少しだけスケッチのルールを知ろう

どうでしょうか、ここまで読むとdigitalWrite(LED_BUILTIN, HIGH);と、digitalWrite(LED_BUILTIN, LOW);の意味が、なんとなくわかってきたのではないでしょうか。

つまりdigitalWriteとは、指定したピンに5Vもしくは0Vを出力するように指示する言葉で、その指定先がLED_BUILTINになるのです。

3つ目のポンイントは、下から2行目と4行目にあるdelay(1000);のdelay（ディレイ）です。英語の辞書を引くと「遅らせる」とか、「延ばす」という意味で載っていますが、マイコンでは「指定した時間だけスケッチの流れを止める」という意味で使われます。

マイコンは上から下に1行ずつ順番に、スケッチの指示通りの動きをします。したがって、delayがあると、その行でスケッチの流れが止まります。このスケッチを止めている時間が、delay(1000);の1000の部分です。この数字の単位はmm（ミリ）秒で、1000mm秒は1秒になります。

delay(1000);があると、その直前のスケッチのdigitalWrite(LED_BUILTIN, HIGH);や、digitalWrite(LED_BUILTIN, LOW);が、その状態のまま1秒間止まるので、HIGHで点灯したら1秒間、LOWで消灯したら1秒間、そのままになります。

そして、4つ目のポイントは、上から4行目のvoid loop( ){（ボイドループ( ){）です。void loop( )のloop（ループ）とは英語で“輪”のことで、この言葉の後に続く{ }の中の指示を無限に繰り返えします。

もう、おわかりだと思います。HIGHで点灯し、次にLOWで消灯する動きを無限に繰り返すので、delay(1000)にすると、1秒ごとに点滅することになります。

ところで、このほかの言葉に1行目のvoid setup( )がありますが、これも同じように{ }の中がvoid setup( )の対象になります。void setup( )とvoid loop( )はArduinoのスケッチには必ず入れなければならない言葉なので、頭の隅でも入れておきましょう。

ちなみに、リスト1－1のスケッチで、/*～*/や、//に入る説明文は日本語でも構いません。スケッチだけがズラズラ並んでいると、あとから見返したときマイコンに何を指示したかったのか、そもそも何を目的に作ったのか、忘れてしまうこともあります。また、これを入れておくと、他人に見せたときに何のスケッチなのか説明しなくてもわかってもらえます。

交通信号機のスケッチ

ところで、なぜ最初にLEDを点滅させるBlinkのスケッチを紹介したのかというと、これを応用すると信号機をはじめとするさまざまなアクセサリィに利用できるからなんです。

たとえば、緑、黄、赤と順次点灯する交通信号機に使うなら、電流を流すピンを3つ指定し、順番に点灯させればいいことになります。

さっそくLEDを点滅させるスケッチを使い、交通信号機のスケッチをつくってみましょう。エッ、いきなり！と思うかもしれません。でも、じつに簡単です。コピペで数行増やし、数字を書き換えるだけですから、ほとんどゲーム感覚でできてしまいます。

まず、最初に知っておきたいことがあります。LEDに電流を流すピンを指定しなければならないのですが、さきほどのBlinkのスケッチにはどこにもピン番号ありません。たとえば、最初の3行はリスト1－4のようになっています。

リスト1－4

```
void setup( ) {

  pinMode(LED_BUILTIN, OUTPUT);
}
```

スケッチが動くよう、次のように準備しろ

LED_ビルトインに電流を流せるようにしろ

pinModeのところには、LED_BUILTIN, としか書いてありません。ここで思い出してほしいのは、Part.0でLEDの点滅実験をした際に、ボード上の「L」のLEDが点滅したことです。このとき、13番ピンを使用しても同じように点滅したことを覚えていますか（図0－20）。じつは「LED_BUILTIN,」は13番ピンと同じ働きをしているのです。つまり、LED_BUILTINはピン番号の13に置き換えることができるのです。

リスト1－5

```
void setup( ) {
```

スケッチが動くよう、次のように準備しろ

```
pinMode(13, OUTPUT);


}
```

13番ピン（LED_ビルトインと同じ）に電流を流せるようにしろ

というわけです。全部13番ピンに書き換えるとリスト１－６のアミかけしたようになります。

リスト１－６

```
void setup( ) {
  pinMode(13, OUTPUT);
}
void loop( ) {
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);
}
```

このほうが、出力ピンとしてわかりやすいですよね。

図１－１ Arduino UNO にはデジタル入出力できるピンが０番から13番まである（この他、A0～A5も利用できる）。

それでは、まずスケッチ・エディタに書かれたBlinkのスケッチを全部書き換えてしまいましょう。英語の説明文が邪魔なら、注意しながら、/*と*/の間の21行と、//以下の説明文を消しても構いません。また、書き換えるさいは、数字のあとの「,」を忘れないようにしてください。

