

# 気象モニターを作ろう（発展編）

—誰にでもできるプロトタイピング—

## 第7回 センサーデータの実測と加工（基礎編の復習）

<http://cms.db.tokushima-u.ac.jp/DAV/person/S10723/気象モニターを作ろう/>

川上 博

2015/10/01

## Example07をDVDからデスクトップに ドラッグしてコピーする

1. DVDを挿入し，エクスプローラでDVDを選択
2. Lecture フォルダをダブルクリック
3. 07 フォルダをダブルクリック
4. Example07.zip フォルダをダブルクリック
5. Example07 フォルダをデスクトップにドラッグ

## ソフトの準備：

Example07を開いてスケッチを準備する

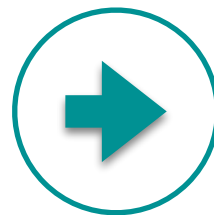
1. Example07 フォルダをダブルクリック
2. Example701 フォルダをダブルクリック
3. Example701.ino ファイルをダブルクリック

Arduino開発環境プログラムが立ち上がり、  
いつでもスケッチを実行できる状態となった

## ハードの準備：

### 気象モニター・ボックスをPCに接続する

1. 気象モニターをUSBでPCに接続する
2. Arduino開発環境のメニュー：
  - ツール→マイコンボード：Arduino Pro or ProMini
  - ツール→プロセッサ：ATmega328(3.3V,8Hz)
  - ツール→シリアルポート：COM3



