コンピューター・アート秋学期ABC

「ProcessingとArduinoによるフィジカルコンピューティング1」

秋学期のテーマ

「Processingとフィジカルコンピューティングを利用したメディアアート作品の制作」

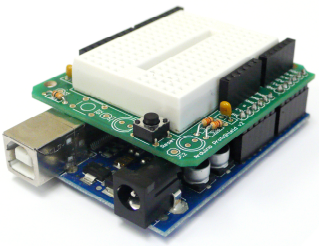
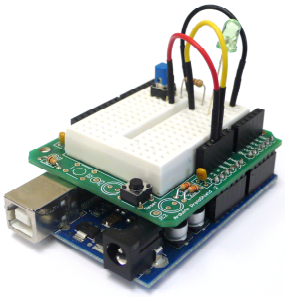
ArduinoボードとProcessingを接続し、センサ等の変化に応じてコンピューターグラフィックスが変化するインタラクティブな作品の制作を行う。

フィジカルコンピューティングとは

「フィジカル・コンピューティング」とはニューヨーク大学から始まった教育プログラムである。既存のコンピュータのインターフェイス（キーボード、マウス、ディスプレイなど）を超えて、「人々がいかにコンピュータとコミュニケーションし得るか」について考え直すことを提案し、コンピュータや電子回路に関する原理原則を習得することで、ブラックボックスになりがちなコンピュータの仕組みについて理解を深めることを目的にしている。

Arduinoボードとは

フィジカル・コンピューティングを実現する為のデバイス。センサやアクチュエータとコンピュータとの間に接続することで、センサの情報をコンピュータに入力したり、コンピューターからの出力でアクチュエータを動作させることが可能である。今回はArduinoボードの中にProcessingと通信するプログラムをあらかじめ組み込んである。この授業では扱わないが、Arduinoボードにプログラムを転送することで、コンピュータと接続せず単体で動作させることも出来る。

Arduinoボード	配線する為にブレッドボードを接続したArduinoボード	ブレッドボード上での電子部品の配線例
		

参考URLと参考図書

●参考URL

Arduinoの本家サイト　http://www.arduino.cc/

Arduinoと様々なソフトを接続する方法　http://www.arduino.cc/playground/Main/InterfacingWithSoftware

ArduinoとProcessingの接続方法とプログラミング　http://www.arduino.cc/playground/Interfacing/Processing

Arduinoを使用した作品集　http://www.arduino.cc/playground/Projects/ArduinoUsers

建築発明工作ゼミ(このサイトは主にArduino単体で動作させることを目的としている)

- http://kousaku-kousaku.blogspot.com/2008/07/arduino.html

●参考書籍

「Getting Started With Arduino」Make Books

「Making Things Talk -Arduinoで作る「会話」するモノたち」オライリージャパン

「Physical Computing: Sensing and Controlling the Physical World with Computers」Course Technology Ptr

（書籍はAmazonで購入可能）

Arduinoボードや電子部品の販売店

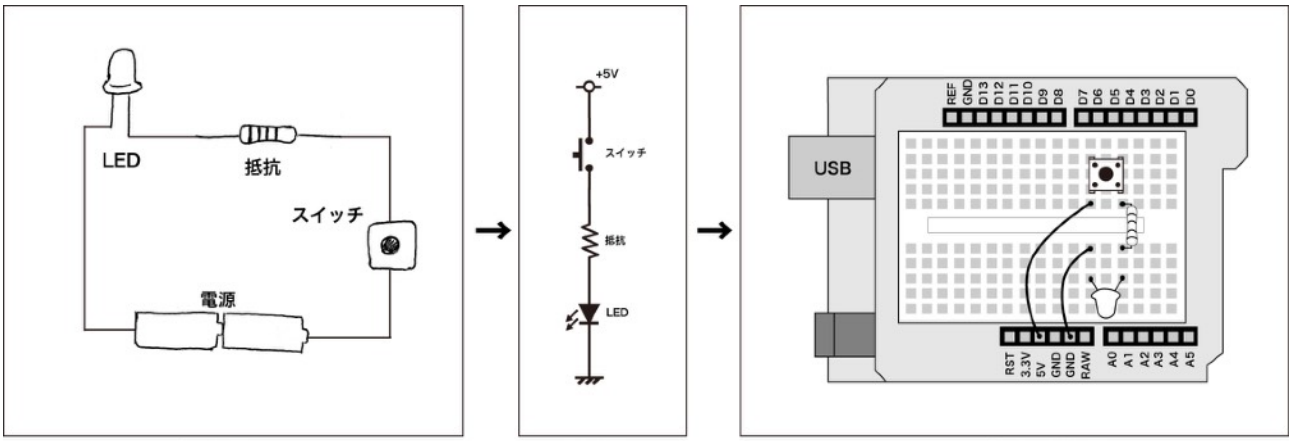
スイッチサイエンス	http://www.switch-science.com/
ストロベリーリナックス	http://strawberry-linux.com/
メカロボショップ	http://www.mecharoboshop.com/
秋月電子通商	http://akizukidenshi.com/
マルツパーツ	https://www.marutsu.co.jp/
Sparkfun	http://www.sparkfun.com/

