プログラミング資料

カラーLED を点灯してみよう

氏　名＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

主催　NPO 法人 学習開発研究所
共催　大阪芸術大学 短期大学部
後援　伊丹市教育委員会

小中学生のための micro:bit を利用したプログラミング教室（第 3 回）
日　時：2025 年 3 月 23 日(日)　10:00～12:00
場　所：大阪芸術大学 短期大学部伊丹学舎　E202 教室

＜小中学生のためのプログラミング教室＞

https://u-manabi.net/ild-pkouza/

＜参考文献＞

高橋参吉、喜家村奨、稲川孝司：micro：bit で学ぶプログラミング、ブロック型から JavaScript そして Python へ、コロナ社(2019)

https://u-manabi.net/microbit/

問い合わせ先：ild-kemsyu@u-manabi.org
担当者：NPO 法人 学習開発研究所　佐藤、西野

1. micro:bit のプログラム

【例題 1-1】 ハートの表示（プログラム p-rei1-1）

「最初だけ」ブロックを利用して、ハートを表示してみよう。

「最初だけ」ブロック

＊「基本」-「LED 画面に表示」 LED をタップして、ハート形に見えるように LED を ON にする。

【例題 1-2】 ハートの点滅（プログラム p-rei1-2）

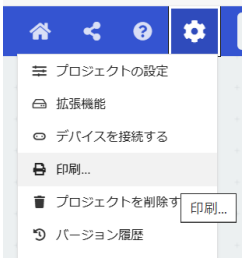
「ずっと」ブロックに変更して、ハートを点滅してみよう。

「ずっと」ブロック

＊「基本」-「一時停止」 時間を 500（ミリ秒）にする。

＊「表示を消す」

＊「基本」-「一時停止」 時間を 500（ミリ秒）にする。



<参考> プログラムの印刷

エディタの画面(右上端)にある歯車（⚙）から「印刷」を選択する。少し経過して「プログラムを印刷する」の画面が表示され、ブロックのプログラムを印刷できる。

2. プログラムの基本（順次構造、反復構造）

【例題 2-1】 アイコンの表示（プログラム p-rei2-1）

3つのアイコン（ハート、小さいダイアモンド、しかく）を表示してみよう。

「最初だけ」ブロック

＊「基本」-「アイコン表示」(ハート)、「基本」-「一時停止」

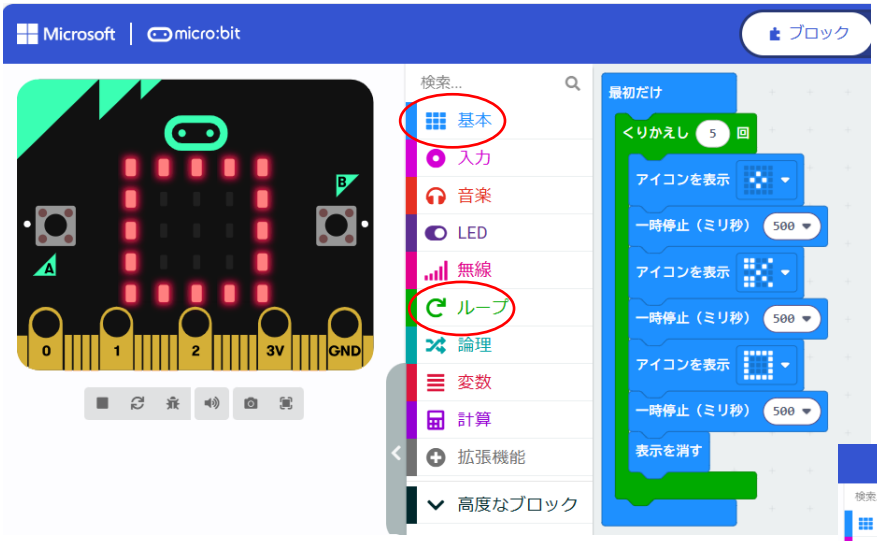
＊「基本」-「アイコン表示」(小さいダイアモンド)、「基本」-「一時停止」

＊「基本」-「アイコン表示」(しかく)、「基本」-「一時停止」

＊「基本」-「表示を消す」

【例題 2-2】 アイコンの繰り返し表示 （プログラム p-rei2-2）

