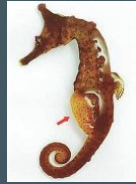


Let's play micro:bit

2019-06-19 katoy

Who am I



- 加藤洋一
- <https://qiita.com/katoy/items/9bcba54b88c5fc7d9aa6>
gem をクリーンにする。(いいね 121)
- <https://github.com/katoy/amazon-orders>
webdriver を使って amazon.co.jp での購入履歴情報を取得する
- https://github.com/katoy/scala_rational
分数 / 循環小数を扱える scala ライブラリー

今日の目的

micro:bit について

- 公式情報, 書籍の紹介
- 公式サンプルの 1 つ(磁気コンパス)を深掘りしてみた例を紹介

micro:bit とは (1/2)

- 英国の公共放送局であるBBCが中心となって開発
- 英国では2015年、11歳から12歳 (7年生) の子ども全員に無償配布！
- 個々にプログラムできる25個のLED
- 2個のボタン、リセットボタン
- 25個の外部接続
- センサー(明るさ, 温度, 加速度計, 地磁気)
- Bluetooth
- USBインターフェース (3Vのパワーで可動)



Let's play micro:bit

micro:bit とは (2/2)

- MakeCodeエディタ (ブロックまたはJavaScript)
- Python エディタ
- web 上のIDE, アプリのIDE(iOS, iPad)

MakeCode (PC, Web0)

The screenshot displays the MakeCode micro:bit editor interface. The top navigation bar includes the 'micro:bit' logo, 'Home', 'Share', 'Blocks' (selected), and 'JavaScript' tabs. On the right side of the bar are icons for help, settings, and the Microsoft logo.

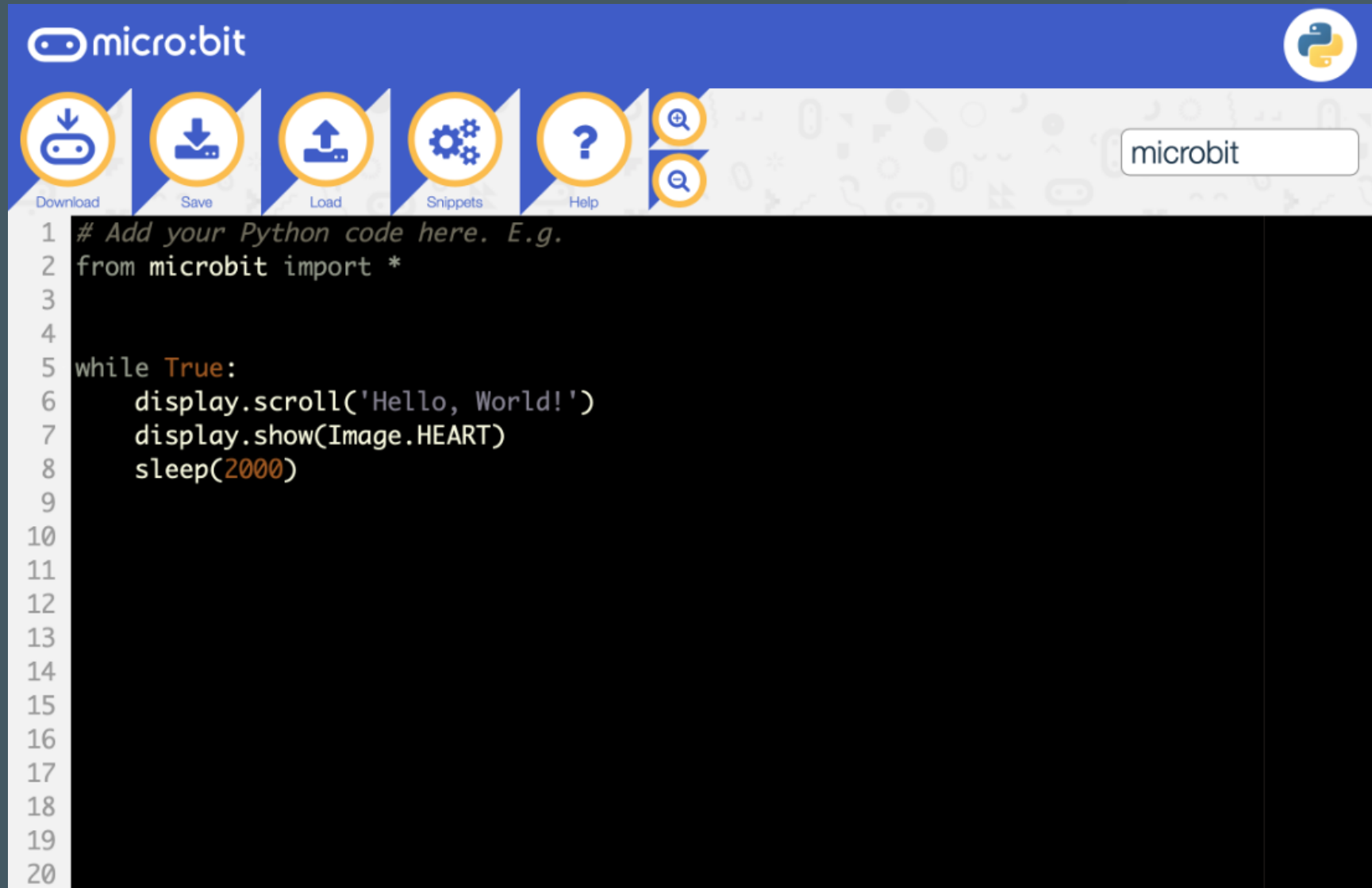
The left sidebar features a search bar and a categorized list of blocks: Basic, Input, Music, Led, Radio, Loops, Logic, Variables, Math, and Advanced. Below this is a visual representation of the micro:bit hardware with its 5x5 LED matrix displaying a pattern of red lights.