（上記はインターネットでも購入可能）

授業のスケジュール

1　フィジカルコンピューティングについて、arduinoについて、参考作品紹介
2　LEDの点灯
3　アナログ入力、デジタル入力・出力について
4　ブレッドボードについて。スイッチの使用
5　様々なボタンやボリューム（可変抵抗器）の使用
6　明るさをはかる(ＣＤＳセンサー)
7　距離をはかる（PSDセンサー）
8　傾き・加速度をはかる（加速度センサ）
9　作品計画
10　中間プレゼンテーション
11　制作
12　制作
13　制作
14　制作
15　発表会

電子回路配線の基礎



概略図　回路図　実体配線図

ボタンを押すとLEDが点灯する回路

各自のパソコンでの事前準備方法

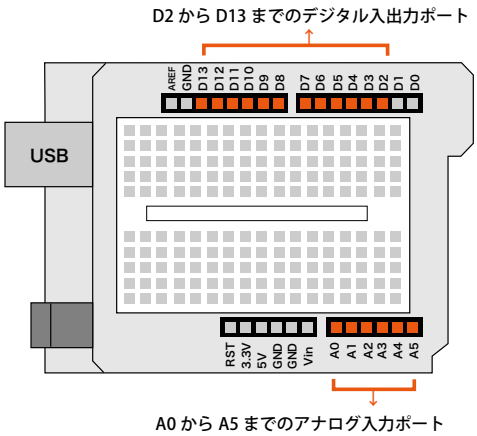
（大学で使用するArduinoとパソコンには既にソフトとドライバがインストール済なので下記の作業は必要ありません）

●準備するもの
Arduino本体、ブレッドボード（SparkFunプロトシールド・キットなど）、USBケーブル、電子部品（センサー等）
●ソフトウェアのインストール

1. Arduinoのページより「Arduino Software」をダウンロードする。(http://arduino.cc/en/Main/Software)
2. 「Arduino and Processing (http://www.arduino.cc/playground/Interfacing/Processing)」のページの手順を見ながら、「Library for Processing v2.0」をインストールする。

**はじめに

Arduinoのポートについて

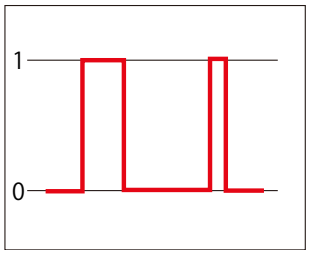


Arduinoには、デジタル信号を扱うD2からD13までの12カ所のデジタル入出力ポートと、アナログ信号を扱うA0からA5までの6カ所のアナログ入力ポートがある。
それぞれのポートごとに電子部品を接続することで、ホストコンピュータ内のソフトウェア（Processing)との通信を行う。

注意:デジタルポートのD1とD0は使用できない

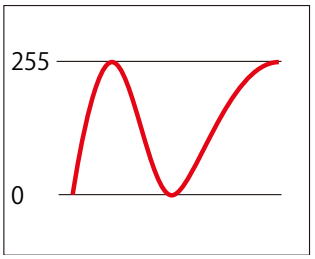
デジタル信号とアナログ信号について
→arduinoはデジタル信号とアナログ信号の両方を扱う事が可能

デジタル信号



スイッチの様に信号のオン・オフが明確な場合はデジタルポートを使用する。

アナログ信号



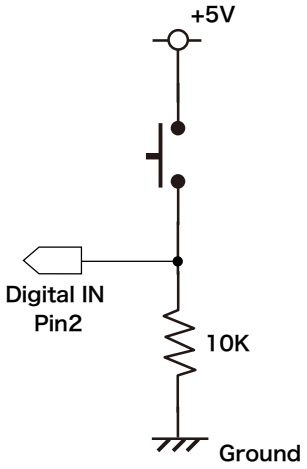
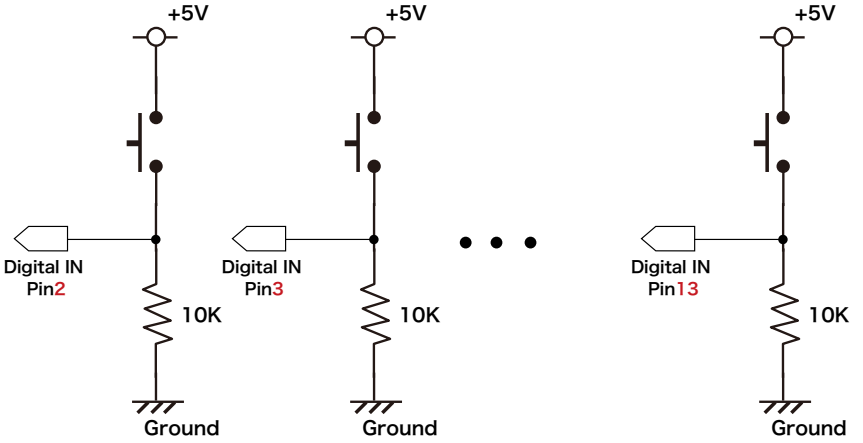
距離を測るセンサやボリュームの様に信号の値が連続的に変化する場合はアナログポートを使用する。

**スイッチを使ってみよう（デジタル入力）

部品の解説

名称	プッシュスイッチ	トグルスイッチ	抵抗
部品の写真			
回路記号			
用途・機能	ボタンを押すことで、電気的な接続と切断を行う。 押している間のみONになるスイッチのことを「モーメンタリスイッチ」、押すごとにON・OFFが切り替わるスイッチのことを「オルタネイトスイッチ」という。 極性なし。	トグルを倒すことで、電気的な接続と切断を行う。 極性なし。	**カラーコードは、茶黒橙金** 電流の流れを制限する。 極性なし。