スケッチを実行

# 今日のテーマ

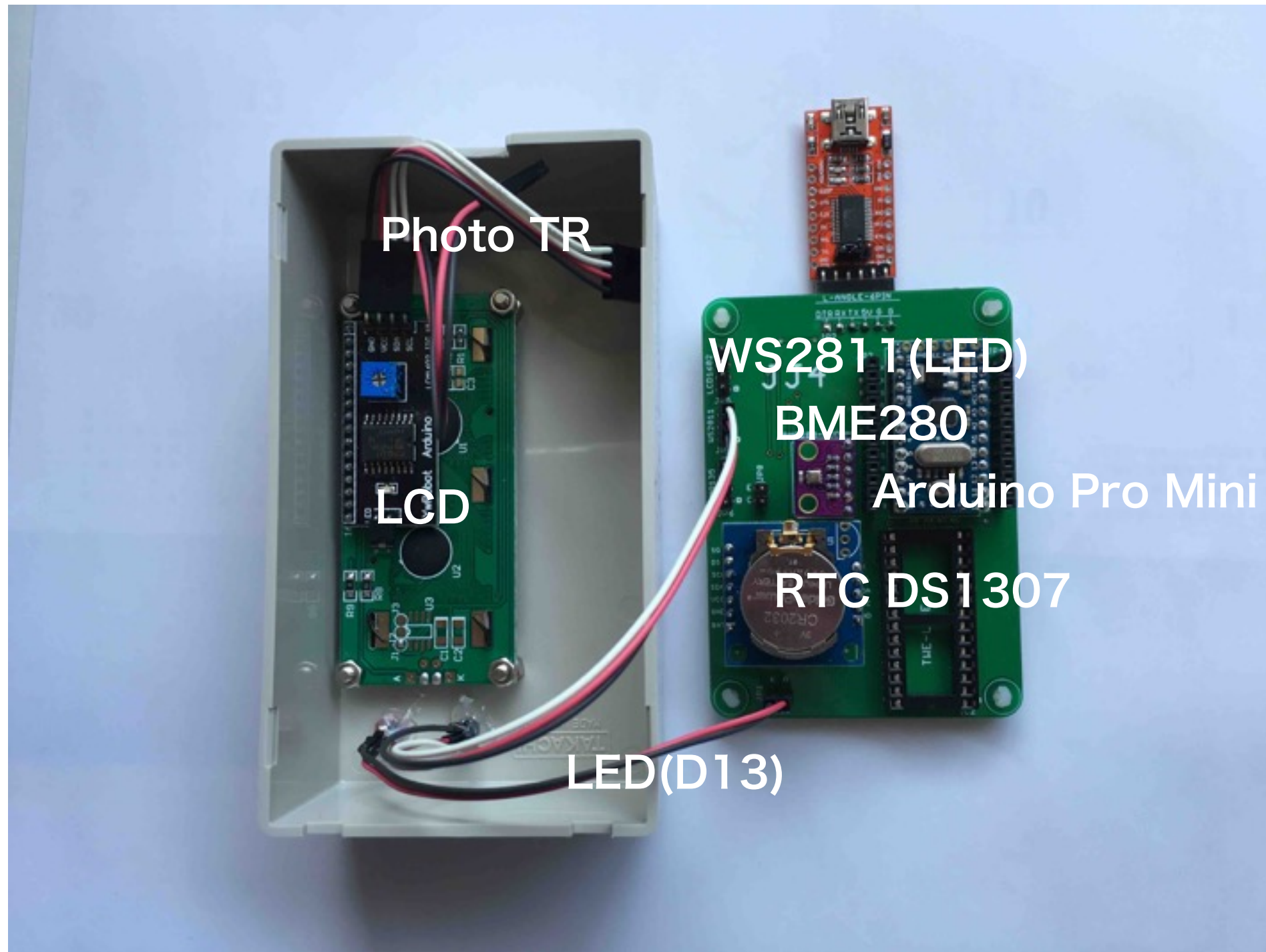
---

## 1. JJ4気象モニターの計測器とデータの表示法をみる

スケッチ：AllTest.ino(Example701.ino) を分解して理解する

## 2. Processing をインストールして使ってみる

# JJ4 での計測・表示



# AllTest.ino(Example701.ino)

---

```
// Filename: Example701.ino(AllTest.ino)
// Author: Akinori TSuji

#include "FastLED.h"
#include "SparkFunBME280.h"
#include "RTClib.h"
#include "LiquidCrystal_I2C.h"

#define LED_PIN 13
#define DATA_PIN 12
#define NUM_LEDS 1

CRGB leds[NUM_LEDS]; //FastLED pin 12
BME280 bme280; //BME280
RTC_DS1307 rtc; //DS1307
LiquidCrystal_I2C lcd(0x27, 16, 2); //LCD1602 (0x27)
```

