IoTプロトタイプ制作 勉強会#2

【Arduino + フルカラーシリアルLED】

平成27年11月24日

ソフトピアジャパン　ドリーム・コア1F　ネクストコア

入出力

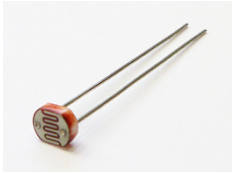
Arduinoは様々な入出力部品を取り付け動作させることが出来ます。
Arduinoでの工作は何かを入力(センシングなど)して何かを出力する(LEDを光らせるなど)パターンが多くなります。

入力例：タクトスイッチ、ボリューム抵抗(半固定抵抗)、
光センサ(CDS)、温度センサ

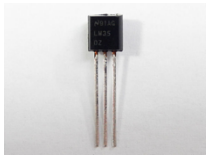
出力例：LED(単色)、ピエゾスピーカー、サーボ、フルカラーLED

タクトスイッチ

ボリューム抵抗
(半固定抵抗)



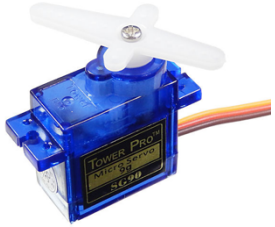
光センサ(CDS)



温度センサ

LED(単色)

ピエゾスピーカー

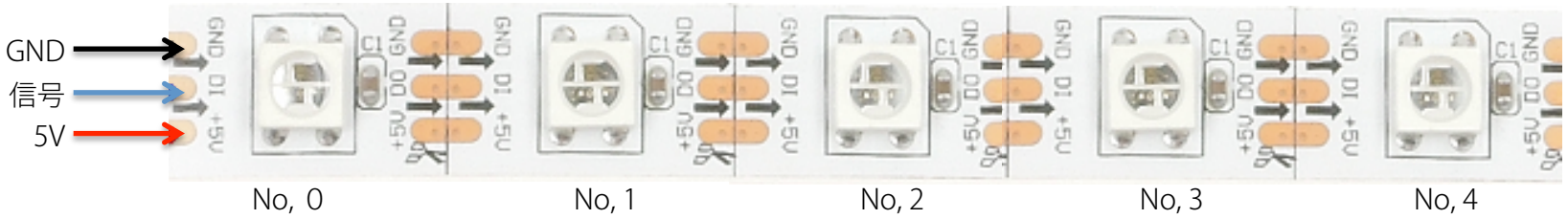


サーボ

フルカラーLED

フルカラーシリアルLEDテープ

信号線１本で制御できるフルカラーLEDテープです。１つひとつ個別に色の制御が出来ます。定格電圧も５Ｖのため Arduinoからも扱いやすいです。

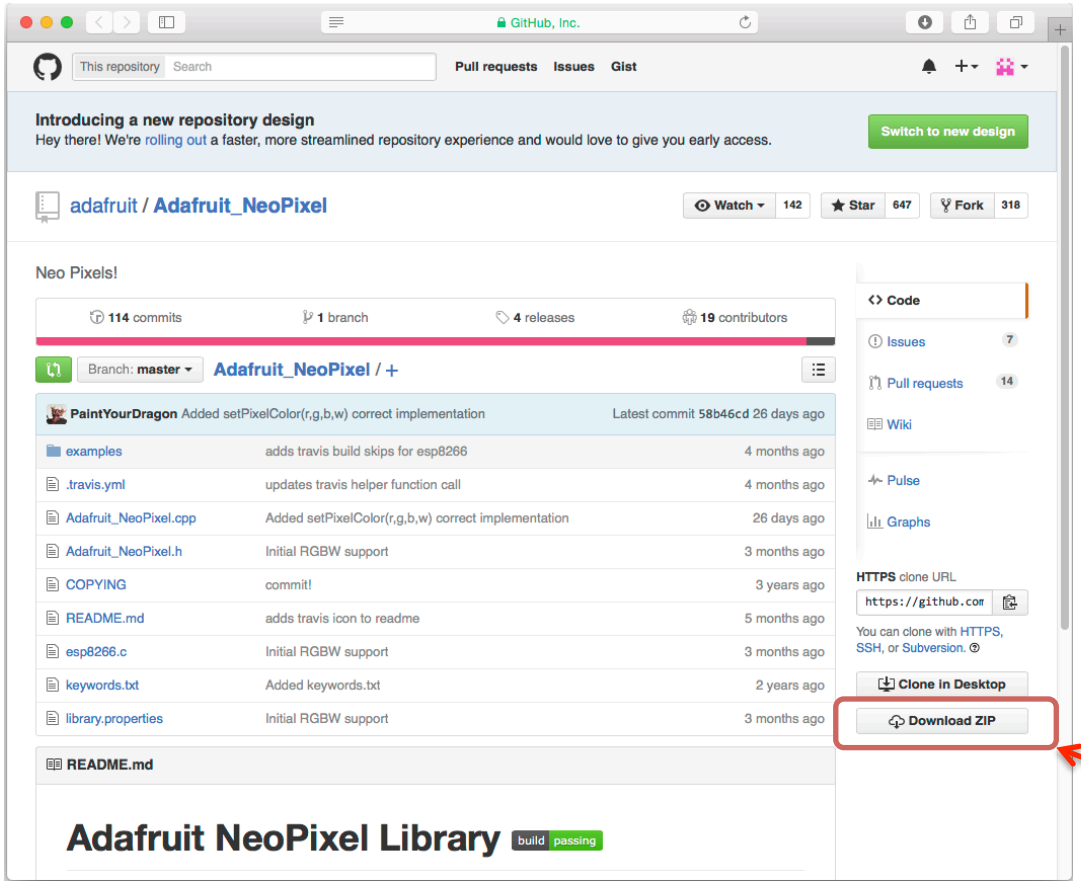


フルカラーシリアルLED (5セル分)

