"
                "09990:"
                "99999:")
raketa3 = Image("05550:"
                "05550:"
                "09990:"
                "99999:"
                "00000:")
raketa4 = Image("09990:"
                "99999:"
                "00000:"
                "00000:"
                "00000:")
raketa5 = Image("99999:"
                "00000:"
                "00000:"
                "00000:"
                "00000:")
raketa6 = Image("00000:"
                "00000:"
                "00000:"
                "00000:"
                "00000:")
raketa = [raketa1, raketa2, raketa3, raketa4, raketa5, raketa6]
display.show(raketa, delay=500)
```

Na řádcích 2 až 36 je postupně šest obrázků, označených raketa1 až raketa6. Na předposledním řádku je z těchto obrázků sestavena struktura raketa. Tato struktura se nazývá list (seznam). Ta je pak na posledním řádku postupně zobrazována, s pauzou půl sekundy mezi jednotlivými snímky.

Práce s konkrétní diodou

Příklad: Sestrojte program, který bude náhodně rozsvěcet jednotlivé diody s různou intenzitou světla.

Řešení:

```
from microbit import *
import random
while True:
    x = random.randint(0, 4)
    y = random.randint(0, 4)
    intenzita = random.randint(0, 9)
    display.set_pixel(x, y, intenzita)
    sleep(10)
```

Program používá generátor náhodných čísel. Pro jeho použití je nutné načíst knihovnu random na řádku 2. Na řádcích 4 až 6 je pak tento generátor volán funkcí random.randint, která má dva parametry a, b a vrací náhodné celé číslo z uzavřeného intervalu <a,b>. Zde je postupně získána x-ová a y-ová souřadnice rozsvěcené diody a intenzita světla dané diody.

Funkce na řádku 7 display.set_pixel má tři parametry x, y, intenzita, získané v předchozím kroku a nastavuje podle nich na souřadnicích x (sloupec) a y (řádek) diodu na intenzitu (0 až 9). Bod (0, 0) je vlevo nahoře, vpravo dole pak (4, 4). Intenzita je 0 (nesvítí) až 9 (svítí naplno).

Použití funkce sleep je nutné jinak dochází k příliš rychlému „blikání“.

Příklad: Upravte předchozí zadání tak, že budete nastavovat pouze dvě úrovně intenzity (0 a 9) a to tak, že budete náhodně vybírat souřadnice a pokud dioda na dané souřadnici nebude svítit, tak jí rozsvítíte a naopak.

Řešení:

```
from microbit import *
import random
while True:
    x = random.randint(0, 4)
    y = random.randint(0, 4)
    if (display.get_pixel(x, y)):
        display.set_pixel(x, y, 0)
    else:
        display.set_pixel(x, y, 9)
    sleep(10)
```

Popis: Zde pouze vybíráme náhodně souřadnice diody. Na řádku šest je použita funkce `display.get_pixel(x, y)`. Ta vrací hodnotu svícení dané diody. Zde může být 0 nebo 9. V našem příkladě využíváme toho, že pokud dioda svítí, vrátí hodnotu větší než nula a tudíž je podmínka u příkazu `if` plněna. Pokud tedy dioda svítí zhasneme jí a naopak.

Pozor druhá úroveň odsazení musí být opět násobek čtyř a je tedy osm mezer.

Příloha – Seznam připravených obrázků

- `Image.HEART`
- `Image.HEART_SMALL`
- `Image.HAPPY`
- `Image.SMILE`
- `Image.SAD`
- `Image.CONFUSED`
- `Image.ANGRY`
- `Image.ASLEEP`
- `Image.SURPRISED`
- `Image.SILLY`
- `Image.FABULOUS`
- `Image.MEH`
- `Image.YES`
- `Image.NO`
- `Image.CLOCK12`, `Image.CLOCK11`, `Image.CLOCK10`, `Image.CLOCK9`, `Image.CLOCK8`, `Image.CLOCK7`, `Image.CLOCK6`, `Image.CLOCK5`, `Image.CLOCK4`, `Image.CLOCK3`, `Image.CLOCK2`, `Image.CLOCK1`
- `Image.ARROW_N`, `Image.ARROW_NE`, `Image.ARROW_E`, `Image.ARROW_SE`, `Image.ARROW_S`, `Image.ARROW_SW`, `Image.ARROW_W`, `Image.ARROW_NW`
- `Image.TRIANGLE`
- `Image.TRIANGLE_LEFT`
- `Image.CHESSBOARD`
- `Image.DIAMOND`
- `Image.DIAMOND_SMALL`
- `Image.SQUARE`
- `Image.SQUARE_SMALL`
- `Image.RABBIT`
- `Image.COW`
- `Image.MUSIC_CROTCHET`

- Image.MUSIC_QUAVER
- Image.MUSIC_QUAVERS
- Image.PITCHFORK
- Image.XMAS
- Image.PACMAN
- Image.TARGET
- Image.TSHIRT
- Image.ROLLERSKATE
- Image.DUCK
- Image.HOUSE
- Image.TORTOISE
- Image.BUTTERFLY
- Image.STICKFIGURE
- Image.GHOST
- Image.SWORD
- Image.GIRAFFE
- Image.SKULL
- Image.UMBRELLA
- Image.SNAKE