2 PRÁCE S TLAČÍTKY

Co se naučíte

- Ovládat obě programovatelná tlačítka
- Psát programy reagující na stisk tlačítka
- Význam logických spojek and a or
- Vnořené funkce

Co budete potřebovat

- PC s nainstalovaným editorem mu
- Propojovací USB kabel s micro USB koncovkou
- Micro:bit

PRŮVODCE HODINOU II-1

Co bude v této hodině potřeba:

- PC se editorem mu.
- Micro:bit s USB kabelem zakončeným micro USB. Pozor nefungují všechny kabely. Pokud budete používat jiné než koupené spolu s micro:bitem, je nutné je předem vyzkoušet.
- Pokud je k dispozici, tak dataprojektor
- Prezentaci k této lekce
- Pracovní listy pro studenty

1. krok 5 minut

Vysvětlete, že micro:bit obsahuje celkem tři tlačítka. Dvě na přední straně a jedno na zadní straně.

Tlačítko umístěné na zadní straně nás nebude v této lekci zajímat. Nelze jej programovat a slouží k resetu micro:bitu.

Na přední straně pak má dvě programovatelná tlačítka označená A a B. V MicroPythonu pro ně existují dvě proměnné button_a a button_b. Pokud studenti ví co je to objektové programování, použijte správný pojem objekty button_a a button_b. Můžete rovněž využít tuto látku k prvnímu seznámení s objektovým programováním.

2. krok 10 minut

Začněte tímto jednoduchým příkladem:

```
1. from microbit import *
2. while True:
3.     if button_a.is_pressed():
4.         display.show(Image.HAPPY)
5.     if button_b.is_pressed():
6.         display.show(Image.SAD)
7.     sleep(100)
8.     display.clear()
```

Pozor **čísla nejsou součástí programu**, slouží pouze pro možnost odkazovat se na konkrétní řádek.

Příkazy pro dotaz zda je tlačítko stisknuto jsou na řádcích 4 a 6. Nechte studenty přijít na jejich význam – test stisku tlačítka. Funkce (metoda) vrací 1 (stisknuto) nebo 0 (nestisknuto). 1 znamená splněná podmínka (True), 0 nesplněná (False). Existuje i funkce `button_a.was_pressed()`, která testuje, zda tlačítko bylo stisklé od minulého testování nebo od začátku programu.

Pozor na správné odsazení bloku ve druhé úrovni (po if) – musí být 8 mezer.

3. krok 15 minut

Nyní si vysvětlíme význam logických spojek **and** (a) a **or** (nebo). Začněte tímto příkladem:

```
1. from microbit import *
2.
3. while True:
4.     if (button_a.is_pressed())and(button_b.is_pressed()):
5.         display.show(Image.HEART)
6.     sleep(100)
7. display.clear()
```

Spojka **and** mezi dvěma testy na řádku 4 má význam a – obě podmínky musí být splněny současně.

Program nepatrně změňte (pouze na řádku 4):

```
1. from microbit import *
2.
3. while True:
4.     if (button_a.is_pressed())or(button_b.is_pressed()):
5.         display.show(Image.HEART)
6.     sleep(100)
7. display.clear()
```

Spojka **or** má význam nebo. Stačí když je splněna alespoň jedna z podmínek.

4. krok 15 minut

Funkce `get_presses()` vrací počet stisknutí daného tlačítka od startu nebo poslední kontroly. Napište a spusťte následující kód:

```
1. from microbit import *
2.
3. sleep(10000)
4. display.show(str(button_a.get_presses()))
```

Pozor na složitou konstrukci na řádku 4 a správný počet závorek. Jedná se vlastně o složenou funkci. Pokud již měli složenou funkci studenti v matematice – můžete porovnat. I zde se funkce vyhodnocují od vnitřní (dotaz na počet stisků) k vnější (zobrazení na displej).

Pokud zbyde čas, nechte studenty příklad přepsat bez složené funkce. Měli by mít něco jako:

```
1. from microbit import *
2.
3.
4. sleep(10000)
5. a = button_a.get_presses()
6. b = str(a)
7. display.show(b)
```

První zápis je kratší, druhý pro začátečníka srozumitelnější.

Neřešený závěrečný příklad: Nechte studenty naprogramovat postřehovou hru. Na Micro:bitu se bude střídavě náhodně A nebo B a hráč bude muset do určité doby stisknout odpovídající tlačítko. Hra může například skončit stiskem obou kláves současně anebo může mít pevný počet pokusů. Doba zobrazení a čekání na stisk může být konstantní nebo se může snižovat dle počtu úspěšných stisků. Na závěr může být vyhodnocení např. Procentem úspěšných pokusů. Pro volbu A nebo B použijte generátor náhodných možností z kapitoly 1.

Pokud postupujete přímo podle curicula, budete na příští hodinu potřebovat ke každému micro:bitu dva vodiče s krokodýly a nějaký hardware pro zvukový výstup.

Řekněte studentům ať si přinesou na příští hodinu sluchátka nebo repráček s jackem. Případně si připravte piezzobuzzer.

Doporučujeme, aby studenti měli sluchátka ať se vzájemně nepřehlušují rámusem. Vy si naopak připravte repráčky ať vše můžete dobře demonstrovat.

PRACOVNÍ LIST II-1

Ukázka programového větvení pomocí stisku programovatelných tlačítek A a B.

Co se naučíte

- Ovládat obě programovatelná tlačítka
- Psát programy reagující na stisk tlačítka
- Význam logických spojek and a or
- Vnořené funkce

Co budete potřebovat

- PC s nainstalovaným editorem mu
- Propojovací USB kabel micro USB koncovkou
- Micro:bit

A jděte na to …

Prohlédněte si dobře micro:bit. Zaměřte svou pozornost na tlačítka.

