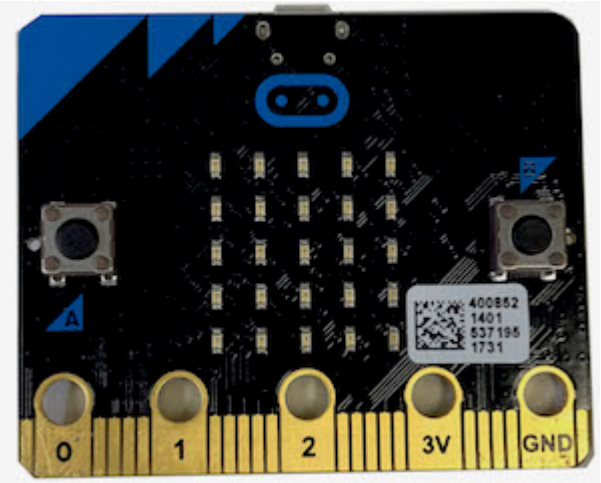
micro:bitを使ってみよう

25個のLEDや各種センサ、通信機能を持つデバイス「**micro:bit**」を使ってプログラムで操作してみよう。

micro:bitについて

micro:bitは、イギリスの公共放送局であるBBCが開発した教育用マイコンボードである。25個のLEDを使って文字列や数字の表現や、センサを使った各種計測ができる。また、外部デバイスを端子に接続し、モータなどのアクチュエータの制御なども行うことが可能である。



microbitに搭載している各部品

microbitには標準でセンサが5つ、LEDが25個、外部接続が可能な端子がある。各部品の説明を表で示す。

部品名	説明
LED	25個のLEDを点灯・消灯することで文字や数字、絵を表現できる
ボタン	ボタンの状態を計測できる
光センサ	周辺の明るさを計測できる
温度センサ	周辺の温度を計測できる
加速度センサ	加速度を計測できる。傾きや落下などの動きを検出できる
コンパス	地磁気を計測し、計測値から方角の判別ができる
端子	センサやモータなどの外部デバイスの入出力ができる

ドリトルとmicrobitの導入手順

Windowsが導入されたコンピュータでドリトルとmicrobitを通信するには、以下の設定をおこなう。このとき、プログラムのコンパイル環境のインストールを行うためOSの管理者権限が必要となる場合がある。

1. **ドリトルをインストールする**
   - ドリトルのサイト（http://dolittle.eplang.jp）のダウンロードから最新版のWindows用のドリトルをダウンロードして任意のフォルダに展開する(C:直下が好ましい)
2. **microbitの追加パッケージをダウンロードする**
   - ダウンロードから追加パッケージ「microbit制御ライブラリ」をダウンロードし、ファイルを解凍する。
   - 「microbit」フォルダ内のファイル(iniファイル,exeファイル等)を、ドリトルのルートフォルダ（dolittle.jarやdolittle.batが置いてあるフォルダ）に上書き保存する。
3. **microbitのプログラミング環境をインストールする**
   - microbit_setup.batを実行する。(インストール画面に、ファイルがありませんと表示する可能性があります。)
   - 「なにかキーを入力してください」が表示した場合「Enter」などのキーを入力する(キーを入力するとcmd.exeが閉じます。)
4. **ドリトルの起動**
   - エクスプローラからdolittle.batを実行する。

プログラムの入力から実行までの手順

手順1:パソコンとmicrobitをUSBケーブルで接続する。
手順2:プログラムを編集画面に記述する

```
システム!"microbit"使う。

最初に実行＝「
「LED!321 表示。
」

繰り返し実行=「
「LED!"hello"表示。
」
mb!転送。
```

手順3:プログラムを実行(実行ボタンを押す)
手順4:「micro:bitへの転送を実行しますか?」と表示されるので「はい(Y)」を選択する
手順5:転送が完了するまで待つ(初回実行時は、転送に10秒程度かかる。以降の実行は、2秒程度で転送が完了する)

