Micro:bit – pierwsze kroki z Pythonem

Autorzy: Dawid Osuchowski, Filip Kłębczyk

Lekcja 7:

Radio

W ramach zajęć uczniowie zapoznają się z modułem radio Micro:bita. Dowiedzą się, jak wysyłać i odbierać dane oraz jakie konstrukcje stosować przy tych operacjach. Poznaną wiedzę wykorzystują do stworzenia programu – czatu obrazkowego pomiędzy dwoma płytkami.

Cele zajęć:
Uczeń powinien:

- Znać pojęcia: radio, kanał radiowy, konfiguracja, transmisja danych;
- Wiedzieć, jakie zastosowanie ma moduł radio;
- Umieć konwertować typy danych;
- Znać podstawowe funkcje biblioteki radio;
- Potrafić przesyłać i odbierać dane za pomocą fal radiowych na płytce Micro:bit w języku Python.

Pojęcia kluczowe:
radio, kanał radiowy, konfiguracja, transmisja danych

Metody pracy:

- Wykład, dyskusja, prowadzenie
- Ćwiczenia praktyczne przy komputerze
- Prezentowanie efektów pracy

Materiały pomocnicze:

- http://microbit.org/
- https://bbcmicrobitmicropython.readthedocs.io/
- https://github.com/MicrobitPolska/FunWithMicrobit

Czas na realizację zajęć: 45 min

Treści programowe (związek z podstawą programową)
Podstawa programowa z Informatyki dla szkół podstawowych, klasy VII-VIII

- I Rozumienie, analizowanie i rozwiązywanie problemów. Uczeń:
  1. formułuje problem w postaci specyfikacji (czyli opisuje dane i wyniki) i wyróżnia kroki w algorytmicznym rozwiązywaniu problemów. Stosuje różne sposoby przedstawiania algorytmów, w tym w języku naturalnym, w postaci schematów blokowych, listy kroków;
  3. przedstawia sposoby reprezentowania w komputerze wartości logicznych, liczb naturalnych (system binarny), znaków (kody ASCII) i tekstów;
  4. rozwija znajomość algorytmów i wykonuje eksperymenty z algorytmami, korzystając z pomocy dydaktycznych lub dostępnego oprogramowania do demonstracji działania algorytmów;
  5. prezentuje przykłady zastosowań informatyki w innych dziedzinach, w zakresie pojęć, obiektów oraz algorytmów.
- II Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych. Uczeń:
  1. projektuje, tworzy i testuje programy w procesie rozwiązywania problemów. W programach stosuje: instrukcje wejścia/wyjścia, wyrażenia arytmetyczne i logiczne, instrukcje warunkowe, instrukcje iteracyjne, funkcje oraz zmienne i tablice. W szczególności programuje algorytmy z działu I pkt 2;
  2. projektuje, tworzy i testuje oprogramowanie sterujące robotem lub innym obiektem na ekranie lub w rzeczywistości;
  4. zapisuje efekty swojej pracy w różnych formatach i przygotowuje wydruki;
- III Posługiwanie się komputerem, urządzeniami cyfrowymi i sieciami komputerowymi. Uczeń:
  2. rozwija umiejętności korzystania z różnych urządzeń do tworzenia elektronicznych wersji tekstów, obrazów, dźwięków, filmów i animacji;
  3. poprawnie posługuje się terminologią związaną z informatyką i technologią.
- IV Rozwijanie kompetencji społecznych. Uczeń:
  1. bierze udział w różnych formach współpracy, jak: programowanie w parach lub w zespole, realizacja projektów, uczestnictwo w zorganizowanej grupie uczących się, projektuje, tworzy i prezentuje efekty wspólnej pracy;

Przebieg zajęć:

1. Wprowadzenie w tematykę zajęć – 5 min.

Rozmawiamy z uczniami na temat komunikacji bezprzewodowej. Pytamy, jakie urządzenia odbierają informacje bezprzewodowo (np. radio FM), jakie wysyłają (nadajnik radiowy, satelitarny), a jakie wykonują obie te czynności (router bezprzewodowy, CB-radio, telefon komórkowy).