スイッチの接続するための回路図

スイッチが1つの場合	スイッチが複数の場合
	 注意: 複数のスイッチを接続する場合は、スイッチごとに抵抗を用意する必要がある。一つの抵抗を複数のスイッチで共有することは出来ない。

電子部品の配置にあたっての注意

基本的に電子部品を配線する間は電源を抜く（USBケーブルを抜く）
USBケーブルを繋ぐ前に、回路が正しいか良く確かめること
濡れた手で触らない

Processingの基本プログラム（下線部分がArduinoとProcessingの通信に必要な部分）

「一つのスイッチを接続する」 (digital_input1.pde)

```
import processing.serial.*;                                  →Arduinoを使用するための初期設定
import cc.arduino.*;
Arduino arduino;

void setup() {
  size(300, 300);

  println(Arduino.list());                                   →Arduinoが接続されている USBポートをコンソール内に表示
  arduino = new Arduino(this, Arduino.list()[0], 57600);     →赤字の部分にUSBのポート番号を入力する
  arduino.pinMode(2, Arduino.INPUT);                         →デジタルピン2を「インプット」に指定します。
}                                                            （複数のポート使用する場合は、この行を複数記述します）

void draw() {
  background(255);
  stroke(0);

  if (arduino.digitalRead(2) == Arduino.HIGH){               →デジタルピン2がオン(HIGH)の場合の処理
    fill(142,240,138);
  }

  if (arduino.digitalRead(2) == Arduino.LOW){                →デジタルピン2がオフ(LOW)の場合の処理
    fill(36,72,147);
  }

  ellipse(150,150,200,200);

}
```

デジタル入力を用いた参考プログラム

「スイッチを用いた、2つのデジタル入力」
(digital_input2.pde)

```
import processing.serial.*;
import cc.arduino.*;
Arduino arduino;

void setup() {
  size(300, 300);
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  arduino.pinMode(2, Arduino.INPUT);
  arduino.pinMode(3, Arduino.INPUT);
}

void draw() {
  background(255);
  stroke(0);

  if (arduino.digitalRead(2) == Arduino.HIGH){
    fill(142,240,138);
  }
    if (arduino.digitalRead(2) == Arduino.LOW){
    fill(36,72,147);
  }
    ellipse(100,150,100,100);

    if (arduino.digitalRead(3) == Arduino.HIGH){
    fill(69,211,62);
  }
    if (arduino.digitalRead(3) == Arduino.LOW){
    fill(255,255,224);
  }
    ellipse(200,150,100,100);

}
```

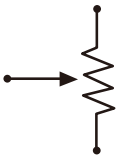
スイッチを使用した作品を制作するにあたってのポイント

既製の様々な種類の電気スイッチを使用したり、スイッチを自作することで新しいインターフェースの提案を行うことが可能である。

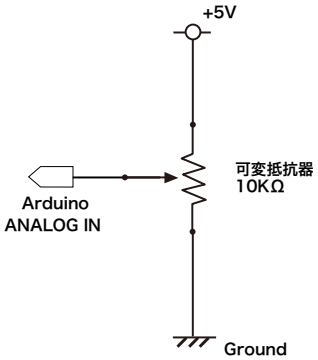
2つのデジタル入力を使用したボールの打ち返しゲーム(digital_input3.pde)

```
import processing.serial.*;
import cc.arduino.*;
Arduino arduino;

float x = 50.0;   //ボールの最初のX座標
float y = 50.0;   //ボールの最初のY座標
float   radius = 15.0;   //ボールの半径
float speedX = 15.0;   //ボールのX軸方向のスピード
float speedY = 12.4;   //ボールのY軸方向のスピード
int directionX = 1;     //ボールの進むX座標方向
int directionY = -1;    //ボールの進むY座標方向

float racket = 250;
int gameover = 0;

void setup(){
  size(500,500);
  smooth();
  noStroke();
  ellipseMode(RADIUS);
  frameRate(24);

  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  arduino.pinMode(2, Arduino.INPUT);
  arduino.pinMode(3, Arduino.INPUT);
}

void draw(){
  fill(255,120);
  rect(0,0,width,height);
  fill(100);
  ellipse(x, y, radius, radius);

  x += speedX * directionX;
  if((x > width-radius) || (x < radius)){     //ボール画面の端で跳ね返る処理
    directionX = -directionX;
  }

  y += speedY * directionY;
  if ((y > height-radius) || (y < radius)){     //ボール画面の端で跳ね返る処理
    directionY = -directionY;
  }

  rect (racket, 450,100,10);//drawing racket
  if (arduino.digitalRead(2) == Arduino.HIGH){
    racket = racket + 10;
    if(racket>width-100)racket = width-100;
  }

   if (arduino.digitalRead(3) == Arduino.HIGH){
    racket = racket - 10;
    if(racket<0)racket = 0;
  }

  if ((racket < x) && (racket+100 > x) && (y < 450)&&(y > 440) ){    //ボールがラケットに当たった場合
    directionY = -directionY;
  }

}
```