次のように「３・２・１」が順番に表示し、

以降、次のように「hello」が何度も表示すればOKである。

microbitのプログラムの基本

ドリトルからmicrobitを制御するプログラムは、次の形で記述する。先頭の行では「microbitを使うプログラムを作成することを示している。以後「microbit本体を表す「mb」というオブジェクトを使うことができるようになる。

```
システム!"microbit"使う。
```

```
最初に実行=「
□□□□□
□□
繰り返し実行=「
□□□□□
□□
mb!転送。
```

最初に実行の「・・・」の部分には、一度だけ実行したいプログラムを書く。
繰り返し実行の「・・・」の部分には、何度も実行したいプログラムを書く。
転送を実行すると、プログラムがコンパイルされ□microbitに転送される。

文字を表示してみよう

□microbitは25個のLEDを使うことで文字や数字を表示できる。25個のLEDに相当するオブジェクト「**LED**」を使い、**表示**や**スクロール表示**を使うことで文字を表示できる。

```
システム!"microbit"使う。
最初に実行=「
□LED!321 表示。
□□
繰り返し実行=「
□LED!"end"スクロール表示。
□□
mb!転送。
```

上記のプログラムでは、最初に一度だけ実行する「最初に実行」のブロックの中に数字の「321」を表示する命令を書いている。何度も繰り返す「繰り返し実行」のブロックの中には、文字の「end□を左から右にスクロールする命令を書いている。文字は「“　”」で囲む必要がある。

LEDオブジェクトの命令一覧

命令	説明
表示	25個のLEDで英数字を表示できる。点灯する箇所を指定することもできる
スクロール表示	表示できる内容は表示命令と同じ。表示する際に左にスクロールしながら表示していく
クリア	LEDをすべて消灯する

絵を表示してみよう

□LEDでは、各々のLEDを点灯させるか消灯させるかを指定することで絵を表示することが可能である。**表示**や**スクロール表示**の引数に1と0を組み合わせた25の数字を指定する□LEDの点灯は「1」、消灯は「0」である。数字と数字の間にはスペースが必要である。

```
システム!"microbit"使う。
最初に実行=「
□□LED□1□0□1□0□1
□□□□1□0□1□0□1
□□□□1□0□1□0□1
```

```
□□□□1□0□1□0□1
        1  0  1  0  1  表示。
□□

繰り返し実行=「
□□LED□0□0□0□0□0
□□□□0□0□0□0□0
□□□□0□0□1□0□0
□□□□0□0□0□0□0
        0  0  0  0  0  スクロール表示。
□□
mb!転送。
```

プログラムを実行し、転送すると最初の一度だけ次のようになり、

以降□□LEdの点が左から右に流れるように動く。

ボタンを使ってみよう

□microbitに搭載しているボタンを使ってみる。**ボタンの押された?**を使うことで、ボタンが押された時に動作するプログラムの作成が可能である□microbitには□A□ボタンと「B□ボタンがある。それぞれ、**ボタンAオブジェクトとボタンBオブジェクトがあり、**またAとBの両方を示す**ボタンABオブジェクト**も準備している。

　ここでは、ボタンを押した時に文字を表示するプログラムを作成してみる。ボタンを使ったプログラムの場合、条件分岐の記述を行う必要がある。ドリトルでは□IF文に相当する「**!なら**」を使うことで条件分岐のプログラムを作ることができる[1)]□

```
システム!"microbit"使う。
最初に実行=「
□LED!クリア。
□□

繰り返し実行=「
  「(ボタンA!押された?)＝＝はい」！なら「
□□□□LED!"A"表示。
  」実行。

  「(ボタンB!押された?)＝＝はい」！なら「
□□□□LED!"B"表示。
  」実行。
□□
mb!転送。
```

上記のプログラムでは、最初に一度だけ実行する「最初に実行」のブロックの中にディスプレイの消灯命令を実行しており、何度も繰り返す「繰り返し実行」のブロックの中に「ボタンAを押したとき」の条件分岐と「ボタンBを押したとき」の条件分岐を実行している。ボタンAを押した場合は、ディスプレイに「A□を表示し、