2. Część zasadnicza – 35 min.

Działanie kolejnych wersji programu będzie polegało na wysyłaniu tekstu, liczb lub obrazków na pierwszym urządzeniu, a następnie odebraniu i wyświetleniu ich na drugim urządzeniu. Uczniowie powinni dobrać się w grupy i ustalić, która z nich będzie nadawać, a która odbierać sygnały.

W tym celu skorzystamy z modułu radio i podstawowych funkcji, jakie nam ten moduł oferuje. W kodzie programu skorzystamy z następujących funkcji:

- on() – włącza radio na urządzeniu (komponent zwiększa zużycie prądu na płytce Micro:bit i wykorzystuje dodatkową pamięć);
- off() – wyłącza radio na urządzeniu (zmniejsza zużycie prądu i zwalnia pamięć);
- config() – konfiguruje parametry radia/transmisji (np. kanały);
- send() – wysyła wiadomość w postaci ciągu znaków;
- receive() – odbiera wiadomość w postaci ciągu znaków.

Nauczyciel powinien przydzielić każdej grupie unikatowy kanał (liczba z zakresu 0-83) w celu uniknięcia konfliktów w transmisji danych pomiędzy grupami. Do skonfigurowania kanału należy użyć funkcji radio.config() z argumentem channel.

Zaczynamy od prostego przykładu, tak aby zapoznać się z działaniem modułu radio – przesyłanie tekstu z jednego urządzenia, jego odbiór oraz wyświetlenie tekstu na drugim urządzeniu. Kod nadajnika będzie prezentował się następująco:

```
1  import radio
2  from microbit import *
3
4  radio.config(channel=1)
5
6  radio.on()
7
8  while True:
9      radio.send("Hello!")
```

Przez większą część trwania programu realizowana jest pętla, w ramach której cały czas wysyłamy napis *Hello!*

Z kolei kod odbiornika będzie wyglądać następująco:

```
1  import radio
2  from microbit import *
3
4  radio.config(channel = 1)
5
6  radio.on()
7
8  while True:
9      text = radio.receive()
10     if text:
11         display.scroll(text)
```

Tutaj program również będzie realizował przez cały czas pętlę, która będzie odbierała tekst. Jeśli odbiór tekstu się powiedzie, zostanie on wyświetlony z wykorzystaniem metody scroll z modułu display. Cała komunikacja pomiędzy nadajnikiem a odbiornikiem w powyższych przykładach odbywa się na kanale pierwszym.

W kolejnym przykładzie będziemy wybierać liczby z wykorzystaniem przycisku i przesyłać je na drugie urządzenie. W tym celu będziemy konwertować je do postaci tekstu. Poniżej znajduje się kod nadajnika:

```
1  import radio
2  from microbit import *
3
```

```
4   radio.config(channel = 5) # konfiguracja radia, aby wykorzystywało kanał 5-ty
5
6   number = 1
7
8   while True:
9       if button_a.was_pressed():
10          if number < 9:
11              number += 1
12          else:
13              number = 1
14      elif button_b.was_pressed():
15          radio.on() # włączenie radia
16              radio.send(str(number)) # wysłanie liczby poprzez radio
17              radio.off() # wyłączenie radia
18      display.show(number)
```

Głównym elementem programu jest pętla, w której obsługujemy zdarzenia naciśnięcia przycisków A i B. Przycisk A służy do wyboru liczby, natomiast przycisk B służy do wysłania jej na drugie urządzenie. Zastosowana metoda radio.send() przyjmuje typ danych str (ciąg znaków), dlatego musimy wcześniej przekonwertować naszą liczbę na ten typ, używając do tego funkcji str().

Kod odbiornika prezentuje się następująco:

```
1   import radio
2   from microbit import *
3
4   radio.config(channel = 5) # radio będzie wykorzystywać kanał 5.
5   radio.on() # włączenie radio
6
7   while True:
8       number = radio.receive() # nasłuchiwanie i odbiór danych
9       if number: # sprawdzenie czy coś zostało przesłane
10          display.show(number) # wyświetlenie przesłanej liczby
```

Podobnie jak w przypadku nadajnika główna część programu opiera się o pętle, w której cały czas próbujemy odbierać dane. Jeśli żadne dane nie zostały wysłane przez nadajnik, to zmienna number przyjmuje wartość pustą None. Natomiast w momencie pojawienia się danych wyrażenie w instrukcji warunkowej if staje się prawdą i w efekcie na wyświetlaczu pokazuje się przesłana liczba.