# JJ4 気象モニター

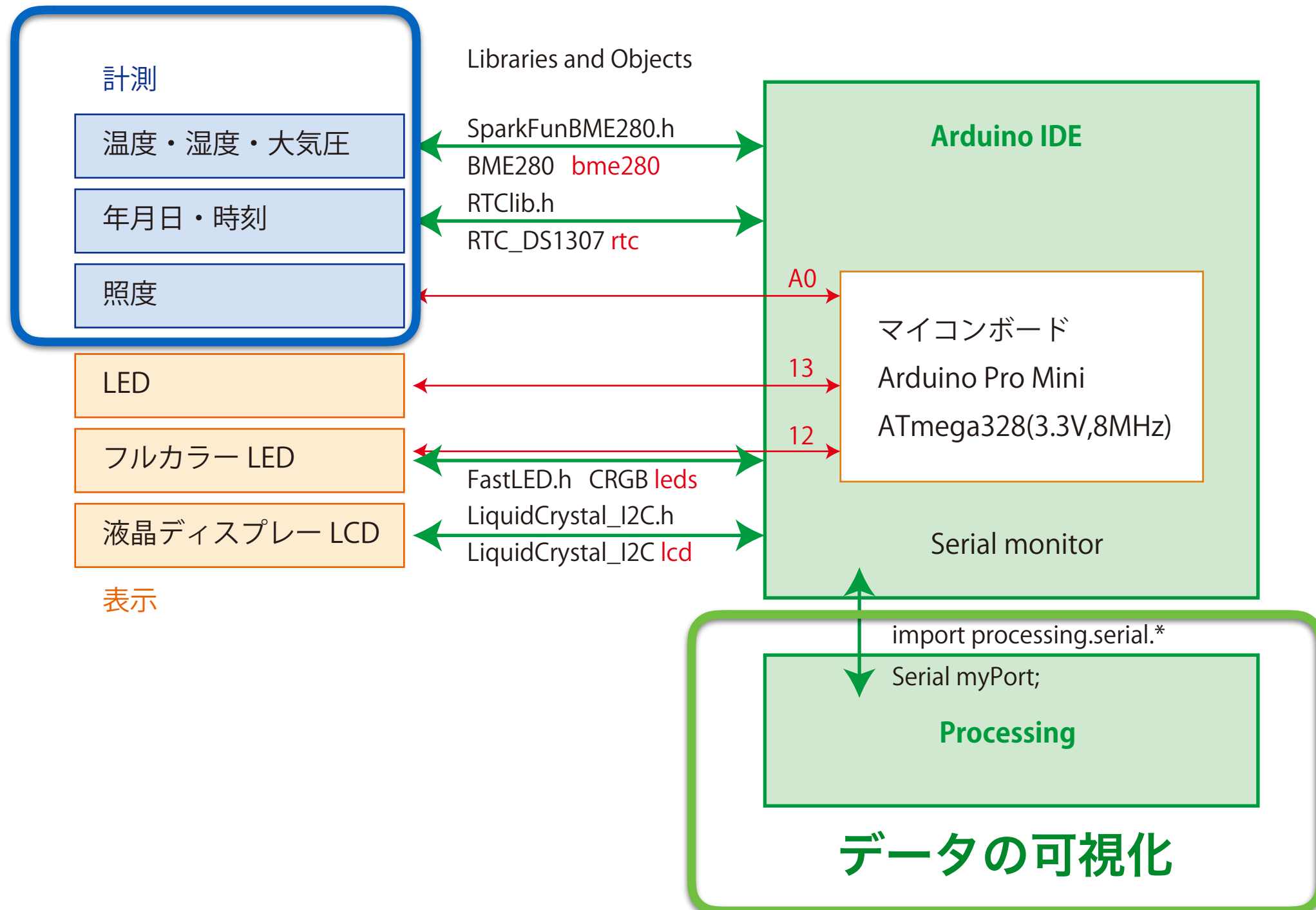
## 各種回路とその接続ポート番号

ポート	接続先	型番	ライブラリ	注釈
D0	USB シリアル変換	FT232RL		予約
D1	USB シリアル変換	FT232RL		予約
D2-D9	—	—	—	空き
D10	ZigBee 無線(親機)*1	TWE-LITE	SoftwareSerial	19200bps
D11	ZigBee 無線(親機)*1	TWE-LITE	SoftwareSerial	19200bps
D12	フルカラーLED (WS2811)	PL9823-F5	FastLED	
D13	LED(赤)5mm	OSDR5113A	digitalWrite	
A0	照度センサ	NJL7502L	analogRead	
A1	温度センサ (アナログ)	LM61CIZ	analogRead	ブレッドボード
A2-A3	—	—	—	空き
A4 (SDA) A5 (SCL)	液晶ディスプレイ 16x2	LCD1602	LiquidCrystal_I2C	I2C (5V)
A4 (SDA) A5 (SCL)	温度・湿度・大気圧センサ (デジタル)	BME280	BME280	I2C (3.3V)
A4 (SDA) A5 (SCL)	リアルタイムクロック	DS1307	RTCLib	I2C (5V)








# JJ4 での計測・表示

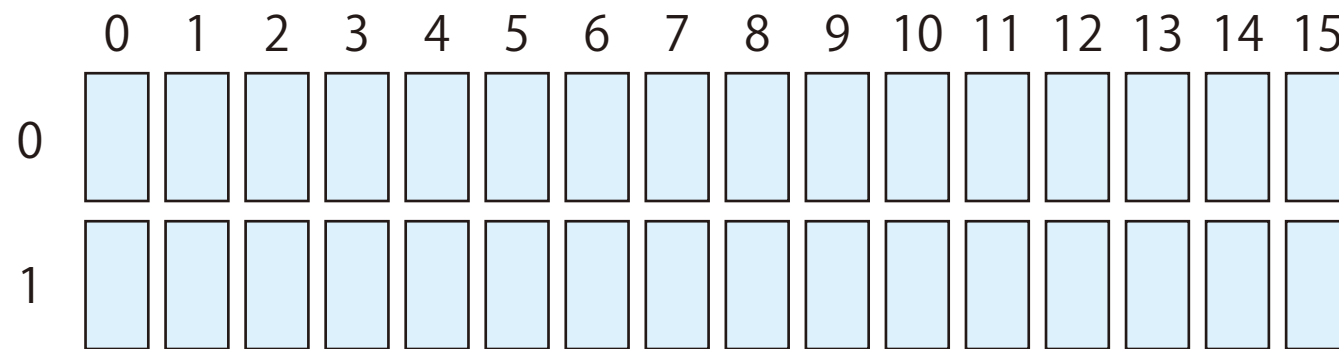
## 無線で通信



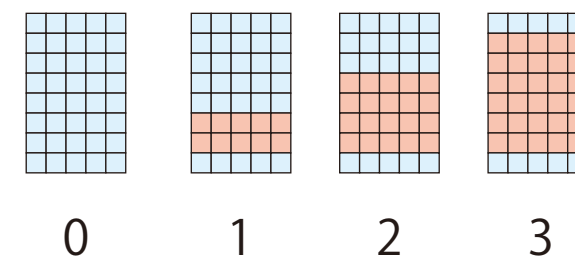
## 液晶ディスプレイのテスト・スケッチ

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0		2	8	.	6			6	8	%			1	1	0	2
1		1	6	/	1	0	/	0	1		1	1	:	1	0	

# 液晶ディスプレイ (LCD: Liquid Crystal Display)



(15, 1) を照度の大きさにより  
4段階の棒グラフに表示する



- `0x1f = 0b11111`
- `uint8_t bar3[8] = {0x0, 0x1f, 0x1f, 0x1f, 0x1f, 0x1f, 0x1f, 0x0};`
- `lcd.createChar(7, bar3);`

```

// Example702.ino(LCD00.ino)
#include "LiquidCrystal_I2C.h"
LiquidCrystal_I2C lcd(0x27, 16, 2); //LCD1602 (0x27)

// Custom character
uint8_t bar1[8] = {0x0, 0x0, 0x0, 0x0, 0x0, 0x1f, 0x1f};
uint8_t bar2[8] = {0x0, 0x0, 0x0, 0x1f, 0x1f, 0x1f, 0x1f};
uint8_t bar3[8] = {0x0, 0x1f, 0x1f, 0x1f, 0x1f, 0x1f, 0x1f};

void setup() {
  Serial.begin(9600); //---Serial
  //---LCD1602
  lcd.init(); // initialize the lcd for 16 chars 2 lines
  lcd.backlight(); // turn on backlight
  lcd.createChar(5, bar1);
  lcd.createChar(6, bar2);
  lcd.createChar(7, bar3);
  lcd.clear();
  lcd.home();
}

