

**Temat:** Naucz się tekstowego języka Python z Codey Rocky

- część 2 obraz i dźwięk

kl7-8 Szkoła Podstawowa

**Przedmiot:** informatyka

**Autor:** Sylwester Zasoński

**Czas trwania:** 1h lekcyjna

**Cele ogólne:**

- Rozwijanie kompetencji miękkich (umiejętność pracy zespołowej, logiczne, algorytmiczne myślenie)
- Wprowadzenie języka Python

**Cele operacyjne:**

Uczeń:

- posługuje się komputerem lub innym urządzeniem cyfrowym oraz urządzeniami zewnętrznymi przy wykonywaniu zadania
- uczeń zapoznaje się z zasadami składni języka Python
- uczeń potrafi pisać proste skrypty w języku Python

**Metody:**

praca indywidualna/zespołowa, wykład

**Środki dydaktyczne:**

1. Robot Codey Rocky + kabel/adapter do połączenia
2. Komputer z zainstalowaną aplikacją mBlock

<http://www.mblock.cc/mblock-software/>

**Przebieg zajęć:**

***Celem lekcji jest poznanie języka tekstowego Python.***

Pierwsza część skupiła się na objaśnieniu komend z kategorii **Zdarzenia**. Tym razem skupimy się na kategoriach: **głośnik, wygląd i błyskawica**.

Zacznij od dźwięków. Codey rocky dysponuje listą wgranych dźwięków. Są one zapisane w postaci plików wav. Do wyboru masz poniższe pliki:

- hello.wav : witaj
- hi.wav : cześć
- bye.wav : żegnaj
- yeah.wav : jej
- wow.wav : WOW
- laugh.wav : śmiech
- hum.wav : nucenie
- sad.wav : smutny
- sigh.wav : westchnienie
- annoyed.wav : zirytowany
- angry.wav : rozgniewany
- surprised.wav : zaskoczony
- yummy.wav : przepyszny
- curious.wav : ciekawy
- embarrassed.wav : zakłopotany
- ready.wav : gotowy
- sprint.wav : sprint
- leepy.wav : senny
- meow.wav : miauczenie
- start.wav : start
- switch.wav : przełącznik
- beeps.wav : brzęczy
- buzzing.wav : brzęczenie
- exhaust.wav : wyczerpany
- explosion.wav : eksplozja
- gotcha.wav : mam cię
- hurt.wav : boli
- jump.wav : skok
- laser.wav : laser
- level up.wav : poziom w górę
- low energy.wav : słaba bateria
- metal clash.wav : brzdęk metalu
- prompt tone.wav : przypomnienie

- right.wav : dobrze
- wrong.wav : źle
- ring.wav : dzwonek
- score.wav : wynik
- shot.wav : strzał
- step\_1.wav : krok\_1
- step\_2.wav : krok\_2
- wake.wav : pobudka
- warning.wav : ostrzeżenie

Dźwięk odtworzysz jedną z 2 komend. Pierwsza to odtworzenie dźwięku, druga to odtworzenie dźwięku i czekanie aż dźwięk skończy być odtwarzany.

`codey.speaker.play_melody('hello.wav')`

`codey.speaker.play_melody('hello.wav', True)`

Składnia komendy jest dosyć prosta `codey` - robot `speaker` - głośnik `play_melody` - odtwórz

Wszystkie dźwięki zatrzymasz komendą

`codey.speaker.stop_sounds()`

Robot Codey Rocky posiada również możliwość odtwarzania nut. Zapis nut oparty jest na zapisie użytym w Scratch. Jest to 128 nut z klawiatury MIDI. Tak więc wartości mogą oscylować w tej granicy.

Clef	Note	MIDI number	Frequency	
<b>Bas</b>	C <sub>3</sub>	48	131 Hz	
	C <sub>3</sub> /D <sub>b3</sub>	49	139 Hz	
	D <sub>3</sub>	50	147 Hz	
	D <sub>3</sub> /E <sub>b3</sub>	51	156 Hz	
	E <sub>3</sub>	52	165 Hz	
	F <sub>3</sub>	53	175 Hz	
	F <sub>3</sub> /G <sub>b3</sub>	54	185 Hz	
	G <sub>3</sub>	55	196 Hz	
	G <sub>3</sub> /A <sub>b3</sub>	56	208 Hz	
	A <sub>3</sub>	57	220 Hz	
<b>Bas i gwizdek</b>	A <sub>3</sub> /B <sub>b3</sub>	58	233 Hz	
	B <sub>3</sub>	59	247 Hz	
	<b>Bas i gwizdek</b>	<b>C<sub>4</sub> (middle C)</b>	<b>60</b>	262 Hz
	<b>gwizdek</b>	C <sub>4</sub> /D <sub>b4</sub>	61	277 Hz
		D <sub>4</sub>	62	294 Hz
		D <sub>4</sub> /E <sub>b4</sub>	63	311 Hz
		E <sub>4</sub>	64	330 Hz
		F <sub>4</sub>	65	349 Hz

	F# <sub>4</sub> /Gb <sub>4</sub>	66	370 Hz
	G <sub>4</sub>	67	392 Hz
	G# <sub>4</sub> /Ab <sub>4</sub>	68	415 Hz
	A <sub>4</sub>	69	<b>440 Hz</b>
	A# <sub>4</sub> /Bb <sub>4</sub>	70	466 Hz
	B <sub>4</sub>	71	494 Hz
	C <sub>5</sub>	72	523 Hz

Link do źródła i zapisu tu:

[https://en.scratch-wiki.info/wiki/Play\\_Note\\_\(\)\\_for\\_\(\)\\_Beats\\_\(block\)](https://en.scratch-wiki.info/wiki/Play_Note_()_for_()_Beats_(block))

A kod wygląda tak

```
codey.speaker.play_note(60, 0.25)
```

Pierwsza wartość w nawiasie to numer MIDI, a druga to długość taktu podawana w sekundach.