The main workspace shows a script for the micro:bit. The 'on start' block contains two 'show' blocks: 'show string "Hello!"' and 'show number 0'. Below this, an 'on shake' block is configured with the following logic:

- 'clear screen'
- 'if' block with a 'Random' block set to 2:
 - 'then' branch: 'show string "YES"'
- 'else if' block with a 'Random' block set to 1:
 - 'then' branch: 'show string "NO"'
- 'else' branch: 'show string "I DON'T KNOW"'

The script concludes with a green loop block, currently set to 'forever'.

Python エディタ



```
1 # Add your Python code here. E.g.
2 from microbit import *
3
4
5 while True:
6     display.scroll('Hello, World!')
7     display.show(Image.HEART)
8     sleep(2000)
9
10
11
12
13
14
15
16
17
18
19
20
```


iOS



Let's play micro:bit

公式情報, 書籍の紹介

- 本家サイト (日本語版もある)
<https://microbit.org/ja/guide/>
- [amazon.com](https://www.amazon.com) の kindle 無料本だけでも 5 冊ある。
(どれも平易な英語)

公式サンプルの1つを深掘り！

サンプルコード

ずっと

変数 **degrees** を **方角(°)** にする

もし **degrees** < 45 なら

矢印を表示 **上向き ↑**

でなければもし **degrees** < 135 なら **⊖**

矢印を表示 **右向き →**

でなければもし **degrees** < 225 なら **⊖**

矢印を表示 **下向き ↓**

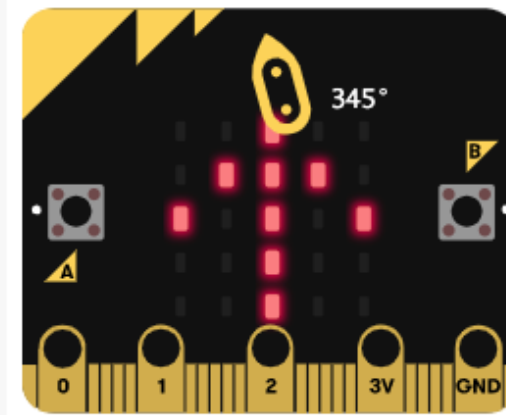
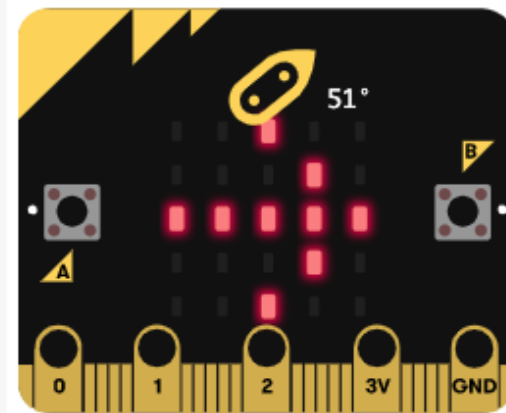
でなければもし **degrees** < 315 なら **⊖**