アイコンを 3 つ（小さいダイアモンド、はさみ、しかく）を繰り返し 5 回表示してみよう。



「最初だけ」ブロック

＊「基本」-「アイコン表示」の「はさみ」を選択する。

＊「ループ」-「くりかえし」 数値 0 を 5 にする。

じゃんけんの「グー」「チョキ」「パー」をこの 3 つのアイコンで表示する。

反復構造は、繰り返し構造ともいう。

【例題 2-3】 カウンターの利用（プログラム p-rei2-3）

「変数 カウンター」を利用して、表示してみよう。

「最初だけ」ブロック

＊「ループ」の「変数 カウンター ～ くりかえす」 数値 0 を 4 にする。

変数「カウンター」は、「0 から始まり、「0, 1, 2, 3, 4」の 5 回繰り返す。

注）変数は数値や文字を入れるための領域

3. プログラムの基本（分岐構造）

【例題 3-1】 「グー」「パー」の表示（プログラム p-rei3-1）

変数 c=0 のときは、「グー」を表示し、c=1 ときは、「パー」を表示してみよう。

「最初だけ」ブロック

＊「変数」-「変数を追加する」 変数 c にする。

＊「変数」-「変数 c を 0 にする」

＊「論理」-「条件判断」（もし なら～ でなければ～）

＊「論理」-「くらべる」「0 ＝▽ 0」を「c ＝▽ 0」にする。

分岐構造は、選択構造ともいう。

順次構造、反復構造(繰り返し構造)、分岐構造(選択構造)をプログラムの基本構造という。

【例題 3-2】 乱数の利用（プログラム p-rei3-2）

乱数を利用して、変数 c に 0 から 1 の数値を入れるように変更してみよう。

「ずっと」ブロック

＊「変数」-「変数を追加する」

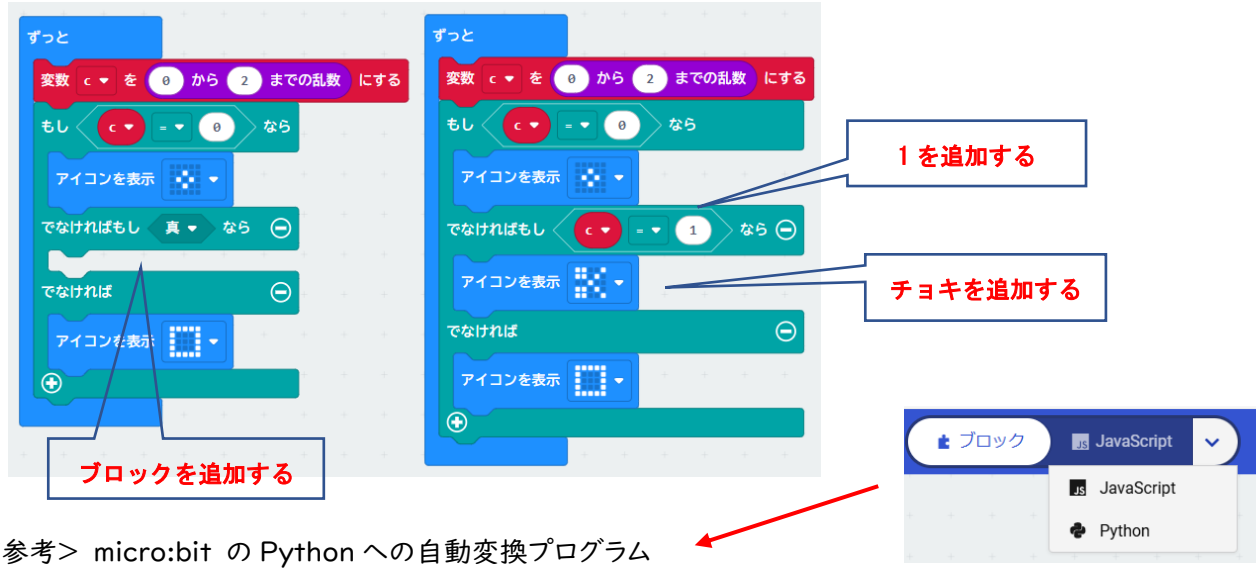
＊変数の名前を c にする。

＊「計算」-「0～10 までの乱数」 数値の 10 を 1 にする。

（乱数は 0、1 のいずれか）

【演習】「グー」「チョキ」「パー」の表示（プログラム p-rei3-2e）

例題 3-2 のプログラムを「グー」「チョキ」「パー」を表示するように変更してみよう。



<参考> micro:bit の Python への自動変換プログラム

(1) 反復構造（prei2-3）

```
for カウンター in range(5):
    basic.show_icon(IconNames.SMALL_DIAMOND)
    basic.pause(500)
    basic.show_icon(IconNames.SCISSORS)
    basic.pause(500)
    basic.show_icon(IconNames.SQUARE)
    basic.clear_screen()
```