```

Libraryを呼んでくる

オブジェクトを定義する

文字パターン (配列) を定義する

文字パターンの辞書を作る

```

int time_cur = 0, time_last = 0;
int data=0;

void loop() {
  time_cur = millis();
  if (time_cur - time_last > 1000) {
    time_last = millis();
    data = analogRead(A0); // read photo sensor
    illum(data);
  }
}

```

1秒(1000ms)ごとに実行する仕事を書く

```

void illum(int data) {
  lcd.clear();
  lcd.setCursor(15, 1);
  int pos = map(data, 0, 1025, 0, 4);
  switch (pos) {
    case 0:
      lcd.print(" ");
      break;
    case 1:
      lcd.write(5);
      break;
    case 2:
      lcd.write(6);
      break;
    case 3:
      lcd.write(7);
      break;
    default:
      break;
  }
}

```

文字パターンを表示する関数

millis()

Returns the number of milliseconds since the Arduino board began running the current program.

This number will overflow (go back to zero), after approximately 50 days.

# スケッチの基本形

```
// Example702.ino(LCD00.ino)
```

```
#include "LiquidCrystal_I2C.h"   ライブラリを呼んでくる
```

```
LiquidCrystal_I2C myLCD(0x27, 16, 2); オブジェクトの定義
```

```
int time_cur, time_last;       大域変数の定義
```

```
void setup() {
```

```
    myLCD.init();  
    myLCD.backlight();  
    myLCD.clear();
```

スケッチの初期化

```
// initialization
```

```
}
```

```
void loop() {
```

```
    time_cur = millis();  
    if (time_cur - time_last > 1000) {  
        time_last = millis();
```

```
        // main program   実行したい仕事
```

```
    }
```

```
}
```

1000msec毎に  
main program  
を実行する



年月日・時刻のテスト・スケッチ

```
// Example703.ino(RealTimeClock00.ino)
```

```
#include "RTCLib.h"
```

Libraryを呼んでくる

```
#include "LiquidCrystal_I2C.h"
```

オブジェクトを定義する

```
RTC_DS1307 rtc; //DS1307
```

```
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

```
// Custom character
```

文字パターンを定義する

```
uint8_t heart1[8] = {0x0, 0xa, 0x1f, 0x1f, 0xe, 0xe, 0x4};
```

```
uint8_t heart2[8] = {0x0, 0x0, 0xa, 0x1f, 0xe, 0x4, 0x0};
```

```
void setup() {
```

```
  //---Serial
```

```
  Serial.begin(9600);
```

```
  //---DS1307
```

```
  if (!rtc.begin()) {
```

```
    // Serial.println("Couldn't find RTC");   rtc はうまく動いているか
```

```
    // while (1);
```

```
  }
```

```
  if (!rtc.isrunning()) {
```

//

```
    // Serial.println("RTC is NOT running!");
```

```
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));   年月日時刻合わせ関数
```

```
    // January 21, 2014 at 3am you would call:
```

```
    // rtc.adjust(DateTime(2014, 1, 21, 3, 0, 0));
```

```
  }
```

//

```
  // . . . .
```

```
}
```



## 大域変数の定義

```
int toggle = 0;
int time_cur = 0, time_last = 0;
```

```
void loop() {
  time_cur = millis();
  if (time_cur - time_last > 1000) {
    time_last = millis();
```

1秒(1000ms)ごとに実行

```
    // heart beat
    lcd.setCursor(0, 1);
    if (toggle == 1) {
      lcd.write(2);
      //      digitalWrite(LED_PIN, 1);
      toggle = 0;
    } else {
      lcd.write(3);
      //      digitalWrite(LED_PIN, 0);
      toggle = 1;
    }
  }
}
```

ハート印を1秒ごとに動かす

## 年月日・時刻の表示

```
//--- time
DateTime now = rtc.now();
lcd.setCursor(1, 1);
lcd.print(now.year()-2000);
lcd.print('/');
lcdprint_zero(now.month());
lcd.print('/');
lcdprint_zero(now.day());
lcd.print(' ');
lcdprint_zero(now.hour());
lcd.print(':');
lcdprint_zero(now.minute());
```

```
}
```

```
}
```

表示桁数を2桁に合わせる関数

```
void lcdprint_zero(int n) {
  if (n >= 0 && n < 10) {
    lcd.write('0');
  }
  lcd.print(n);
}
```

## 温度・湿度・大気圧のテスト・スケッチ

```
// Filename: Example704.ino(THPSensor00.ino)
```

```
#include "SparkFunBME280.h" Libraryを呼んでくる
#include <LiquidCrystal_I2C.h>
```

```
BME280 bme280; //BME280 オブジェクトを定義する
LiquidCrystal_I2C lcd(0x27, 16, 2); //LCD1602 (0x27)
```

```
// Custom character 文字パターンを定義する
uint8_t humid[8] = {0x4, 0x4, 0xa, 0xa, 0x11, 0x11, 0x11, 0xe};
uint8_t pressure[8] = {0x04, 0xe, 0xe, 0x11, 0x11, 0x1f, 0x1f};
```