矢印を表示 **左向き ←**

でなければ **⊖**

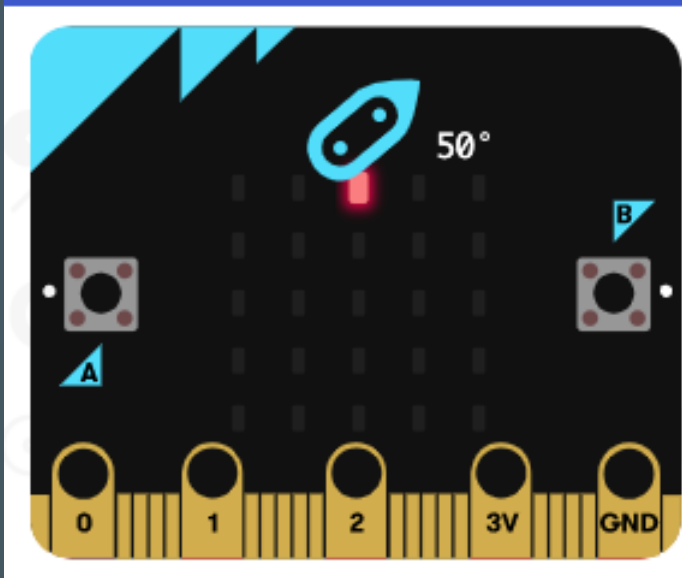
矢印を表示 **上向き ↑**

⊕



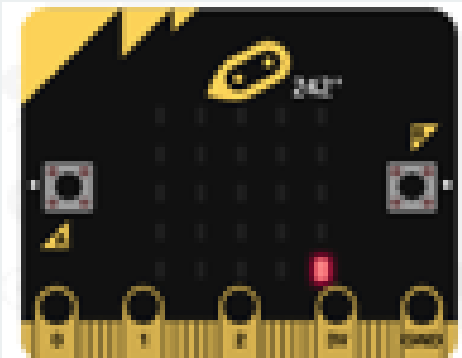
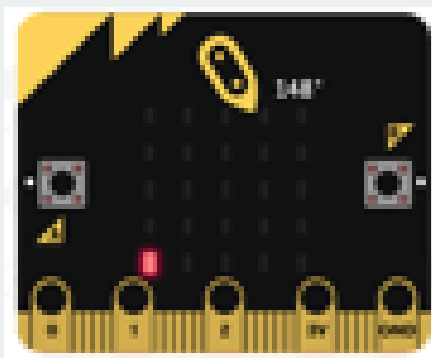
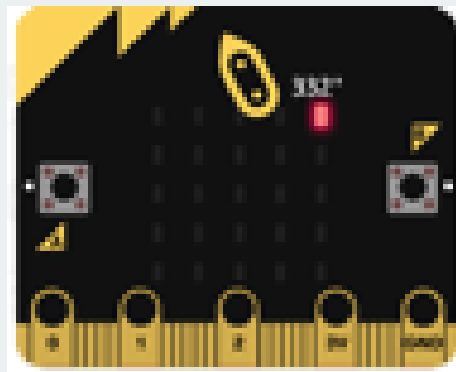
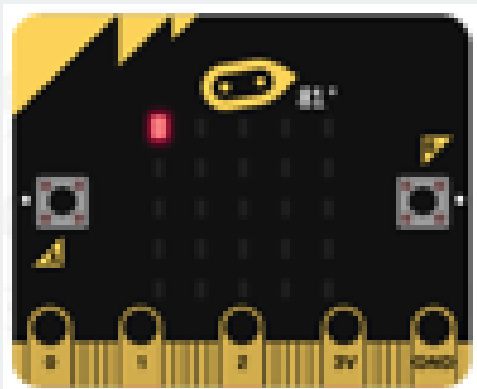
**4方向しか示せてない！
もっと分解能を高めて
みよう！**