(2) 分岐構造（prei3-1）

```
c = 0
if c == 0:
    basic.show_icon(IconNames.SMALL_DIAMOND)
else:
    basic.show_icon(IconNames.SQUARE)
```

4. LED の点灯

【例題 4-1】 光センサによる「ハート」の点滅（プログラム p-rei4-1）

光センサで明るさを調べて、暗ければ、アイコンの「ハート」を点滅してみよう。

「ずっと」ブロック

＊「論理」－「条件判断」で、「もし なら～」を選択する。

＊「論理」－「くらべる」で、「0 ＜＝ 0」 を選択する。

＊「入力」－「明るさ」を重ねる。数値 0 を 150 にする。シミュレータの明るさの数値は 128

【例題 4-2】 光センサによる LED の点灯（プログラム p-rei4-2）

光センサで明るさを調べて、暗ければ、LED を点灯してみよう。

「ずっと」ブロック

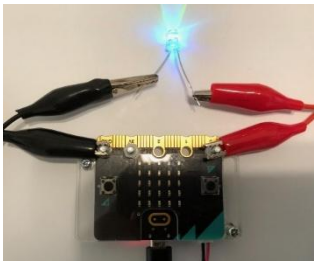
＊「論理」－「条件判断」で、「もし なら～」を選択する。

＊「論理」－「くらべる」で、「0 ＜＝ 0」を選択する。

＊「入力」－「明るさ」を重ねる。数値 0 を 125 にする。

＊「高度なブロック」で「入出力端子」選択し、

「デジタルで出力する 端子 P0 ▼ 値」を選択、数値は、1 と 0 にする。



注)図のシミュレータでは、明るさの数値が 70 で、デジタル端子 P0 の出力は 1 になっている。

【例題 4-3】 スイッチによる LED の点灯（プログラム p-rei4-3）

A ボタンを押すと、LED が点灯し、もう 1 回押すと、消灯するプログラムを作成してみよう。

「最初だけ」ブロック

＊「変数」－「変数を追加」で、変数は、s にする。

＊「変数」－「変数 s を 0 にする」

「ボタン A」

＊「論理」－「条件判断」で、「もし なら～」を選択し、「s ＝▼ 0」を重ねる。

＊「もし」ブロックでは、

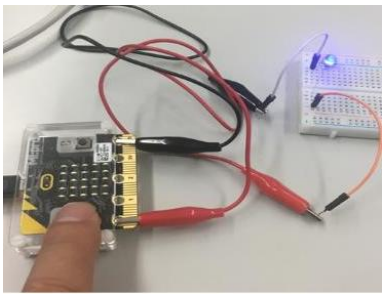
「デジタルで出力する 端子 P0 ▼ 値」の数値は 1

「変数」－「変数 s を 0 にする」で、数値は 1 にする。

＊「でなければ～」ブロックでは、

「デジタルで出力する 端子 P0 ▼ 値」の数値は 0

「変数」－「変数 s を 0 にする」で、数値は 0 にする。



<参考文献>

高橋、喜家村、稲川：micro:bit で学ぶプログラミング、
pp.35-36、p.38、コロナ社(2019)

<micro:bit V2 を初めて接続すると・・・>

パソコンの USB ケーブルで接続した（電池ボックスにつなぐ）場合には、初期プログラムが起動します。

・「HELLO」、「顔文字」が表示される。
・「←」と表示されるので、A ボタンを押す。
・「→」と表示されるので、B ボタンを押す。
・「SHAKE!」と表示されるので、micro:bit を振る。
・「TILT!」と表示されるので、中央の光を点滅する光に重なるように、micro:bit を傾けて動かす。
・「CLAP!」と表示されるので、5 回認識されるまで拍手をする。
・「WOW!」と表示され、「ハートマーク」が出れば、初期動作を確認する初期プログラムは終了する。

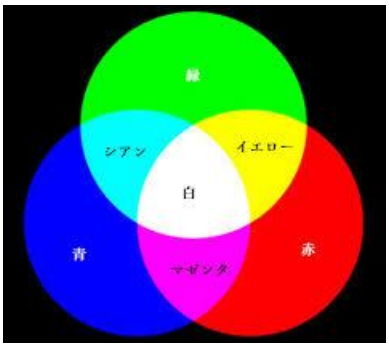
プログラムをダウンロードすると、初期プログラムは起動しません。

5．フルカラーLED の制御

【Neopixel】

Neopixel は、1 つのセルごとに赤（R）、緑（G）、青（B）の 3 つの LED と制御回路が入っており、フルカラーで光らせることができる LED の集合体である。