**可変抵抗器を接続してみよう（アナログ入力）

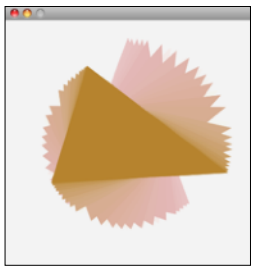
名称	可変抵抗器（Potentiometers）
部品の写真	
回路記号	
用途・機能	抵抗値を変化させることで、連続的に電圧を変化させる

可変抵抗器を接続するための回路図



可変抵抗器を使用した基本プログラム（下線部分がArduinoに関係する部分）

(analog_input1.pde)



```
import processing.serial.*;
import cc.arduino.*;
Arduino arduino;

void setup() {
  size(400, 400);

  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  colorMode(HSB);
  smooth();
}

void draw() {
  noStroke();
  fill(255,10);
  rect(0,0,width,height);
  println(arduino.analogRead(0));
  translate(width/2,height/2);
  rotate(radians(arduino.analogRead(0)));
  fill(arduino.analogRead(0) / 4 , 180,180);
  triangle(0,-170,-100,100,100,100);
}
```

→Arduinoを使用するための初期設定

→Arduinoが接続されている USBポートをコンソール内に表示
→赤字の部分にUSBのポート番号を入力する

→現在のアナログピン0の値をコンソールに表示（なくても良い）

→「アナログピン0」の値によって座標を回転させる
→「アナログピン0」の値によって塗りつぶしの色（色相）を変化させる

ポイント

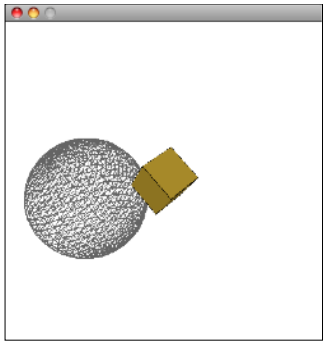
1. アナログ入出力の場合はデジタル入出力と異なり、setup()内で入出力の設定を行う必要はない
2. arduino.analogRead(アナログポート番号)には、現在の可変抵抗器の値（0～1023）が代入される

「可変抵抗器を使用した基本プログラム（その2）」

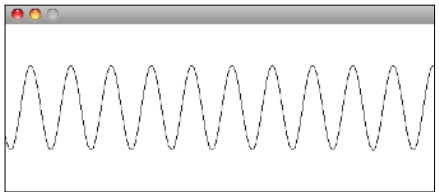
(analog_input2.pde)

（このプログラム内では、3次元のグラフィックを扱っている。紹介については、下記のURLを参照のこと）
http://processing.org/learning/3d/

```
import processing.serial.*;
import cc.arduino.*;
Arduino arduino;

void setup() {
  size(400, 400,P3D);

  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  colorMode(HSB);

}

void draw() {
  stroke(0);
  background(255);
  lights();
  translate(width/2,height/2);
  rotateY(radians(arduino.analogRead(0)));
  rotateX(radians(30));
  rotateZ(radians(30));
  fill(arduino.analogRead(0) / 4 , 180,180);
  box(50,50,50);

  noFill();
  translate(250,-100,0);
  stroke(100);
  sphere(120);
}
```



analog_input2.pde



analog_input_to_sound.pde

「可変抵抗器を使用したプログラム（サウンドオシレーター）」

(analog_input_to_sound.pde)

（このプログラム内では、サウンドライブラリminim扱っている。紹介については、下記のURLを参照のこと）
http://code.compartmental.net/tools/minim/

```
import processing.serial.*;
import cc.arduino.*;
import ddf.minim.*;
import ddf.minim.signals.*;
Minim minim;
Arduino arduino;
AudioOutput out;
SineWave sine;

void setup()
{
  println(Arduino.list());
  size(512, 200);
  // always start Minim before you do anything with it
  Minim.start(this);
  out = Minim.getLineOut(Minim.STEREO);
  // create a sine wave Oscillator
  //set to 440 Hz, at 0.5 amplitude,
  //sample rate 44100 to match the line out
  sine = new SineWave(440, 0.5, 44100);
  // set the portamento speed
  //on the oscillator to 200 milliseconds
  sine.portamento(200);
  // add the oscillator to the line out
  out.addSignal(sine);
  arduino = new Arduino(this, Arduino.list()[0], 57600);
}

void draw()
{
  background(0);
  stroke(255);
  // draw the waveforms
  for(int i = 0; i < out.left.size()-1; i++)
  {
    line(i, 100 + out.left.get(i)*100, i+1, 100 +
out.left.get(i+1)*100);
  }

  float freq = map(arduino.analogRead(0), 0, 1023, 1500,
60);
  sine.setFreq(freq);
}

void stop()
{
  out.close();
  super.stop();
}
```

**様々なセンサーを接続してみよう（アナログ入力）

基本のプログラム（下線部分がArduinoに関係する部分）(analog_input_simple.pde)

```
import processing.serial.*;
import cc.arduino.*;
Arduino arduino;
int s;

void setup() {
  size(800, 800);
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  colorMode(HSB);
  smooth();
}
void draw() {
  background(255);
  stroke(0);
  fill(100);
  s = arduino.analogRead(0);
  s = s / 2;
  println(s);
  ellipse(400,400,s,s);
}
```