Kolik jich najdete a jaký je jejich význam?

Nyní zapište, odlaďte a nahrajte do micro:bitu následující příklad:

```
1. from microbit import *
9.
10. while True:
11.     if button_a.is_pressed():
12.         display.show(Image.HAPPY)
13.     if button_b.is_pressed():
14.         display.show(Image.SAD)
15.     sleep(100)
16.     display.clear()
```

Pozor **čísla nejsou součástí programu**, slouží pouze pro možnost odkazovat se na konkrétní řádek.

Pozor na správná odsazení. Odsazení na druhé úrovni (pod if) musí být o čtyři mezery oproti první úrovni, celkem tedy 8 mezer.

Které příkazy a jak testují stisk tlačítek?

Existuje i příkaz `button_a.was_pressed()` - ten vrací informaci, zda tlačítko bylo stisklé od začátku programu nebo od minulé kontroly.

Nyní si vyzkoušíte práci s oběma tlačítky současně. Odlaďte následující program:

```
1. from microbit import *
17.
18. while True:
19.     if (button_a.is_pressed())and(button_b.is_pressed()):
20.         display.show(Image.HEART)
21.     sleep(100)
22.     display.clear()
```

Co program dělá?

Jaký je význam logické spojky **and** na řádku 4?

Program nepatrně změňte na řádku 4:

```
23. 4    if (button_a.is_pressed())or(button_b.is_pressed()):
```

Jaká je změna ve funkci programu?

Jaký je tedy význam logické spojky **or**?

Nyní zapište a odlaďte následující program:

```
1. from microbit import *
24.
25. sleep(10000)
26. display.show(str(button_a.get_presses()))
```

Jaký je význam konstrukce na řádku 4?

V jakém pořadí se jednotlivé funkce vyhodnocují?

Proč je použita funkce `str()`?

Zkuste program přepsat bez vnořených funkcí?

Který zápis je kratší a který přehlednější?

Neřešený závěrečný příklad: Naprogramujte postřehovou hru. Na Micro:bitu se bude střídavě náhodně A nebo B a hráč bude muset do určité doby stisknout odpovídající tlačítko. Hra může například skončit stiskem obou kláves současně anebo může mít pevný počet pokusů. Doba zobrazení a čekání na stisk může být konstantní nebo se může snižovat dle počtu úspěšných stisků. Na závěr může být vyhodnocení např. Procentem úspěšných pokusů. Pro volbu A nebo B použijte generátor náhodných možností z kapitoly 1.

PRŮVODCE TEORIÍ

Micro:bit obsahuje celkem tři tlačítka. Tlačítko umístěné na zadní straně mezi vstupy pro USB kabel a kabel napájení je tlačítko *reset* a dále vás nebude zajímat. Na přední straně jsou umístěná dvě programovatelná tlačítka A a B. Jejich programováním a využitím se bude zabývat tato kapitola.

Začněte jednoduchým příkladem:

```
1. from microbit import *
2.
3. while True:
4.     if button_a.is_pressed():
5.         display.show(Image.HAPPY)
6.     if button_b.is_pressed():
7.         display.show(Image.SAD)
8.     sleep(100)
9. display.clear()
```

Práce s tlačítky je ukázána na řádcích 4 a 6. Jedná se vlastně o dotaz, zda tlačítko je zmáčknuté. Micropython má jak vidíte připraveny dvě proměnné button_a a button_b. Funkce button_a.is_presed() vrací 1, pokud je tlačítko stisknuté jinak vrací 0. Existuje ještě funkce button_a.was_presed(), která testuje zda tlačítko bylo stisknuté od minulé kontroly nebo od zapnutí micro:bitu.

Chcete-li testovat současný stisk obou tlačítek použijte následující konstrukce:

```
1. from microbit import *
2.
3. while True:
4.     if (button_a.is_pressed())and(button_b.is_pressed()):
5.         display.show(Image.HEART)
6.     sleep(100)
7. display.clear()
```

Mezi oběma testovacími funkcemi na řádku 4 je použita logická spojka `and`, která znamená, že celkově podmínka platí pouze pokud platí obě dílčí podmínky.

Naopak, pokud testujete, zda je stisklé libovolné tlačítko (A nebo B), použijte následující konstrukci se spojkou `or` (nebo):

```
1. from microbit import *
2.
3. while True:
4.     if (button_a.is_pressed())or(button_b.is_pressed()):
5.         display.show(Image.HEART)
6.     sleep(100)
7. display.clear()
```

Kromě uvedených funkcí `is_presed` a `was_presed`, je pro objekty `button_a` a `button_b` definována ještě funkce `get_presses()`. Tato funkce zjistí počet stisknutí tlačítka od posledního testování a nastaví jej na nulu.

Následující příklad vyčká po zapnutí (nebo stisku reset) micro:bitu deset sekund a pak zobrazí počet stisků tlačítka A od zapnutí:

```
1. from microbit import *
2.
3. sleep(10000)
4. display.show(str(button_a.get_presses()))
```

Na řádku 4 jsou do sebe vnořené tři funkce. Nejvíc vevnitř (provádí se jako první) je `button_.get.presses()`. Její výsledek pak je vstupem funkce `str()` která tento výsledek převede na řetězec a ten je pak zobrazen na displeji pomocí funkce `display.show()`. Jedná se vlastně o zkrácený zápis následující konstrukce:

```
1. from microbit import *
2.
3. sleep(10000)
4. a = button_a.get_presses()
5. b = str(a)
6. display.show(b)
```