フルカラーシリアルLED 1m (60セル分)
https://www.switch-science.com/catalog/1399/

Adafruit NeoPixelライブラリ

デバイスメーカーのAdafruitがフルカラーシリアルLEDをArduinoで容易に扱うためのライブラリを公開しています。



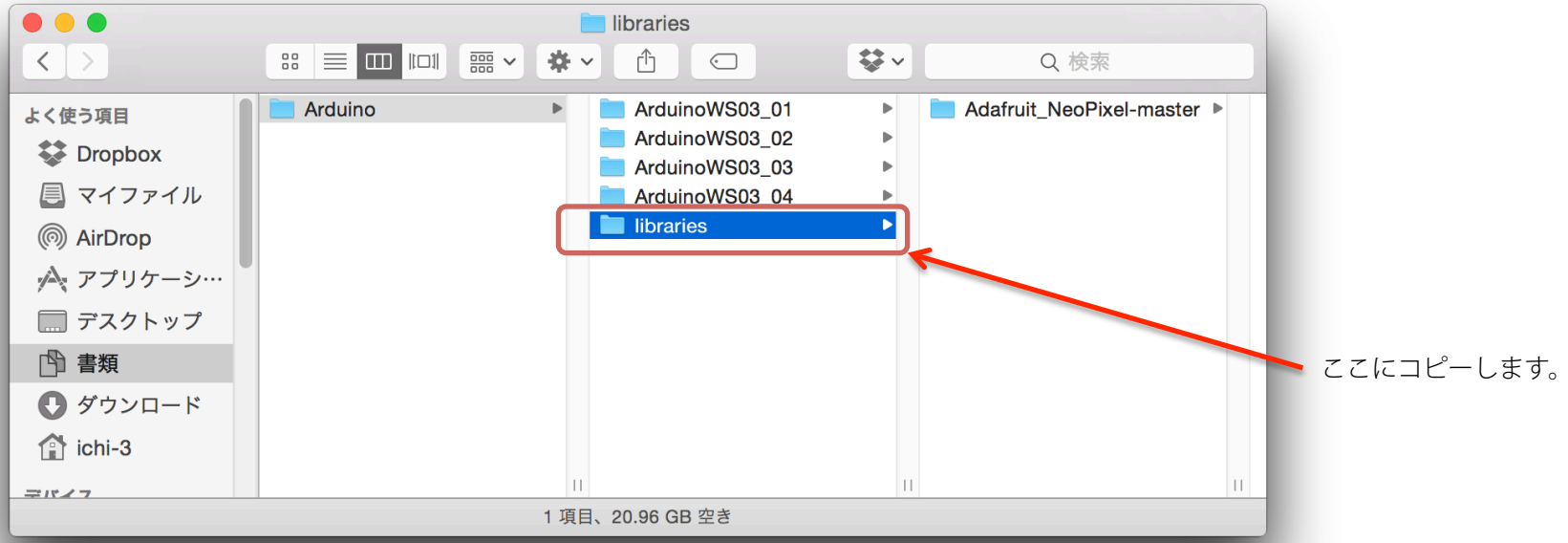
ココをクリックして
ダウンロードします。

https://github.com/adafruit/Adafruit_NeoPixel

ライブラリの導入

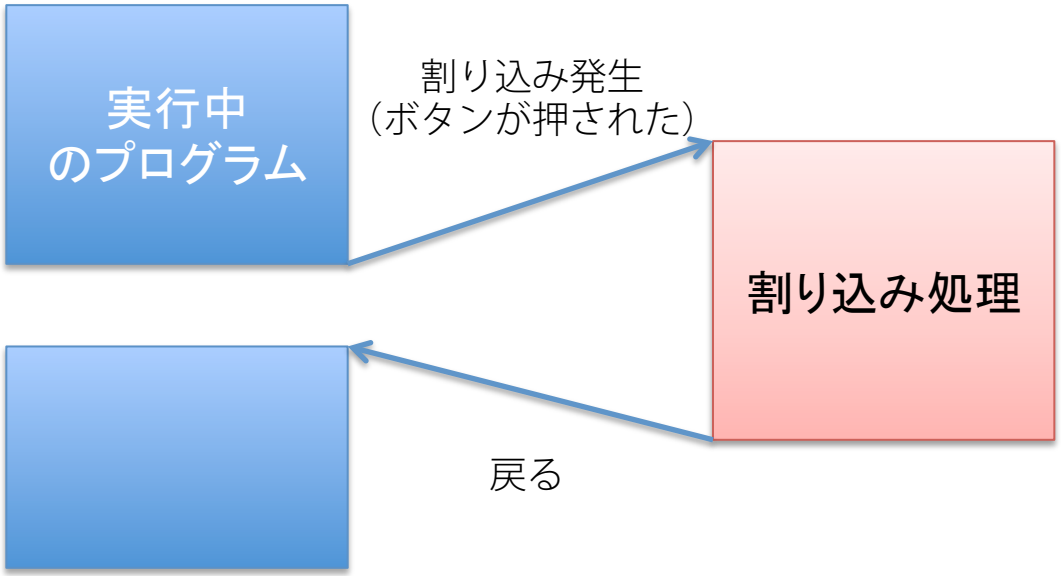
デバイスメーカー等が配布しているライブラリは「Arduinoフォルダ」直下の「libraries」フォルダにコピーします。

コピー後にArduinoIDEを再起動するとメニューの
「スケッチ」→「include Library」
に「Adafruit_NeoPixel」が追加されます。



外部割り込み処理

Arduinoのデジタルピン2番3番は入力電圧の状態変化をトリガとして関数(割り込み関数)の呼び出しを行うことが出来ます。loop()関数からピンの状態を監視するよりも確実にピンの状態変化を検出することが出来ます。今回は、ボタンが押されたことの検出に使います。