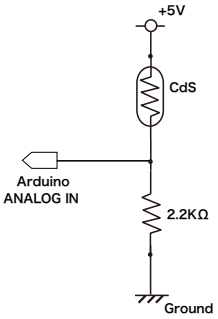
Ⅰ.明るさをはかる

アナログ入力にCdS光センサを接続し、明るさの変化を計測する

●必要な部品　→　CdS光センサ、抵抗2.2KΩ（赤赤赤金）

名称	CdS光センサ
部品の写真	
回路記号	
用途・機能	表面に当たる光の量に従って抵抗値が変化し、暗いときには抵抗値が大きく、明るいときには抵抗値が小さくなる。極性なし。

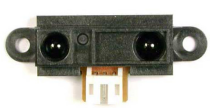
CdS光センサを接続するための回路図



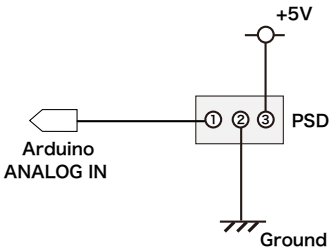
Ⅱ.距離をはかる

アナログ入力にPSD測距センサを接続し、センサと対象物の距離を計測する

●必要な部品　→　PSD測距センサ

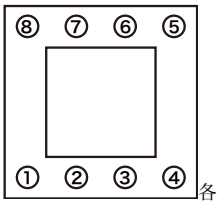
名称	PSD測距センサ
部品の写真	
回路記号	① ② ③
	端子1　アナログ信号出力 端子2　Ground(マイナス) 端子3　5V（プラス）
用途・機能	PSD測距センサは、赤外線を投射し、三角測量の原理を用いて物体との距離を測る。各端子の接続先は決まっており間違えないこと。

PSD測距センサの回路図



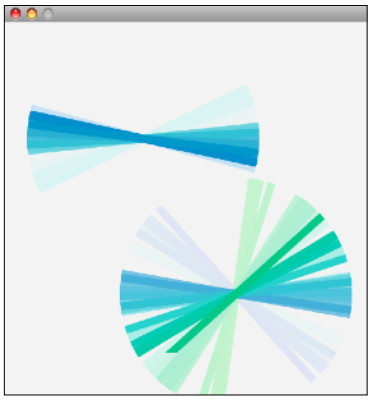
**加速度センサーを接続してみよう（アナログ入力）

●必要な部品　→　加速度センサ KXM52

名称	加速度センサー（acceleration sensor）
部品の写真	
回路記号と 接続先	 ⑧ ⑦ ⑥ ⑤ ① ② ③ ④ 各ピンの接続先　ピンの番号は緑色の基板上に表示されています **1番ピン →　+5V** **2番ピン →　+5V** **3番ピン →　Ground** **4番ピン →　接続しない** **5番ピン →　Ground** **6番ピン →　アナログ信号出力（X軸出力）**　アナログ入力ピン0へ **7番ピン→　アナログ信号出力（Y軸出力）**　アナログ入力ピン1へ **8番ピン→　接続しない**
用途・機能	物体の加速度及び、重力加速度を計測する
注意点	誤配線は故障の原因になるので、正しく配線すること。衝撃に弱いので落下させないこと

加速度センサを使用したサンプルプログラム1　(tilt_sensor_1.pde)

```
import processing.serial.*;
import cc.arduino.*;
Arduino arduino;
int a;
int b;
void setup() {
  size(470, 480);
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  smooth();
  colorMode(HSB);
}

void draw() {
  fill(255,10);
  rect(0,0,width,height);
  noStroke();

  a = arduino.analogRead(0) ;
  pushMatrix();
  fill(a/4,255,200);
  translate(180,150);
  rotate(radians(a/2));
  rect(-5,-150,10,300);
  popMatrix();

  b = arduino.analogRead(1) ;
  pushMatrix();
  fill(b/4,255,200);
  translate(300,350);
  rotate(radians(b/2));
  rect(-5,-150,10,300);
  popMatrix();
}
```



→現在のアナログポート0の値を読み込む（X軸の加速度）
→現在の座標を保存（保存した座標は下記のpopMatrixで使用する）
→「アナログポート0」の値によって塗りつぶしの色（色相）を変化させる
→原点座標を(180,150)に変更
→「アナログポート0」の値によって座標を回転させる
→座標(-5,-150)に幅10ピクセル、高さ300ピクセルの矩形を描く
→pushMatrixで保存した座標に戻す

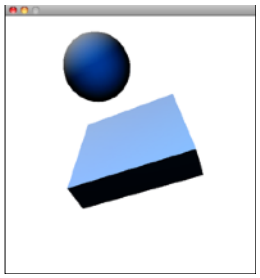
→現在のアナログポート1の値を読み込む（Y軸の加速度）
→現在の座標を保存（保存した座標は下記のpopMatrixで使用する）
→「アナログポート1」の値によって塗りつぶしの色（色相）を変化させる
→原点座標を(300,350)に変更
→「アナログポート1」の値によって座標を回転させる
→座標(-5,-150)に幅10ピクセル、高さ300ピクセルの矩形を描く
→pushMatrixで保存した座標に戻す

サンプルプログラム2（三次元その1)

(tilt_sensor_2.pde)