```
void setup() {
  //---BME280
  bme280.settings.commInterface = I2C_MODE;
  bme280.settings.I2CAddress = 0x76;
  bme280.settings.runMode = 3; //Normal mode
  bme280.settings.tStandby = 0;
  bme280.settings.filter = 0; bme280の初期設定
  bme280.settings.tempOverSample = 1;
  bme280.settings.pressOverSample = 1;
  bme280.settings.humidOverSample = 1;
  bme280.begin();

  lcd.init();
  lcd.backlight();
  lcd.createChar(0, temp); 文字パターンの辞書を作る
  lcd.createChar(1, humid);
  lcd.createChar(4, pressure);
  lcd.clear();
  lcd.home();
}
```

```
int toggle = 0;
int time_cur = 0, time_last = 0;
```

```
void loop() {
  time_cur = millis();
  if (time_cur - time_last > 1000) {
```

1秒(1000ms)ごとに実行

```
    time_last = millis();
```

```
    //--- temperature, humidity, pressure
```

```
    lcd.setCursor(0, 0);
```

```
    lcd.write(0); // temprature
```

```
    lcd.print(bme280.readTempC(), 1);
```

bme280の表示

```
    lcd.print(" ");
```

```
    lcd.write(1); // humidity
```

```
    lcd.print(bme280.readFloatHumidity(), 0);
```

```
    lcd.print("%");
```

```
    lcd.print(" ");
```

```
    lcd.write(4); // pressure
```

```
    lcd.print(bme280.readFloatPressure() / 100.0, 0);
```

```
}
```

```
}
```

## フルカラーLEDのテスト・スケッチ

このスケッチには解説がありません。各自、適宜注釈をつけてください。

```
// Example705.ino(fastLEDTTest00.ino)

#include "FastLED.h"

#define DATA_PIN 12
#define NUM_LEDS 1

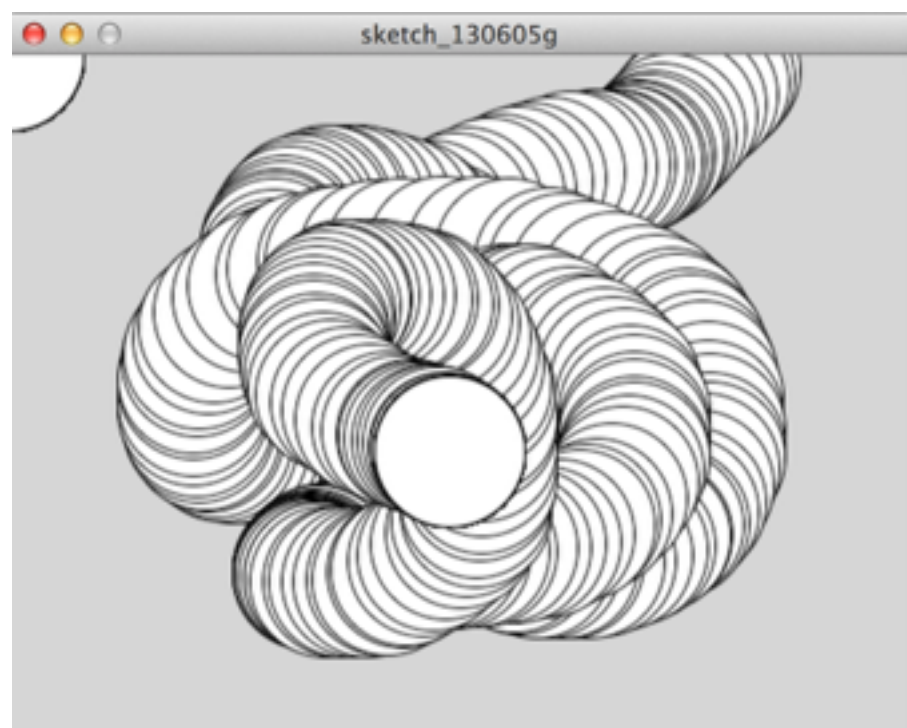
CRGB leds[NUM_LEDS]; //FastLED pin 12
int h=0;

void setup() {
  FastLED.addLeds<WS2811, DATA_PIN, RGB>(leds, NUM_LEDS);
  FastLED.setBrightness(16); // 0-255
  leds[0] = CRGB(0, 0, 0);
  FastLED.show();
}