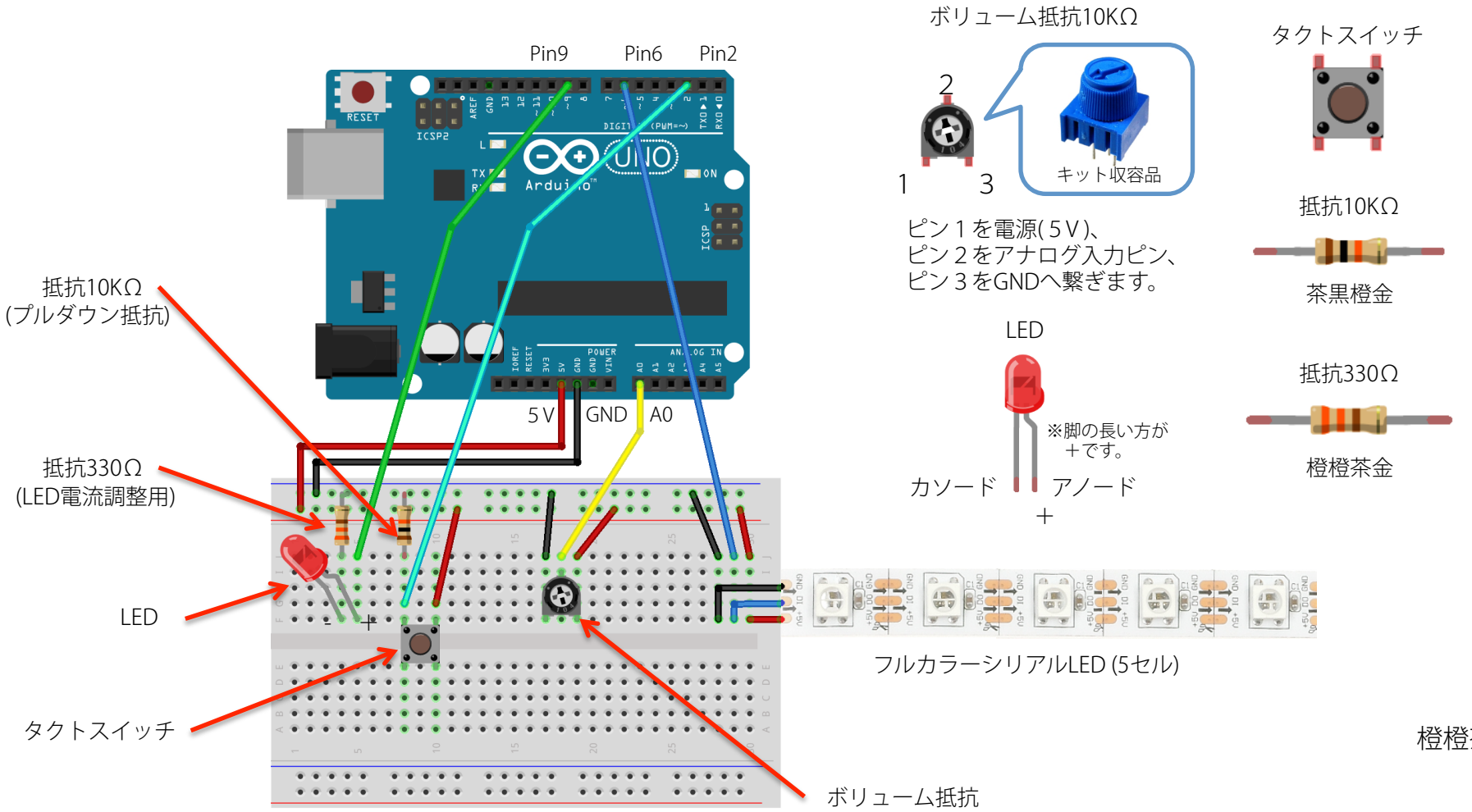


割り込み処理のイメージ

Arduino 日本語リファレンス：attachInterrupt(interrupt, function, mode)
http://www.musashinodenpa.com/arduino/ref/index.php?f=0&pos=3045

フルカラーシリアルLED:回路

入力用センサとしてボリューム抵抗とボタン(タクトスイッチ)、
出力デバイスとしてフルカラーシリアルLEDと単色LEDを接続します。



フルカラーシリアルLED:コード①

各種定義・宣言

```
#include <Adafruit_NeoPixel.h> //テープLED
#define LED_NUM 5
#define TAPE_CONTROL 6

//テープLEDライブラリのAdafruit_NeoPixelのインスタンス作成
// Parameter 1 = number of pixels in strip
// Parameter 2 = pin number (most are valid)
// Parameter 3 = pixel type flags, add together as needed:
//   NEO_RGB     Pixels are wired for RGB bitstream
//   NEO_GRB     Pixels are wired for GRB bitstream
//   NEO_KHZ400  400 KHz bitstream (e.g. FLORA pixels)