（このプログラム内では、3次元のグラフィックを扱っている。紹介については、下記のURLを参照のこと）

http://processing.org/learning/3d/



```
import processing.serial.*; //Arduinoを扱う際に必要
import processing.opengl.*; //3Dグラフィックスを扱う際に必要
import cc.arduino.*;  //Arduinoを扱う際に必要
Arduino arduino;  //Arduinoを扱う際に必要

float a; //アナログポート1を代入する変数を用意する
float b; //アナログポート0を代入する変数を用意する

void setup() {  //setupは一度しか実行されない
  //3Dグラフィックスを扱うためにはOPENGLを記入すること
  size(470, 480,OPENGL);
  //Arduinoが接続されているポートを指定する(数字を変更すること)
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  smooth();   //グラフィックを滑らかにする
  frameRate(15); //一秒間にアニメーションを表示する回数
}

void draw() { //draw内はプログラムを終了しない限り止まらない
  background(255);  //画面を白色で塗りつぶす
  //三次元空間上(200,150,200)に(0,100,255)の色のライトを設置する
  pointLight(0,100,255,200,150,200);
  //三次元空間上(0,-150,200)に(255,255,255)の色のライトを設置する
  pointLight(255,255,255,0,-150,200);

  //変数aにアナログポート1に接続されたセンサーの現在値を代入する
  a = arduino.analogRead(1) ;
  //変数bにアナログポート0に接続されたセンサーの現在値を代入する
  b = arduino.analogRead(0) ;

  fill(150);  //これから描く図形の塗りつぶし色をグレー(150)に設定
  translate(240,240); //座標(240,240)を原点(0,0)とする
  //X軸方向に(-1*(a/3+10))度、座標を回転させる
  rotateX(-1*radians(a/3+10));
  //Z軸方向に(b/3)度、座標を回転させる
  rotateZ(1*radians(b/3));

  //これから描く図形の線は描かない
  noStroke();
  //幅200ピクセル高さ20ピクセル奥行き200ピクセルの直方体を描く
  box(200,50,200);
  //現在の座標からY軸方向にマイナス150ピクセル原点を移動する
  translate(0,-150,0);
  //半径50ピクセルの球を描く
  sphere(50);

}
```

サンプルプログラム3（三次元その2)

(tilt_sensor_3.pde)

左側のプログラムと同等であるが、スムーズに動作するようプログラムが改良させている。

```
import processing.serial.*; //Arduinoを扱う際に必要
import processing.opengl.*; //3Dグラフィックスを扱う際に必要
import cc.arduino.*; //Arduinoを扱う際に必要
Arduino arduino; //Arduinoを扱う際に必要

float a; //現在のアナログポート1の値を代入する変数を定義
float a_old; //1つ前のアナログポート1の値を代入する変数を定義
float a_old2; //2つ前のアナログポート1の値を代入する変数を定義
float a_old3; //3つ前のアナログポート1の値を代入する変数を定義

float b;  //現在のアナログポート0の値を代入する変数を定義
float b_old; //1つ前のアナログポート0の値を代入する変数を定義
float b_old2; //2つ前のアナログポート0の値を代入する変数を定義
float b_old3; //3つ前のアナログポート0の値を代入する変数を定義

void setup() { //setupは一度しか実行されない
  //3Dグラフィックスを扱うためにはOPENGLを記入すること
  size(470, 480,OPENGL);
  //Arduinoが接続されているポートを指定する(数字を変更すること)
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  smooth(); //グラフィックを滑らかにする
  frameRate(24); //一秒間にアニメーションを表示する回数
}

void draw() {//draw内はプログラムを終了しない限り止まらない
  background(255); //画面を白色で塗りつぶす
  //三次元空間上(200,150,200)に(0,100,255)の色のライトを設置する
  pointLight(0,100,255,200,150,200);
  //三次元空間上(0,-150,200)に(255,255,255)の色のライトを設置する
  pointLight(255,255,255,0,-150,200);

  //変数aにアナログポート1に接続されたセンサーの現在値を代入する
  a = arduino.analogRead(1) ;
  //今回と前回と前々回と前々前回を、全て足して4で割りaに代入する
  //平均化
  a = (a+a_old+a_old2+a_old3)/4;

  //変数bにアナログポート0に接続されたセンサーの現在値を代入する
  b = arduino.analogRead(0) ;
  //今回と前回と前々回と前々前回を、全て足して4で割りbに代入する
  //平均化
  b = (b+b_old+b_old2+b_old3)/4;

  fill(150);  //これから描く図形の塗りつぶし色をグレー(150)に設定
  translate(240,240); //座標(240,240)を原点(0,0)とする
  //X軸方向に(-1*(a/3+10))度、座標を回転させる
  rotateX(-1*radians(a/3+10));
  //Z軸方向に(b/3)度、座標を回転させる
  rotateZ(1*radians(b/3));

  noStroke();  //これから描く図形の線は描かない
 //幅200ピクセル高さ20ピクセル奥行き200ピクセルの直方体を描く
  box(200,50,200);
  //現在の座標からY軸方向にマイナス150ピクセル原点を移動する
  translate(0,-150,0);
  sphere(50);  //半径50ピクセルの球を描く

  a_old3 = a_old2; //前々前回の値に前々回の値を代入
  a_old2 = a_old; //前々回の値に前回の値を代入
  a_old = a; //前回の値に今回の値を代入
  b_old3 = b_old2; //前々前回の値に前々回の値を代入
  b_old2 = b_old; //前々回の値に前回の値を代入
  b_old = b; //前回の値に今回の値を代入
}
```

サンプルプログラム4

(tilt_sensor_4.pde)