フルカラーは、色の明るさを、赤(R)、緑(G)、青(B)が、それぞれ 0～255 の 256 段階で選べるので、256*256*256＝16,777,216 色の表現が可能となる。なお、R:255、G:255、B:255 では白、R:0、G:0、B:0 では黒になる。

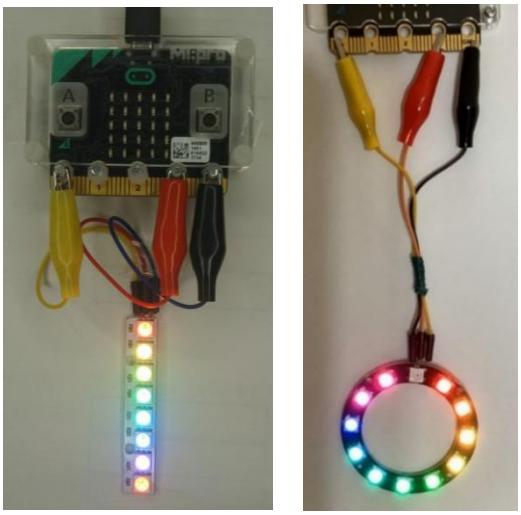


光の 3 原色　R(赤) G(緑) B(青)

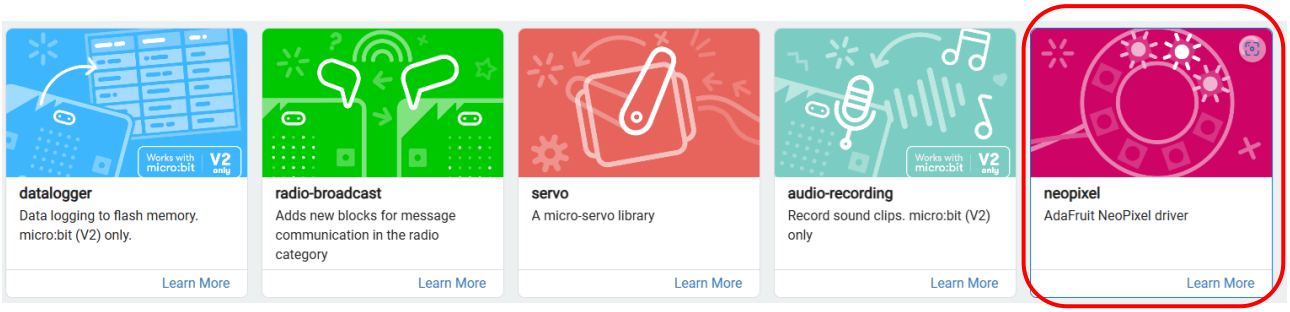
【micro:bit と Neopixel の接続】

micro:bit と Neopixel を右図のように接続する（黄色は P0 端子、赤は 3V 端子、黒は GND 端子）。

右図の Neopixel は、8 個の LED が棒状（Stick 型）に接続された製品で、その他にも、Neopixel には、リング状の製品がある。

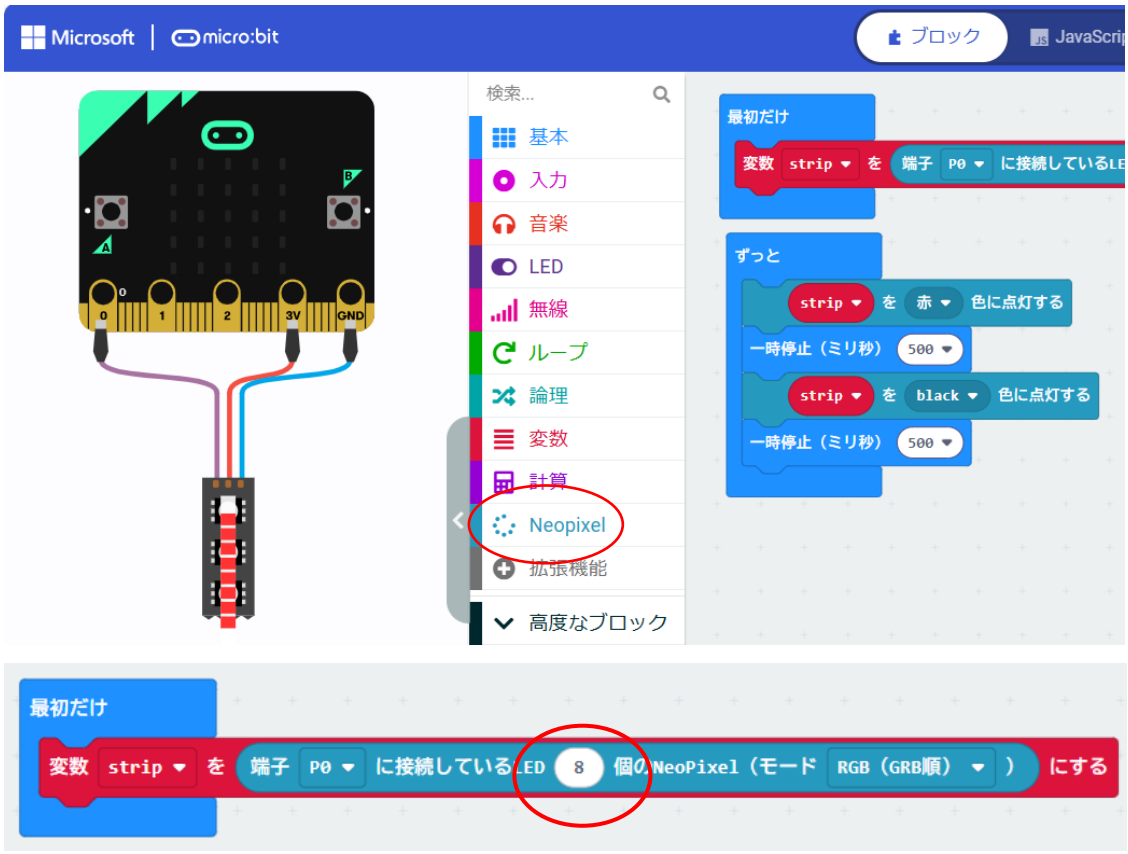