Po udanym przetestowaniu wcześniejszych wersji programów, przechodzimy do kolejnego etapu, w ramach którego będziemy wysyłać obrazki zamiast liczb. Do tego celu wykorzystamy między innymi poznaną na jednej z wcześniejszych lekcji listę obrazków. Poniżej znajduje się kod nadajnika:

```
1  import radio
2  from microbit import *
3
4  radio.config(length = 42, channel = 5) # konfiguracja radia
5  index = 0
6  images_list = [Image.HAPPY, Image.SAD, Image.ANGRY, Image.ASLEEP, Image.CONFUSED]
7
8  while True:
9      if button_a.was_pressed():
10         index += 1
11             if index == len(images_list):
12                 index = 0
13     elif button_b.was_pressed():
14         radio.on() # włączenie radia
15         radio.send(repr(images_list[index])) # wysłanie obrazka poprzez radio
16         radio.off() # wyłączenie radia
17     display.show(images_list[index])
18
```

Przy początkowej konfiguracji radia musimy zwiększyć maksymalną długość pojedynczej wiadomości w bajtach, aby móc wysyłać obrazki. W tym celu w funkcji radio.config() ustawiamy parametr length.

Funkcja repr() konwertuje obiekt obrazka do ciągu znakowego. Przykładowo dla Image.HAPPY ciąg ten prezentuje się w następujący sposób:
"Image('00000:09090:00000:90009:09990:')"

Taka konwersja jest konieczna, żeby można było skorzystać z funkcji send().

Modyfikujemy też nieznacznie kod odbiornika w porównaniu z poprzednią wersją:

```
1  import radio
2  from microbit import *
3
4  radio.config(length = 42, channel = 5)  # radio będzie wykorzystywać kanał 5-ty
5  radio.on()  # włączenie radio
6
7  while True:
8      image = radio.receive() # nasłuchiwanie i odbiór danych
9      if image: # sprawdzenie czy coś zostało przesłane
10         image = eval(image)
11         display.show(image) # wyświetlenie przesłanej liczby
```

Otrzymując obrazek w postaci ciągu znakowego po stronie odbiornika, konwertujemy go ponownie na obiekt typu Image przy pomocy funkcji eval().

3. Podsumowanie – 5 min.

Pod koniec lekcji dyskutujemy z uczniami, co było najtrudniejszym elementem zrealizowanych programów i komunikacji z użyciem modułu radio. Uczniowie prezentują efekty swojej pracy.

4. Alternatywa

Bardziej skomplikowana wersja programu zawiera połączony kod nadajnika i odbiornika, w wyniku czego otrzymujemy prosty czat obrazkowy (dwukierunkowa komunikacja).

```
import radio
from microbit import *

radio.config(length = 42, channel = 5) # konfiguracja radia
radio.on() # włączenie radio

index = 0
images_list = [Image.HAPPY, Image.SAD, Image.ANGRY, Image.ASLEEP, Image.CONFUSED]
recv_mode = False # zmienna pomocnicza, określająca początkowy tryb - nadajnik lub odbiornik

while True:
    if recv_mode: # jeżeli znajdujemy się w trybie odbiornika
        data = radio.receive()
        if data:
            image = eval(data)
            display.show(image)
            sleep(2500)
            display.show(" ")
            sleep(500)
            recv_mode = False
    else: # jeżeli znajdujemy się w trybie nadajnika
        if button_a.was_pressed():
            index += 1
            if index == len(images_list):
                index = 0
        elif button_b.was_pressed():
            radio.send(repr(images_list[index])) # wysłanie obrazka
            recv_mode = True
        display.show(images_list[index])
```

Istotnym dodatkiem powyższego programu jest zmienna recv_mode, która określa, czy urządzenie w danej chwili pełni rolę nadajnika, czy odbiornika. Urządzenia na przemian zamieniają się rolami, w efekcie czat odbywa się w trybie turowym. Otrzymany obrazek znika po 2.5 sekundy na odbiorniku, który staje się nadajnikiem i umożliwia wybranie z przewijanej listy wysyłanego obrazka.