Istnieje również możliwość kodowania dźwięku podając jego częstotliwość. Pierwsza wartość to herc, druga to czas podany w sekundach.

```
codey.speaker.play_tone(700, 1)
```

W trakcie komponowania odgrywanej melodii możesz również użyć pauzy

```
codey.speaker.rest(0.25)
```

Wartość w nawiasie to długość pauzy w sekundach.

Programując Codey Rocky masz możliwość również ustawienia głośności w zakresie **0 ~ 100**

```
codey.speaker.volume = codey.speaker.volume + (10)
```

Powyższy kod zmienia ją o 10% w wyż, w przypadku zmniejszenia głośności wystarczy przez liczbę wstawić -

```
codey.speaker.volume = codey.speaker.volume + (-10)
```

Możesz również ustawić wartość głośności na sztywno

```
codey.speaker.volume = (100)
```

Ewentualnie użyć jej zagnieżdżając ją w inne skrypty

```
Codey.speaker.volume
```

np.

```
@event.start
```

```
def on_start():
```

```
if codey.speaker.volume > 50:
```

```
codey.speaker.play_melody('hello.wav')
```

## Codey Rocky posiada również programowalną diodę LED.

```
codey.led.show(255, 0, 0)
```

```
codey.led.show(255, 0, 0, 1)
```

3 pierwsze wartości to wartości RGB

Każda mająca zakres wartości 0 ~ 255.

R – red, czyli wartość koloru czerwonego

G – green, czyli wartość koloru zielonego

B – blue, czyli wartość koloru niebieskiego

Wartość żądanego koloru możesz poznać korzystając z linku poniżej:

[https://www.rapidtables.com/web/color/RGB\\_Color.html](https://www.rapidtables.com/web/color/RGB_Color.html)

Czwarta wartość w drugiej linijce kodu to oczywiście czas wyświetlania zapalanej diody (mierzony w sekundach).

Kolor diody ponadto może być ustawiany za pomocą

```
codey.led.set_red(255)
```

```
codey.led.set_green(255)
```

```
codey.led.set_blue(255)
```

Gdzie wartość w nawiasie to odcień koloru.

Jeżeli zdecydujesz się wyłączyć diodę głowy Codey, użyj

```
codey.led.off()
```



```
codey.display.show_image(" e00d0000000000000000000000000000")
```

Najprostszym systemem będzie zaprojektowanie wyświetlanego obrazka w sekcji Scratch i wyświetlenie go podglądzie Python.

Jeżeli jednak chcesz zgłębić temat zajrzyj na stronę:

<https://makeblock-micropython-api.readthedocs.io/en/latest/codey&rocky/codey/face%20panel.html>

Jeżeli chcesz, aby zaprojektowany obraz wyświetlał się przez określony czas potrzebujesz dodatkowy parametr `time_s=1`

```
codey.display.show_image("00003c7e7e3c000000003c7e7e3c0000", time_s=1)
```

To samo będzie się tyczyć obrazu wyświetlanego przy użyciu koordynatów X i Y

```
codey.display.show_image("00003c7e7e3c000000003c7e7e3c0000", 0, 0)
```

2 ostatnie wartości (tu 0,0) to położenie obrazku. Zasięg parametrów to:

**Oś X -15 ~ 15**

**Oś Y -7 ~ 7**

Na ekranie możesz wyświetlać również tekst i ciągi znaków

```
codey.display.show('hello')
```

```
codey.display.show('hello', wait = False)
```

```
codey.display.show('hello', 0, 0)
```

Pierwsza linijka to komenda wyświetl tekst 'hello'

Druga linijka to komenda wyświetl tekst 'hello' i czekaj

Trzecia linijka to komenda wyświetl tekst 'hello' na pozycji X i Y

Dodatkowo za pomocą kodu możesz zapalać i gasić poszczególne diody.

Wystarczy podać ich pozycję, gdzie pierwsza wartość to X o zakresie 0 ~ 15 a druga to Y o zakresie 0 ~ 7

```
codey.display.set_pixel(0, 0, True)
```

```
codey.display.set_pixel(0, 0, False)
```

Następnie mamy Boolean, czyli status diody. `True` to zapalona, `False` to zgaszona.

Diody można jeszcze przełączać pomiędzy zapalaniem i gaszeniem za pomocą kodu

```
codey.display.toggle_pixel(0, 0)
```

Oczywiście pierwsza wartość to pozycja na osi X, druga pozycja na osi Y

W kategorii związanej z wyświetlaczem znajdziesz jeszcze kod, który możesz zagnieżdżyć np. w warunkach.

```
codey.display.get_pixel(0, 0
```

Przykładowy kod może wyglądać następująco

```
@event.start
```

```
def on_start():
```

```
if codey.display.get_pixel(0, 0):
```

```
codey.led.show(255, 0, 0)
```

Jeżeli dioda w pozycji  $x=0$  i  $y=0$  czyli lewym będzie zapalona wtedy zaświeci się dioda RGB poniżej wyświetlacza.

Finalnie znajduje się jeszcze jedna komenda. Ekran wyczyścisz kodem

```
codey.display.clear()
```

Mam nadzieję, że powyższy tekst pomógł wyjaśnić komendy odpowiedzialne za wyświetlanie obrazów i danych, wyświetlanie kolorów diody RGB oraz odtwarzanie dźwięków.

W kolejnej części zajmiemy się programowaniem poruszania robotem oraz warunkami i pętlami dostępnymi w kategorii **Kontrola**.