Neopixel を利用するには、ライブラリが必要である。ツールボックスの下にある「拡張機能」をクリックすると、拡張機能の一覧が表示されるので、「Neopixel」を選択する。ツールボックスの「計算」の下に、「Neopixel」のブロックが追加される。



【例題 5-1】 Neopixel の点滅（プログラム p-rei5-1）

Neopixel(Stick 型)を赤色で点滅してみよう。



「最初だけ」ブロック

＊「Neopixel」－「変数 strip を 端子 P0 に接続している LED 個… にする」

ここで、LED 24 個→ 8 個」に変更しておく。

「ずっと」ブロック

＊「Neopixel」－「strip 赤色に点灯する」 色は、赤色のままにしておく。

＊「Neopixel」－「strip black 色に点灯する」 black では、消灯になる。

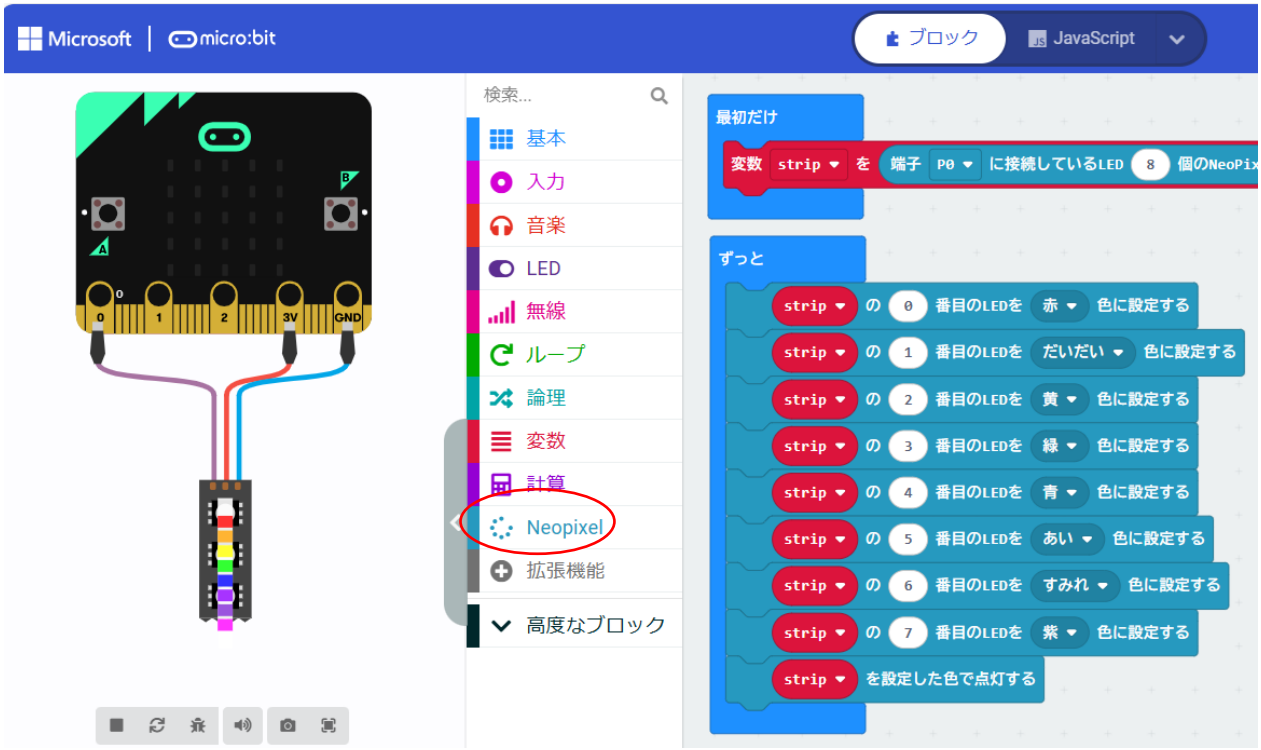
＜参考＞ フルカラー(プログラム p-rei5-1s)