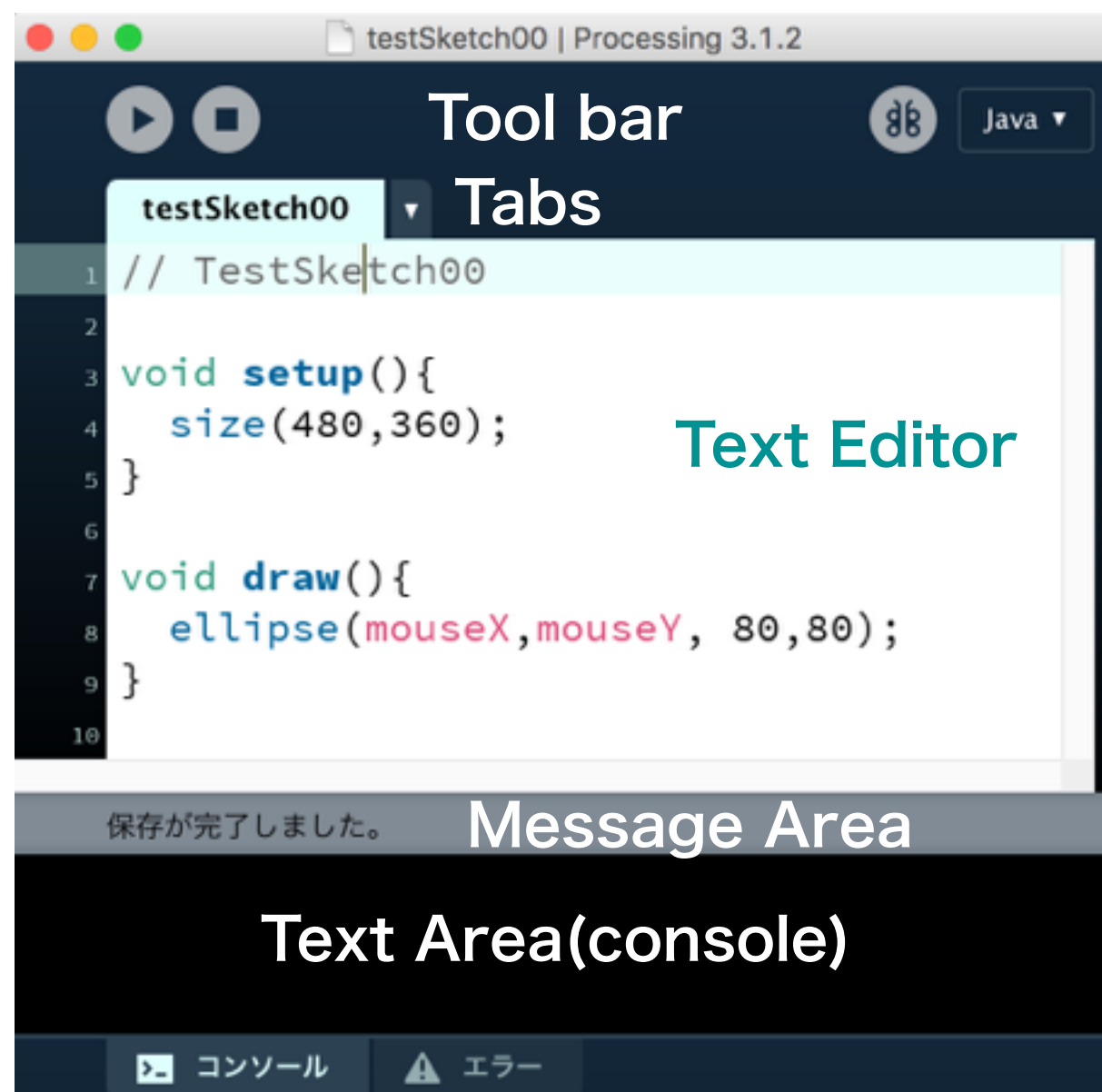
void loop() {
  leds[0] = CHSV(h, 255, 255);
  FastLED.show();
  h += 10;
  if (h > 255) h = 0;
  delay(500);
}
```

## 2. Processing をインストールして使ってみる

# Processing の開発環境(PDE)



Display Window



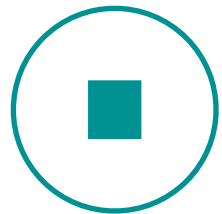


## プログラム（スケッチ）をつくる作業の流れ

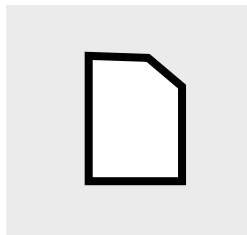
---