LED 1つでコンパス



- 北向きに近くなるほどLEDの点灯間隔を点滅
- ほぼ北を向いたら、通常の点灯にする

LED 2x2 のコンパス



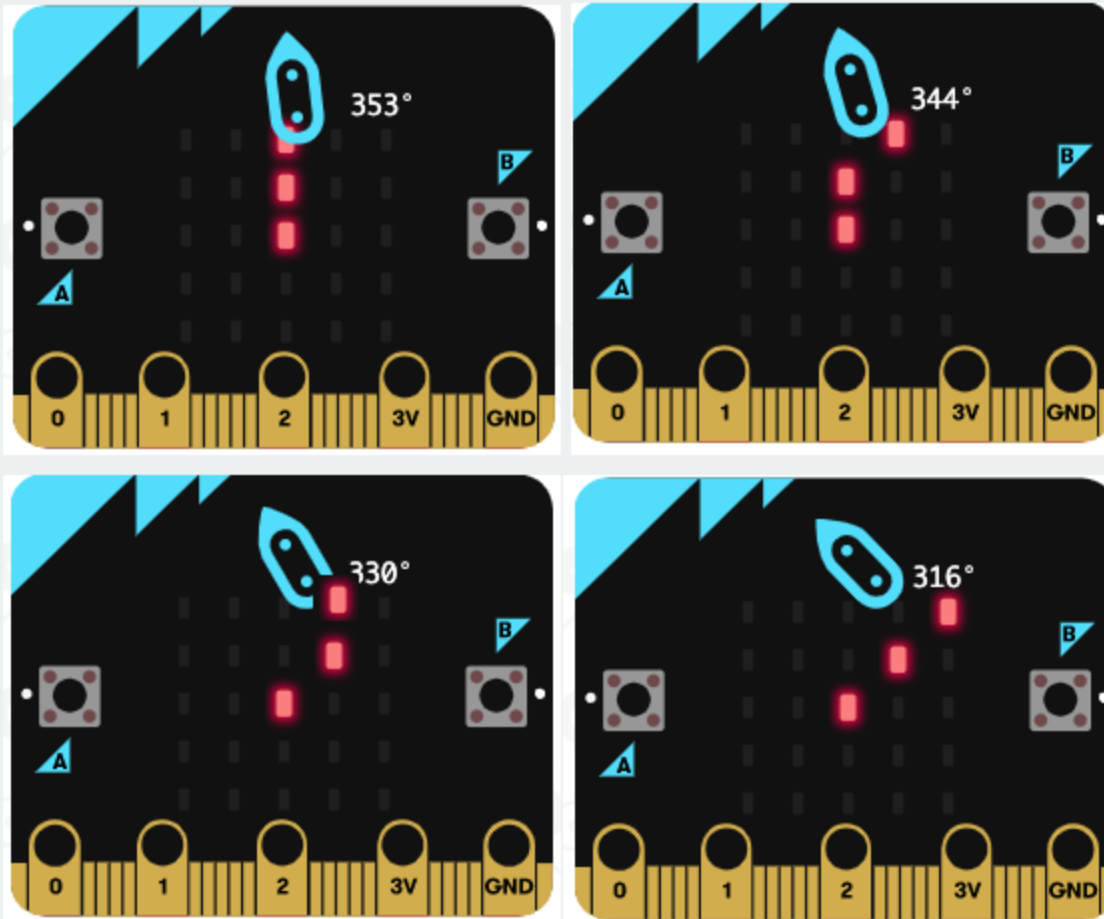
LED 3x3 のコンパス

- 省略

LED 4x4 のコンパス

- 省略

LED 5x5 のコンパス



コード

The image displays two Scratch code snippets for a micro:bit project, likely for drawing a shape using LEDs.

Left Snippet (Loop):

- ずっと (Forever) loop containing:
 - 変数 角度 を 方角(°) にする (Set angle variable to direction in degrees)
 - 変数 rad を $-90 - \text{角度} \times 3.14 \div 180$ にする (Set rad variable to $-90 - \text{angle} \times 3.14 \div 180$)
 - 変数 dx を 小数点以下四捨五入 (cos rad $\times 2.4$) にする (Set dx variable to round(cos(rad) * 2.4))
 - 変数 dy を 小数点以下四捨五入 (sin rad $\times 2.4$) にする (Set dy variable to round(sin(rad) * 2.4))
 - 変数 dx2 を 小数点以下四捨五入 (cos rad $\times 1.4$) にする (Set dx2 variable to round(cos(rad) * 1.4))
 - 変数 dy2 を 小数点以下四捨五入 (sin rad $\times 1.4$) にする (Set dy2 variable to round(sin(rad) * 1.4))
 - 一時停止(ミリ秒) 50 (Wait 50 milliseconds)

Right Snippet (Sequence):

- ずっと (Forever) loop containing:
 - 表示を消す (Clear display)
 - 変数 x を 2 にする (Set x variable to 2)
 - 変数 y を 2 にする (Set y variable to 2)
 - 変数 x1 を $x + dx$ にする (Set x1 variable to $x + dx$)
 - 変数 y1 を $y + dy$ にする (Set y1 variable to $y + dy$)
 - 変数 x2 を $x + dx2$ にする (Set x2 variable to $x + dx2$)
 - 変数 y2 を $y + dy2$ にする (Set y2 variable to $y + dy2$)
 - 点灯 x x y y (Turn on LED at x, y)
 - 点灯 x x1 y y1 (Turn on LED at x1, y1)
 - 点灯 x x2 y y2 (Turn on LED at x2, y2)
 - 一時停止(ミリ秒) 400 (Wait 400 milliseconds)
 - 反転 x x1 y y1 (Flip x1, y1)
 - 一時停止(ミリ秒) 400 (Wait 400 milliseconds)

分解能 24を実現！

わずかな工夫で 教材コードを拡張できる

Thanks!

前回: Let's Rspec

次回: Let's HaloCode

あるいは

Let's docker

このスライドは marp を使って作成しました。

原稿: <https://github.com/katoy/sample-microbit/blob/work/compass/slide/README.md>