「Neopixel」で、「RGB（赤 緑 青）」色に点灯する」に変更し、色はフルカラーで表現できる。

下図では、シアン（水色に近い青緑色）になる。

【例題 5-2】 Neopixel の点滅（プログラム p-rei5-2）

Neopixel を 8 色で点灯させてみよう。



「最初だけ」ブロック

＊「Neopixel」－「変数 strip を 端子 P0 に接続している LED 個… にする」

ここで、LED 24 個→ 8 個」に変更しておく。

「ずっと」ブロック

＊「Neopixel」－「strip の 0 番目 色に設定する」 0~7 番目を図の色（赤・・・紫）にする。

＊「Neopixel」－「strip で設定した色で点灯する」

【例題 5-3】 Neopixel の点滅（プログラム p-rei5-3）

Neopixel を 8 色で点灯させ、上下移動してみよう。

「最初だけ」ブロック

省略（前の例題に同じ）

「ずっと」ブロック

＊「Neopixel」－「strip の 0 番目 赤 色に設定する」

＊「ループ」－「くりかえし 8 回」にする。

＊「Neopixel」－「strip で設定した色で点灯する」

＊「Neopixel」－「strip 設定されている色を 1 個ずらす」

【例題 5−4】 2 色の上下移動）（プログラム p-rei5-4）

Neopixel を 2 色で点灯させ、2 色の上下移動してみよう。

まず、A ボタンで、赤色が上から下へ移動するように変更してみよう。

「ボタン A」

＊前の例題の「ずっと」ブロックを変更し、

「入力」−「ボタン A　が押されたときを選択する。

ブロック内のプログラムは、前の例題に同じ。

つぎに、B ボタンで、緑色が下から上へ移動するプログラムを作成するよう。

「ボタン B」

＊「strip の 0 番目　緑　色に設定する」

＊「strip で設定した色で点灯する」

＊「strip 設定されている色を LED −1 個ずらす」とする。

【演習】 エレベータの表示とシミュレーション(プログラム p-rei5-4e)

色が上下移動している Neopixel の点灯を見ていると、なんとなく、小さなエレベータの箱に見えてきました。実際のエレベータにできるだけ近い例題を考えてみよう。

<考え方>

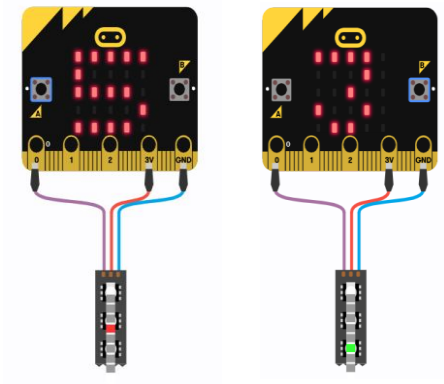
エレベータは、8 階建てマンションのエレベータで、1 階から 8 階までエレベータで移動できるものする。ただし、下記の条件で表示するプログラムを考えてみよう。

・エレベータは、最初は 1 階（もしくは、8 階）にある。

・1 階で A ボタンを押すと、1 階から上へ移動して、5 階で止まる。

・8 階で B ボタンを押すと、8 階から下へ移動して、3 階で止まる。

・上行きの移動は、赤で表示し、下行きの移動は、緑で表示する。



（参考） 1.micro:bit の特徴

micro:bit は、イギリス BBC（英国放送協会）が開発し、Micro:bit 教育財団が 7 年生（11～12 歳）の生徒を対象に無料配布した手のひらサイズの安価なコンピュータです。