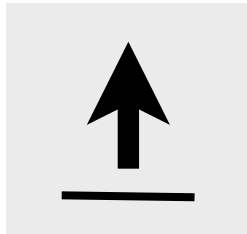
Run



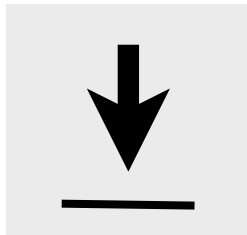
Stop



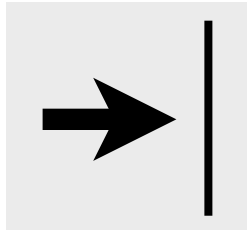
New



Open



Save



Export

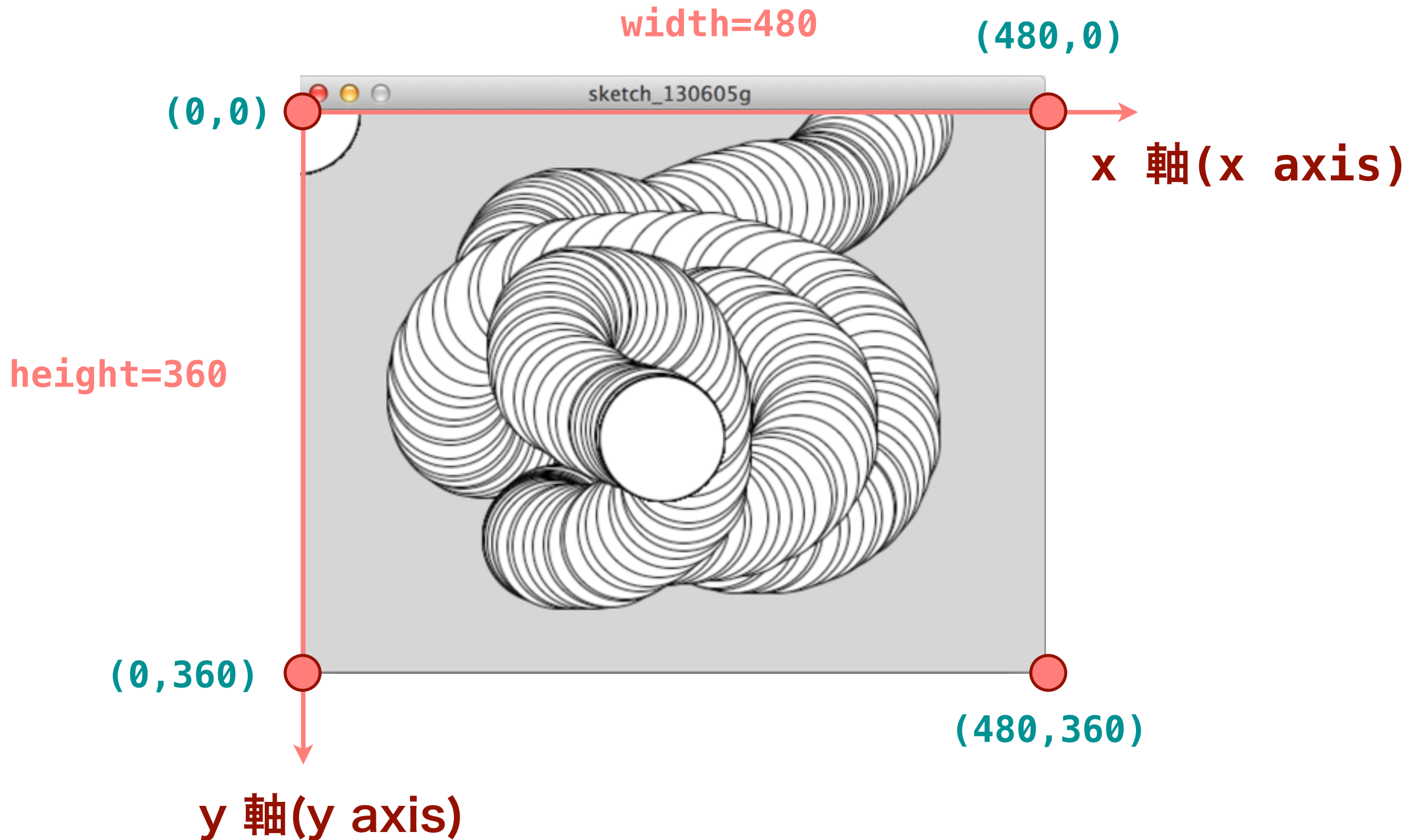
(1) スケッチを書く

(2) Run を押して実行

(3) Stop を押して止める

スケッチの管理

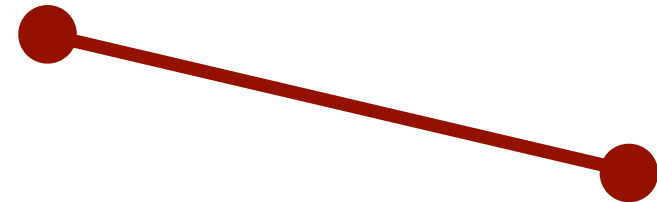
# 表示ウィンドウ (display window)



## 基本图形

