



#### Unit 2: TI-Innovator™ Hubのforループ

#### Skill Builder 2: 色をループ

このレッスンでは、いくつかのforループを使ってカラーLEDでさまざまな色を作成する混色について学習します。

#### 目標

- range()関数でforループを使用
- カラーLEDで可能な多くの色を制御
- 長いプログラムのコーディング簡略化のためコピー/貼り付け/編集を使用

TI-Innovator HubのカラーLEDを使うと、虹のすべての色(さらにそれ以上)が可能です。このレッスンでは、forループを使ってカラーLEDをさまざまな色に変更するプログラムを作成します。さまざまなループオプションを使って練習することができます。



#### 1. 新規のPython Hub Projectを開始します。

このプログラムは、LEDのカラーチャンネルを一度に1つずつ徐々に増減して、いくつかの色を混ぜ合わせます。

**step**(ステップ)値と**delay**(遅延)値を入力する2つの入力ステートメントを記述します。**step**値は**int()**にすることができますが、**delay**は0から1(10進数の値)の間である場合があるので、**float()**にします。

**step**は色の値の間のスペースであり、**delay**は各色のステップの時間遅延です。

```

1.3 1.4 1.5 ▶Unit 2 Py...ops RAD 11/36
*u2sb2.py
#-----
from ti_hub import *
from math import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#-----
step=int(input("Step? "))
delay=float(input("Delay? "))

```

#### 2. menu > Built-ins > Control (メニュー>組み込み>制御)を押して、メニューのfor index in...セクションを確認します(右図参照)。

これらのステートメントはすべて**range()**関数を使いますが、引数が異なります。3つの中で最も用途が広いのは、次のステートメントです。

### for index in range(start, stop, step):

```

4 for index in range(size):
5 for index in range(start, stop):
6 for index in range(start, stop, step):

```

#### 3. ステートメントを選択します。

### for index in range(start, stop, step):

indexを変数*i*に変更します。**start**は0、**stop**は256、**step**は**step**を入力します。(これは入力ステートメントで使われる変数であるため、stepという単語を入力する必要があります。)

これにより、ループの各ステップに**step**を追加することにより、ループを0から最大255まで移動させます。たとえば、**step**が10の場合、*i*の値は次のようになります。

0, 10, 20, 30, ..., ?

```

1.2 1.3 1.4 ▶Unit 2 Py...ops RAD 12/57
*u2sb2.py
from ti_hub import *
from math import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#-----
step=int(input("Step? "))
delay=float(input("Delay? "))
# increase red
for i in range(0,256,step):
  +=block

```



#### 4. ループblockの使用

**color.rgb(i, 0, 0)** (赤のみ)

これは、**menu > TI Hub > Hub Built in Devices > Color Output** (メニュー>TI Hub>Hub内蔵デバイス>カラー出力)にあります。

**ctrl+R**を押してプログラムを実行します。LEDがすぐに点灯することに注意します。ゆっくり点灯するはずです。**delay**(遅延)変数を使って、プロセスを少し遅くする必要があります。次のステップに進む前に、自分で試してみましょう。

```
1.2 1.3 1.4 *Unit 2 Py...ops RAD 12/62
*u2sb2.py
from math import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#=====
step=int(input("Step? "))
delay=float(input("Delay? "))
# increase red
for i in range(0,256,step):
    color.rgb(i,0,0)
```

#### 5. **sleep(delay)**ステートメントを追加しましたか。

これで、LEDは徐々に赤くなります。つぎに、別の**for**ループを記述して、赤色LEDに徐々に緑色を加えます。

赤の**for**ループをコピーしてループの下に貼り付け、テキストを編集して緑のチャンネルのみを制御します。

インデント(字下げ)に注意してください。

赤チャンネルはつねに点灯していることを確認します。

```
1.2 1.3 1.4 *Unit 2 Py...ops RAD 6/62
*u2sb2.py
from ti_plotlib import text_at,cls
from ti_system import get_key
#=====
step=int(input("Step? "))
delay=float(input("Delay? "))
# increase red
for i in range(0,256,step):
    color.rgb(i,0,0)
    sleep(delay)
```

#### 6. 緑色のループは次のとおりです。

**for i in range(0, 256, step):**  
**color.rgb(255, i, 0)**  
**sleep(delay)**

赤のチャンネルは255に設定され、ループ変数が緑の位置にあることに注意します。

プログラムを再び実行します。プログラムの最後に、何色が見えますか。

```
1.2 1.3 1.4 *Unit 2 Py...ops RAD 14/52
*u2sb2.py
from ti_system import get_key
#=====
step=int(input("Step? "))
delay=float(input("Delay? "))
# increase red
for i in range(0,256,step):
    color.rgb(i,0,0)
    sleep(delay)
for i in range(0,256,step):
    color.rgb(255,i,0)
    sleep(delay)
```

#### 7. ここで、トリッキーな部分について...**for**ループの向きを変更して、赤みを徐々に取り除きます。

**for i in range(255, 0, -step):**

このループは*i*=255で始まり、ループインデックス*i*はステップ値だけ減少します。

```
1.2 1.3 1.4 *Unit 2 Py...ops RAD 22/53
*u2sb2.py
for i in range(0,256,step):
    color.rgb(i,0,0)
    sleep(delay)
for i in range(0,256,step):
    color.rgb(255,i,0)
    sleep(delay)
for i in range(255,0,-step):
    color.rgb(i,255,0)
    sleep(delay)
```



8. 各forステートメントの前に# commentsを追加して、それが何をするのかを説明します。物事は整理しておく役立つ場合があります。コメントは#記号で始まり(ctrl+Tを押す)、プログラムの実行時には無視されます。

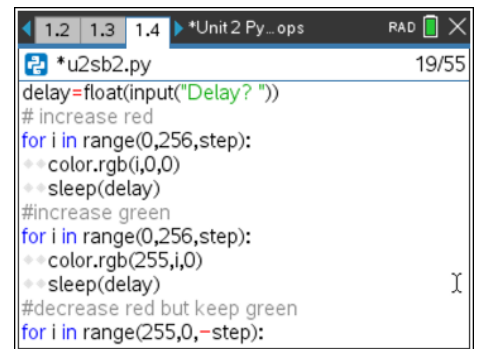
このプロジェクトを完了するには、次の場所にループを追加します。

- 青を増やす
- 緑を減らし、青は維持する
- 赤を増やす(再び)
- 青を減らす、赤は維持する
- 赤を減らしてゆっくりとオフにする

このプロセスでは、赤-緑、緑-青、青-赤の3つの色のペアすべてが混合されます。その過程で、黄色、シアン(青緑色)、マゼンタ(赤紫色)も表示されます。

プログラムの終了時に、LEDはオフになっているはずですが。

作業内容を保存することを忘れないでください。



```
1.2 1.3 1.4 *Unit 2 Py...ops RAD 19/55
*u2sb2.py
delay=float(input("Delay? "))
# increase red
for i in range(0,256,step):
    color.rgb(i,0,0)
    sleep(delay)
#increase green
for i in range(0,256,step):
    color.rgb(255,i,0)
    sleep(delay)
#decrease red but keep green
for i in range(255,0,-step):
```