このスケッチをもとにピンをあと２つ増やして緑、黄、赤のLEDを順番に点灯させます。点灯時間も好きなように設定できます。作例では緑15秒、黄を５秒、赤を10秒にしてみました。

ピン番号には12番と11番を加えました。方法は、pinMode(13, OUTPUT);を２回コピペして、12番、11番ピンとして書き換えます（リスト１－７）。

リスト１－７

```
void setup( ) {
  pinMode(13, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(11, OUTPUT);
}
```

これで13、12、11番ピンから出力する準備が整いました。次に、出力を指示するdigitalWriteと、点灯時間を指示するdelayも同じようにコピペして増やします（リスト１－８）。各ピンともBlinkと同じように、HIGHのあとにdelayで時間を設定し、次にLOWにします。

13番ピンの部分をまとめてコピペし、それぞれを12番ピンと11番ピンに書き換え、あとでdelayの数字だけ書き換えると手間もかかりません。

delayは緑になる13番ピンを15秒点灯させるので15000mm秒、黄の12番ピンは５秒で5000mm秒、赤の11番ピンは10秒で10000mm秒になります。

リスト１－８

```
digitalWrite(13, HIGH);
delay(15000);
digitalWrite(13, LOW);
digitalWrite(12, HIGH);
delay(5000);
digitalWrite(12, LOW);
```

```
digitalWrite(11, HIGH);
delay(10000);
digitalWrite(11, LOW);
```

どうでしょうか、思ったより簡単にできたのではないでしょうか。もしも途中で間違えてしまったら、メニューバーの「編集」に「元に戻す」があるので、ワンクリックで１つ前に戻れます。スケッチ全体はリスト１－９のようになります。

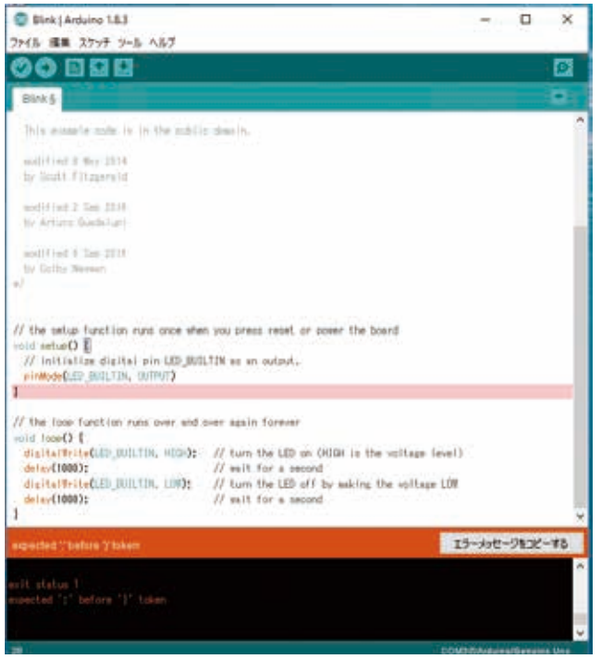
リスト１－９

```
void setup( ) {
    pinMode(13, OUTPUT);
    pinMode(12, OUTPUT);
    pinMode(11, OUTPUT);
  }
void loop( ) {
    digitalWrite(13, HIGH);
    delay(15000);
    digitalWrite(13, LOW);
    digitalWrite(12, HIGH);
    delay(5000);
    digitalWrite(12, LOW);
    digitalWrite(11, HIGH);
    delay(10000);
    digitalWrite(11, LOW);
}
```


Arduino に書き込む

スケッチが完成したら Arduino に書き込みます。書き込む方法は、Part.0で紹介したのと同じように をクリックします。このとき、スケッチを正確にコピペしていなかったり、文字を書き換えるさいに、前後の記号や文字を誤って削除していたりすると、不完全なスケッチになり、エラーメッセージが表示され、書き込めません。エラーメッセージが出たら、もう一度、スケッチをよく確認してみてください。

図１－２ スケッチが間違っているとエラーメッセージが画面の下に出る。コピペしたときに文字列の最後にある「；」や「，」を誤って消してしまうことが多いので、注意深く行いたい。



なお、ツールバーには のとなりに、スケッチの検証・コンパイルを行う のアイコンもあります。このアイコンをクリックすると、書き込む前にスケッチが正しいかどうかチェックできます。もし正しくなければ、同じ