```
line(x1, y1, x2, y2);
```

$(x_1, y_1)$



$(x_2, y_2)$

```
rect(x, y, width, height);
```

$(x, y)$



height

width

$(x_1, y_1)$

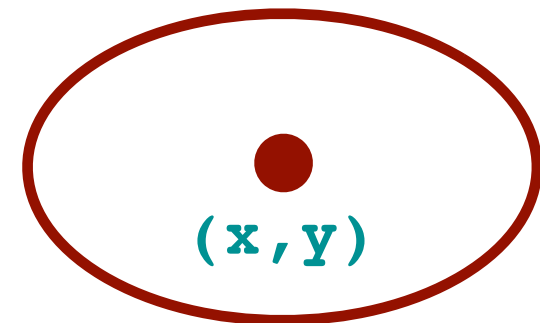


$(x_2, y_2)$

```
ellipse(x, y, width, height);
```

```
rectMode(CORNERS);
```

```
rect(x1, y1, x2, y2);
```



height

width

# Arduino のスケッチ : Example 707A

---

```
// Example 707A
// Serial communication with Processing

int ledPin=13;
int sensorPin = A0;
int val = 0;

void setup() {
  Serial.begin(9600); ←
}

void loop() {
  val = analogRead(sensorPin)/4;
  analogWrite(ledPin, val);
  Serial.write(val); ←
  delay(100);
}
```

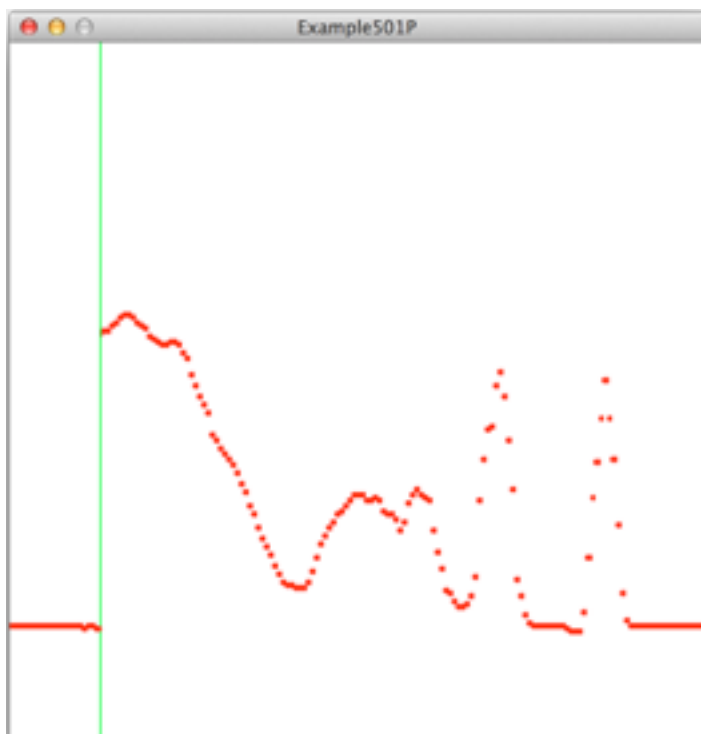
# Processing のスケッチ 707P

```
// Example 707P
// Serial communication with Arduino

import processing.serial.*;

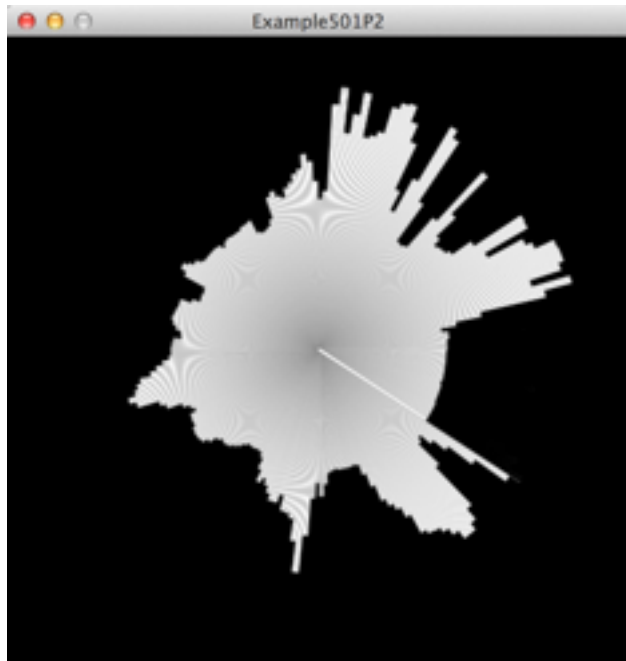
Serial port;
int x;
float val;

void setup() {
  size(500, 500);
  frameRate(30);
  String arduinoPort = Serial.list()[5];
  port = new Serial(this, arduinoPort, 9600);
  background(255);
}
```



```
void draw() {
  if ( port.available() > 0) {
    val = port.read();
    val = map(val, 0, 255, height, 0);
  }
  strokeWeight(1);
  stroke(255);
  line(x, 0, x, height); // Black line
  stroke(0,255,0);
  line(x+1, 0, x+1, height); // White line
  strokeWeight(4);
  stroke(255,0,0);
  point(x, val-50);
  x++;
  if (x > width) {
    x = 0;
  }
}
```

# Processing のスケッチ 708P



```
import processing.serial.*;
```

```
Serial port;  
float val;  
float angle;  
float radius;
```

```
void setup() {  
  size(440, 440);  
  frameRate(30);  
  strokeWeight(2);  
  String arduinoPort = Serial.list()[5];  
  port = new Serial(this, arduinoPort, 9600);  
  background(0);  
}
```

```
void draw() {  
  if ( port.available() > 0) {  
    val = port.read();  
    radius = map(val, 0, 255, 0, height * 0.45);  
  }  
}
```

```
int middleX = width/2;  
int middleY = height/2;  
float x = middleX + cos(angle) * height/2;  
float y = middleY + sin(angle) * height/2;  
stroke(0);  
line(middleX, middleY, x, y);
```

```
x = middleX + cos(angle) * radius;  
y = middleY + sin(angle) * radius;  
stroke(255);  
line(middleX, middleY, x, y);  
angle += 0.01;  
}
```