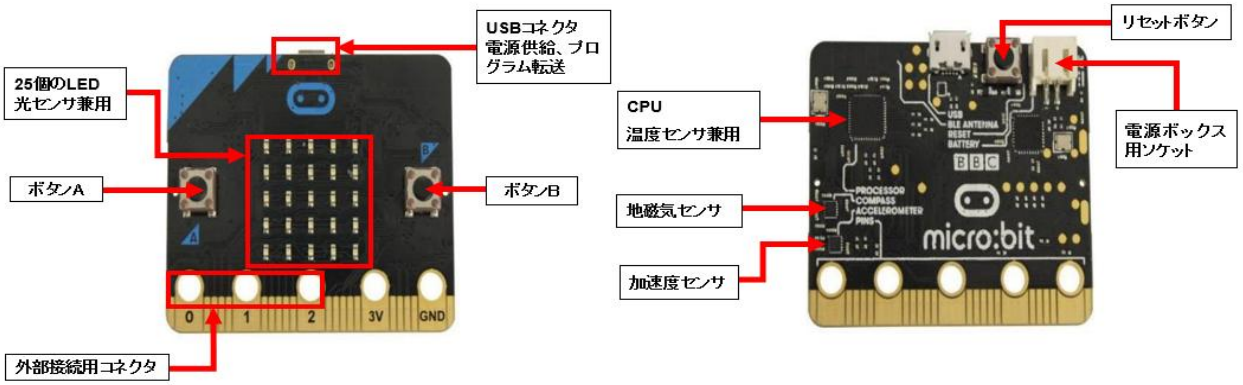


図 1-1 micro:bit（初期のバージョン）

micro:bit のハードウエア機能としては、

・25 個の LED（表示、センサ）、光、温度、加速度計などのセンサ

・プログラムができるスイッチボタン（2 個）

・Bluetooth による無線通信

・物理的に接続するための端子

などがあります。さらに、以下のような特徴があります。

・ビジュアル言語で、簡単な操作で利用できる

・シミュレータがついている

・JavaScript、Python に自動変換できる

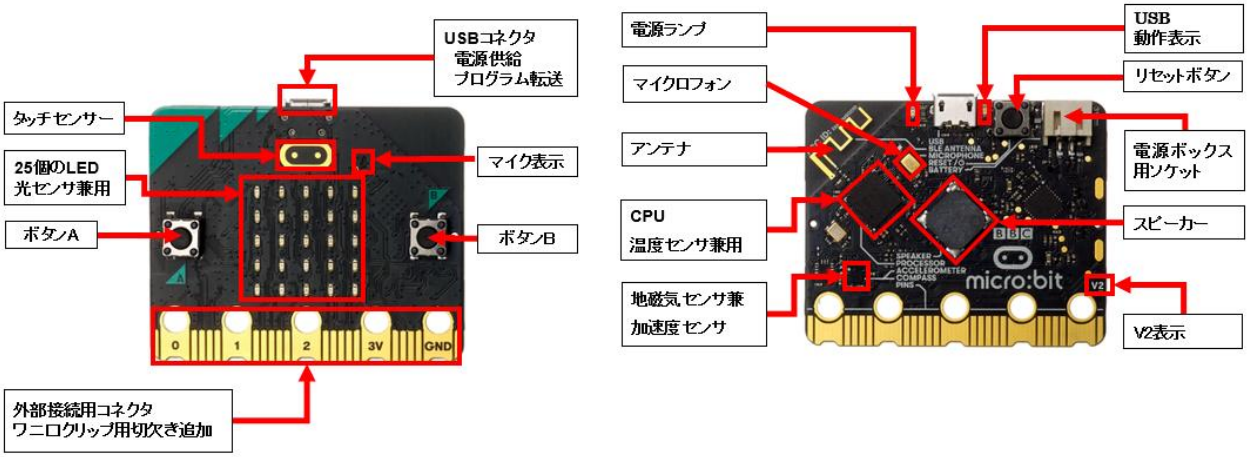


図 1-2 micro:bit V2（2021 年 8 月以降、販売されているバージョン）

【参考資料】
https://microbit.org/ja/new-microbit/
https://tech.microbit.org/hardware/

2．エディタによるプログラムの作成

インターネットに接続し、Web ブラウザで、Microsoft MakeCode for micro:bit の Wb サイトに入ると、図 2-1 のような画面（ホーム）が表示されます。ここで、「新しいプロジェクト」をタップすると、「プロジェクトを作成する」ダイアログで、プロジェクト（プログラム）の名前（ここでは、prei1-1）をつけて、「作成」をクリックします。