//   NEO_KHZ800  800 KHz bitstream (e.g. High Density LED strip)
Adafruit_NeoPixel strip = Adafruit_NeoPixel(LED_NUM, TAPE_CONTROL, NEO_GRB + NEO_KHZ800);

float s = 1.0;  //彩度
float v = 0.50;  //明度
int r,g,b;  //RGB値

int rgbSts[LED_NUM][3]; //各LEDのRGB値記録用配列
int setCntr;  //現在操作中のLED番号

int buttonLedPin = 9; //単色LED用ピン番号
bool interruptStat;   //割り込みステータス
```

Arduinioメニューから選択で自動で書き込まれます。
「スケッチ」→「include Library」→「Adafruit_NeoPixel」

LED 数

制御ピン番号

LED数 5 制御ピン 6を指定

Adafruit_NeoPixelクラスのインスタンスを
stripという名前で作成

彩度・明度は固定

LED数(5) × RGB値(3)の2次元配列

ボタンが押されたかどうかの判断に使用

初期化

```
void setup() {
  //現在操作中のLED番号の初期化
 setCntr=0;
  //LEDのRGB情報の初期化
 for(int i=0; i<LED_NUM; i++) {
    rgbSts[i][0] = 0;
    rgbSts[i][1] = 0;
    rgbSts[i][2] = 0;
  }
```