ボタンBを押したときはディスプレイに「B□を表示する。

光センサ

```
システム!"microbit"使う。
最初に実行=「
□LED!クリア。
□□

繰り返し実行=「
    「(光センサ!明るさ?)>100」！なら「
□□□□LED!"A"表示。
    」そうでなければ「
□□□□LED!クリア。
    」実行。
□□
mb!転送。
```

加速度センサ

```
システム!"microbit"使う。
最初に実行=「
□LED!クリア。
□□

繰り返し実行=「
    「(加速度センサ!横の傾き?)>0」！なら「
□□□□LED!□□□□□□□□□
□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□□
          0　0　1　0　0　表示。
    」実行。
□□
mb!転送。
```

地磁気センサ

```
システム!"microbit"使う。
最初に実行=「
□LED!クリア。
□□

繰り返し実行=「
  「ボタンA!押されている?」！なら「
□□□□LED!(コンパス!角度?)表示。
  」実行。
```

□□
mb!転送。

温度センサ

```
システム!"microbit"使う。
最初に実行=「
□LED!クリア。
□□

繰り返し実行=「
  「ボタンA!押された?」！なら「
□□□□LED!(温度センサ!温度?)表示。
  」実行。
□□
mb!転送。
```

外部端子の利用

```
システム!"microbit"使う。
最初に実行=「
□LED!クリア。
□□

繰り返し実行=「
  「ボタンA!押された?」！なら「
      通信端子!(加速度センサ!横の傾き?)送信。
  」実行。

□□
mb!転送。
```

パソコン側

```
システム!"microbit"使う。
mb=シリアルポート!1024 115200 作る。
mb!(システム!シリアルポート選択)開く。
かめた=タートル!作る。
□
    かめた!10 歩く。
    向き=mb!100 読む。
    かめた!(向き/4)左回り。
」!1000 繰り返す。
mb!転送。
```

□microbitで無線通信をしてみよう

ドリトルでは無線のオブジェクトである「**無線**」を使い、「**送信**」や「**受信**」の命令を使って文字の交換やセンサの計測値の送受信が可能である。無線で送受信する機器間の設定は「**グループ**」の命令を使い、通信するグループを0〜255の間で設定することで同じグループ間で無線通信することが可能になる。

文字列の送受信

```
システム□"microbit"使う。
最初に実行＝「
    無線！0 グループ。
□□
繰り返し実行＝「
    無線□"hello" 送信。
□□
無線が受信したとき＝「
□□LED□(無線！受信) 表示。
□□
mb!転送。
```

光センサの計測値の無線通信

```
システム□"microbit"使う。
最初に実行＝「
    無線！0 グループ。
□□
繰り返し実行＝「
□□LED□(無線！数値受信) 表示。
□□
ボタンAが押されたとき＝「
    無線！(光センサ!明るさ?) 送信。
□□
mb!転送。
```

加速度センサの計測値の無線通信

```
システム□"microbit"使う。
最初に実行＝「
    無線！0 グループ。
□□
繰り返し実行＝「
    無線！(加速度センサ！横の傾き?) 送信。
□□
無線が受信したとき＝「
□□LED□(無線！数値受信) 表示。
□□
mb!転送。
```

```
システム□"microbit"使う。
最初に実行＝「
    無線！0 グループ。
□□
繰り返し実行＝「
    無線！(加速度センサ！上下加速度?) 送信。
□□
```

無線が受信したとき＝「
□□LED□(無線！数値受信) 表示。
□□
mb!転送。

1)
条件分岐のプログラムの書き方に関しては、(https://dolittle.eplang.jp/ref_basic)のページの真偽値の部分を参照してください。

From:
https://dolittle.eplang.jp/ - **プログラミング言語「ドリトル」**

Permanent link:
https://dolittle.eplang.jp/ch_microbit?rev=1549017481

Last update: **2019/02/01 19:38**