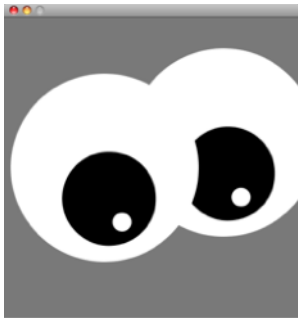
アナログポート1のみ使用し、センサーの値に応じて画面内の線の進行方向を変化させる

```
import processing.serial.*; //Arduinoを扱う際に必要
import cc.arduino.*; //Arduinoを扱う際に必要
Arduino arduino; //Arduinoを扱う際に必要

int a; //現在のアナログポート1の値を代入する変数を定義
float x=200; //現在の線のX座標を変数「x」として200を代入
float y=200; //現在の線のY座標を変数「y」として200を代入
float speed= 3; //線が動く速度

void setup() { //setupは一度しか実行されない
  //キャンバスのサイズを幅470ピクセル、高さ480ピクセルにする
  size(470, 480);
  //Arduinoが接続されているポートを指定する(数字を変更すること)
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  smooth(); //グラフィックを滑らかにする
  colorMode(HSB); //色をHSB(色相、彩度、輝度）で指定する
  frameRate(15); //一秒間にアニメーションを表示する回数
  background(255); //画面を白色で塗りつぶす
}

void draw() {//draw内はプログラムを終了しない限り止まらない
  //変数aにアナログポート1に接続されたセンサーの現在値を代入する
  a = arduino.analogRead(1) ;
  //変数aの値をコンソールに表示。プログラムの実行には影響しない
  println(a);
  //これより図形が描く線は、ブラックで透明度(200)で描写する
  stroke(0,200);
  //変数xに値を追加し、X座標方向に描く図形を移動させる
  //傾き「cos(radians(a/2))」に応じて、X座標の移動量が変化する
  //また、変数「speed」によっても、移動量を変化させている
  x = x + cos(radians(a/2)) * speed;
  //変数xに値を追加し、Y座標方向に描く図形を移動させる
  //傾き「sin(radians(a/2))」に応じて、Y座標の移動量が変化する
  //また、変数「speed」によっても、移動量を変化させている
  y = y +sin(radians(a/2)) * speed;
  //変数xに値を追加し、X座標方向に描く図形を移動させる
  //原点(0,0)をX軸方向へxピクセル、Y軸方向へyピクセル移動する
  translate(x,y);
  //座標を「a/2」度、回転させる
  rotate(radians(a/2));
  line(0,-20,0,20); //座標(0,-20)から(0,20)までの直線を描く

  if(x < 0) x = width; //もし、変数xが0以下になった場合、画面外にはみ出したと判断し、xにwidth(この場合は470)を代入する
  if(y < 0) y = height; //もし、変数yが0以下になった場合、 画面外にはみ出したと判断し、yにheight(この場合は480)を代入する
  if(x > width) x = 0; //もし、変数xがwidth以上(470以上)になった場合、 画面外にはみ出したと判断し、xに0を代入する
  if(y > height) y =0 ; //もし、変数 y がheight以上(480以上)になった場合、画面外にはみ出したと判断し、yに0を代入する

}
```

サンプルプログラム5

(tilt_sensor_5.pde)

アナログポート1のみ使用し、センサーの値に応じて画面内の図形を回転させる



```
import processing.serial.*;  //Arduinoを扱う際に必要
import cc.arduino.*;  //Arduinoを扱う際に必要
Arduino arduino;  //Arduinoを扱う際に必要

float a;  //現在のアナログポート1の値を代入する変数を定義
float a_old; //1つ前のアナログポート1の値を代入する変数を定義


void setup() { //setupは一度しか実行されない
  //キャンバスのサイズを幅470ピクセル、高さ480ピクセルにする
  size(470, 480);
  //Arduinoが接続されているポートを指定する(数字を変更すること)
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  smooth(); //グラフィックを滑らかにする
  frameRate(15); //一秒間にアニメーションを表示する回数
}

void draw() {//draw内はプログラムを終了しない限り止まらない
  background(120); //画面をグレーで塗りつぶす

 //変数aにアナログポート1に接続されたセンサーの現在値を代入する
  a = arduino.analogRead(1) ;
  //今回と前回の値を、足して2で割りaに代入する
  //平均化
  a = (a+a_old) / 2;

  eye(350,200);  //構造化を使用。座標(350,200)に図形を描く
  eye(160,240); //構造化を使用。座標(160,240)に図形を描く

  a_old = a; //前回の値に今回の値を代入


}

void eye(int x, int y){ //構造化文eye
  noStroke(); //今後の図形は線を描かない
  pushMatrix(); //現在の座標を保存
  //座標をX軸方向へxピクセル、Y軸方向へyピクセル移動する
  translate(x,y);
  fill(255);  //塗りつぶし色を白に指定
  ellipse(0,0,300,300);  //半径300ピクセルの円を描く
  fill(0);  //塗りつぶし色を黒に指定
  rotate(radians(a));  //座標をa度回転させる
  //原点よりX軸方向に50ピクセルの所に、半径150ピクセルの円を描く
  ellipse(50,0,150,150);
  fill(255);  //塗りつぶし色を白に指定
  //原点よりX軸方向に90ピクセル、Y軸方向に-15ピクセルで
  //半径30ピクセルの円を描く
  ellipse(90,-15,30,30);
  popMatrix(); //pushMatrixで保存した座標の戻す
}
```