すると、micro:bit 用のエディタ（MakeCode）やシミュレータの機能があるシミュレータ画面が表示されます(図 2-2)。右側のプログラミングエリアには、「最初だけ」ブロックと「ずっと」ブロックが置かれています。

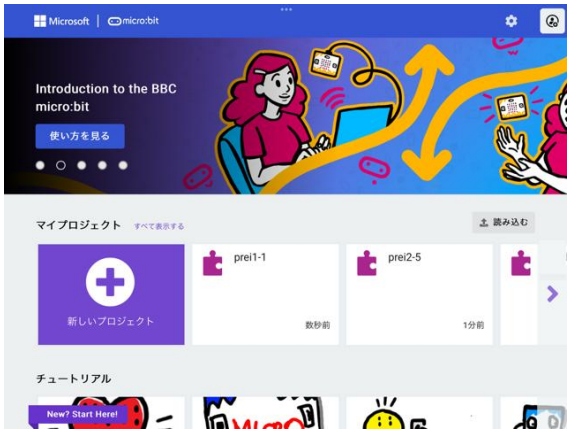


図 2-1 新しいプロジェクト

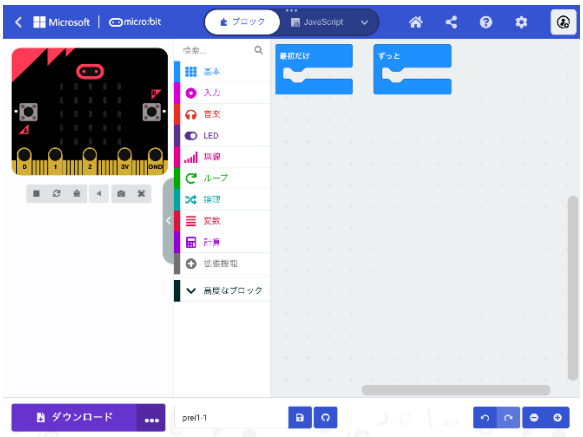


図 2-2 シミュレータ画面

図 2-3 シミュレータ画面の名称

シミュレータ画面の名称は、図 2-3 に示す通りで、それぞれの概要は、以下の通りです。

[ツールボックス]
　基本、入力、音楽、LED、無線、ループ、論理、変数、計算、そして、高度なブロックがあり、それぞれのツールをクリックすると、利用できるブロックが表示される。
[プログラミングエリア]
　ツールボックスで選択したブロックをエリア内にドロップすることによって、プログラムが作成できる。「最初だけ」「ずっと」のブロックが、最初に置かれている。
[ホーム]
　「ホーム」を選択すると、新しいプロジェクトの場合には、名前を付けることができる。最初に名前をつけていれば、最初の画面に戻る。
　注）最初に名前をつけていない場合は、「題名未設定」となる。
[ブロック]
　「ブロック」を「JavaScript」や「Python」に切り替えることによって、「ブロック」で書かれたプログラムをそれぞれの言語で表示することができる。
[ダウンロード]
　「ダウンロード」では、プログラムを micro:bit に書き込み(ダウンロード)することができる。
　注）最初は、micro:bit をパソコンへの接続やブラウザとのペアリングが求められる。
　　「…」を開くと、micro:bit への接続や切断、ファイルとしてのダウンロードなどがある。
　「FD のアイコン」では、プロジェクト名のついたプログラムをファイルとして、指定した場所（ダウンロード）に保存することができる。
[シミュレータ]
　micro:bit の画面の下にはボタンがあり、四角ボタン（■）でプログラムを停止、三角ボタン（▶）で開始できる。そのほか、再起動、デバッグモード、オーディオミュート、スクリーンショット、全画面表示のボタンがある。

　新しいプロジェクトを作成すると、プログラミングエリアには、「最初だけ」ブロックと「ずっと」ブロックが、最初に置かれています。図 2-3 では、不要なブロックである「ずっと」ブロックは、ツールボックスへ、ドラッグ&ドロップして削除しています。
　そして、ツールボックスの「基本」をタップし、「LED 画面に表示」ブロックを、ドラッグ&ドロップで、プログラミングエリアに移動し、「最初だけ」ブロックにつなげます。LED をタップ（光の ON/OFF が切り替わる）して、ハート形に見えるように LED を ON にしています。

<MakeCode エディタ>

Windows からは、下記の Web サイトにアクセスすれば、ホーム（図 2-1 に同じ）画面が表示されます。

https://makecode.microbit.org/