初期状態は0番

すべてのLEDを消灯

フルカラーシリアルLED:コード②

初期化

メイン処理の記述

```
  //テープLED制御の開始
  strip.begin();
  strip.show(); // Initialize all pixels to 'off'

  //デバッグ用シリアル通信の初期化
  Serial.begin(9600);
  Serial.print("LEDs: ");
  Serial.println(strip.numPixels());

  //割り込みを設定：割り込み番号0(Pin2),呼び出す関数 interrupt(),モード 電圧の立ち上がり
  attachInterrupt(0, interrupt, RISING);
  //割り込みフラグの初期化
  interruptStat = false;

  //単色LEDピンのピンモードを出力に設定
  pinMode(buttonLedPin,OUTPUT);
}

void loop() {
 //割り込みがあったかチェック
 if(interruptStat){
  //カウンターアップ
  setCntr++;
  //LED個数を超えていた場合0にする
  setCntr=setCntr % LED_NUM;
  //単色LEDを目印として光らせる
  digitalWrite(buttonLedPin,HIGH);
  delay(300);
  digitalWrite(buttonLedPin,LOW);
  //割り込みフラグをリセット
  interruptStat=false;
 }
```

LEDについて何か設定を行った後に、show()関数を呼ぶことで設定を反映させる

Pin2に接続されたボタンが押された時に関数interrupt()を呼び出すように設定

ボタンが押されたらカウントアップして操作対象のLEDをひとつ移動させる。

フラグを戻す

フルカラーシリアルLED:コード③

メイン処理の記述

```
  //ボリューム抵抗の読み取り
 int temp01 = analogRead(A0);
  //デバッグ用にPCへシリアルで送信
  Serial.print("A0: ");
  Serial.print(temp01);

  //読み取ったリューム抵抗の値から色相を計算
  int h = map(temp01, 0, 1024, 0, 360);
  //デバッグ用にPCへシリアルで送信
 Serial.print(" _ h: ");
  Serial.print(h);
  //HSV(色相・彩度・明度)からRGBを計算
  hsv2rgb(h,s,v);
  //デバッグ用にPCへシリアルで送信
  Serial.print(" _ r: ");
  Serial.print(r);
  Serial.print(" _ g: ");
  Serial.print(g);
  Serial.print(" _ b: ");
  Serial.println(b);

  //求めたRGB値を現在操作中のLED番号に設定
  rgbSts[setCntr][0] = r;
  rgbSts[setCntr][1] = g;
  rgbSts[setCntr][2] = b;

  //LEDの発光状態の設定
  strip.setPixelColor(setCntr, rgbSts[setCntr][0], rgbSts[setCntr][1], rgbSts[setCntr][2]);
  //設定を反映させる
  strip.show();

  delay(100);
}
```

ボリューム抵抗の値(0～1024)を取得して、色相(0°～360°)として使用する。

map()関数で0~1024を0～360に変換。

HSVからRGB値を計算する関数を呼び出す。
RGB値はグローバル変数の int r,g,b に格納されます。

ボタンを押す毎にカウントアップされるsetCntrが現在操作中のLEDを示しています。

LEDの色を設定する関数
引数は、(LED番号 , R値 , G値 , B値) です

設定＋反映で実際に色が変化します

100ミリ秒待機

フルカラーシリアルLED:コード④

ユーザ定義関数

```
/*////////////////////////////////////////////////////////////////
HSV(色相・彩度・明度)からRGBを計算する関数
RGBの戻り値はグローバル変数に記述
////////////////////////////////////////////////////////////////*/
void hsv2rgb(int h , float s , float v){
  int i , lr , lg , lb , vi;
  int p1 , p2 , p3;
  float f;
  i = (int)(h / 60.0);
  f = h / 60.0 - i;
  p1 = (int)(v * (1.0 - s) * 255.0);
  p2 = (int)(v * (1.0 - s * f) * 255.0);
  p3 = (int)(v * (1.0 - s * (1.0 - f)) * 255.0);
  vi = (int)(v * 255.0);
  if(i == 0) {lr = vi ; lg = p3 ; lb = p1;}
  if(i == 1) {lr = p2 ; lg = vi ; lb = p1;}
  if(i == 2) {lr = p1 ; lg = vi ; lb = p3;}
  if(i == 3) {lr = p1 ; lg = p2 ; lb = vi;}
  if(i == 4) {lr = p3 ; lg = p1 ; lb = vi;}
  if(i == 5) {lr = vi ; lg = p1 ; lb = p2;}
  r = lr;
  g = lg;
  b = lb;
}
```

HSVからRGBへの変換は以下のサイトを参考にしました。
- http://maicommon.ciao.jp/ss/Arduino_q/PWM/index.htm
- http://www.peko-step.com/tool/hsvrgb.html

割り込み関数

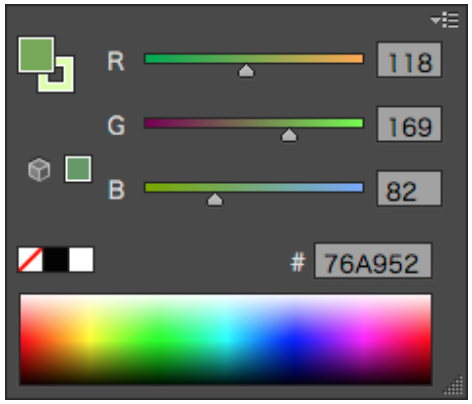
```
/*////////////////////////////////////////////////////////////////
割り込み処理で呼ばれる関数
割り込みがあったことを示すフラグを立てる
////////////////////////////////////////////////////////////////*/
void interrupt() {
  interruptStat = true;
}
```

ボタンを押すと呼ばれます

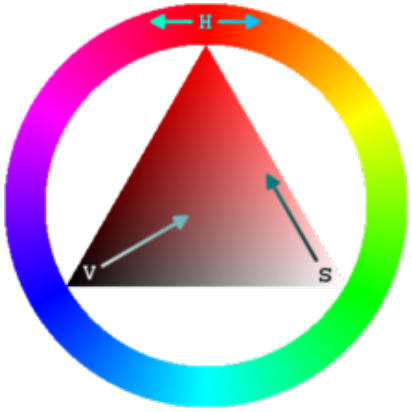
ボタンが押されたフラグ
interruptStatをtrue

補足：RGBとHSVによる色の指定

フルカラーLEDは赤(R)緑(G)青(B)三色のLEDから構成されており、その色を決めるには、三色それぞれの発光具合を指定して行います。これをRGB方式といいます。ハードウェア的な仕組みから言えば合理的な方法ですが、人の感覚ではイメージし難い方法です。
一方、色相(H)彩度(S)明度(V)で色の定義を行うHSV方式は人がイメージしやすく、連続して色の変更を行う場合などに扱いやすいものです。
そのため、本サンプルではHSVからRGBへ変換する関数を使用しています。



RGBによる色指定



HSVによる色指定

下記サイトのHSV→RGB変換を参考にしています。
http://maicommon.ciao.jp/ss/Arduino_g/PWM/index.htm
http://www.peko-step.com/tool/hsvrgb.html

IoTプロトタイプ制作 勉強会#2

【購入部品選定】

平成27年11月24日

ソフトピアジャパン　ドリーム・コア1F　ネクストコア

部品を選ぼう

よく利用される電子部品の通販サイトです。

・スイッチサイエンス
https://www.switch-science.com

Arduinoといえばここ。各種センサ、サーボも豊富、価格高め。
Raspberry Piも買えます。

・RSコンポーネンツ
http://jp.rs-online.com/web/

Raspberry Piの本家

・秋月電子通商
http://akizukidenshi.com/

電子部品一般。
ACアダプタ、ユニバーサル基板、LED、振動モータなどが豊富
オリジナルのキットが豊富(安定化電源キットなど)

部品を選ぼう

・ストロベリー・リナックス
http://strawberry-linux.com
　　電源関係(DC-DCコンバータ)、
　　ステッピングモータドライバなどのオリジナル基板が得意

・千石電商
http://www.sengoku.co.jp
　　電子部品一般。配線材、コネクタ類が秋月より豊富。

・オヤイデ電気
http://www.oyaide.com/ja
　　ケーブル専門店

・HobbyKing
http://www.hobbyking.com/hobbyking/store/index.asp
　　ラジコンパーツ店。サーボが豊富で安い。
　　ただし、海外のため送料・関税に注意。