サンプルプログラム6

(tilt_sensor_6.pde)

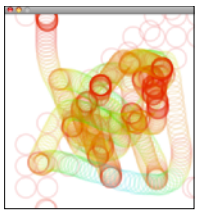
構造文tail内で、8ピクセルごとに描く線を上側に移動させ、回転させることで、曲線を描いている。また、センサーの値に応じて、回転量を変化させている。

```
import processing.serial.*; //Arduinoを扱う際に必要
import cc.arduino.*; //Arduinoを扱う際に必要
Arduino arduino; //Arduinoを扱う際に必要

float angle; //現在のアナログポート1の値を代入する変数を定義
float angle_old; //1つ前のアナログポート1の値を代入する変数を定義

void setup() { //setupは一度しか実行されない
  //キャンバスのサイズを幅400ピクセル、高さ400ピクセルにする
  size(400, 400);
  //Arduinoが接続されているポートを指定する(数字を変更すること)
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  //線の色を白、透明度204に指定する
  stroke(255,204);
  frameRate(15); //一秒間にアニメーションを表示する回数
  smooth(); //グラフィックを滑らかにする
}

void draw() {//draw内はプログラムを終了しない限り止まらない
  background(100); //画面をグレーで塗りつぶす
  //変数aにアナログポート1に接続されたセンサーの現在値を代入する
  //値の調節の為に、0.02を掛けて10を引いている
  angle =arduino.analogRead(1)* 0.02 - 10;
  //変数angleの値をコンソールに表示
  //プログラムの実行には影響しない
  println(angle);

  //今回と前回の値を、足して2で割りangleに代入する
  //平均化
  angle = (angle + angle_old) / 2;

  //構造化文
  //1番目の項目は図形を描写するX座標、2番目は図形の半径、3番目は図形の傾き具合
  tail(150,40,angle/1.3);
  tail(180,30,angle);
  tail(300,20,angle/1.3);
  tail(250,15,angle);
  tail(88,35,angle/2);
  tail(200,10,angle);
  angle_old = angle; //前回の値に今回の値を代入
}

void tail(int x, int units, float angle){//構造化文tail
  pushMatrix(); //現在の座標を保存
  //座標をX軸方向へxピクセル、Y軸方向へ300ピクセル移動する
  translate(x,300);
  for(int i = units; i> 0; i--){ //変数unitsの回数だけ繰り返す
    strokeWeight(i); //変数iの太さで図形を描く
    line(0,0,0,-8); //座標(0,0)から(0,-8)に直線をを引く
    translate(0,-8); //Y軸方向へ-8ずらす（上側に8ピクセル移動する）
    rotate(radians(angle)); //座標をangle度回転させる
  }
  popMatrix(); //pushMatrixで保存した座標の戻す
}
```

サンプルプログラム7

(tilt_sensor_7.pde)

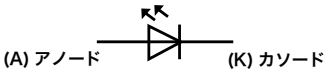
構造文tail内で、8ピクセルごとに描く線を上側に移動させ、回転させることで、曲線を描いている。また、センサーの値に応じて、回転量を変化させている。



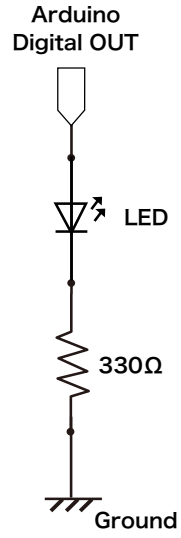
```
import processing.serial.*; //Arduinoを扱う際に必要
import cc.arduino.*; //Arduinoを扱う際に必要
Arduino arduino; //Arduinoを扱う際に必要

int a;
int b;
float x=200;
float y=200;

void setup() {//setupは一度しか実行されない
  //キャンバスのサイズを幅470ピクセル、高さ480ピクセルにする
  size(470, 480);
  //Arduinoが接続されているポートを指定する(数字を変更すること)
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  smooth(); //グラフィックを滑らかにする
  colorMode(HSB); //色をHSB(色相、彩度、輝度）で指定する
  frameRate(30); //一秒間にアニメーションを表示する回数
  background(255); //画面を白で塗りつぶす
}

void draw() {//draw内はプログラムを終了しない限り止まらない
  //変数aにアナログポート0に接続されたセンサーの現在値を代入する
  a = arduino.analogRead(0) ;
  //変数bにアナログポート1に接続されたセンサーの現在値を代入する
  b = arduino.analogRead(1) ;
  //変数aの値をコンソールに表示。プログラムの実行には影響しない
  println(a);
  //センサーの中間点を530とし、それより多い場合は変数を増加し、
  //少ない場合は変数を減少させる
  //10で割っているのは、一回あたりの増減量を調節するため
  x = x + (530 -a) / 10;
  y = y - (530 -b) / 10;
  //座標の原点を(x,y)に設定する
  translate(x,y);
  noFill();  //図形の塗りつぶしを行わない
  strokeWeight(5);  //線の太さを5ピクセルにすす
 //線の色を設定する。彩度は255、明度は220、透明度は30とし、
 //色相を530を中心にセンサーの値に応じて変化させる
 //absは絶対値を出力するための関数（色相はマイナスでの指定が出来ないため）
  stroke(abs(530-a),255,220,30);

  ellipse(0,0,50,50);  //半径50ピクセルの円を原点に描く
  //もし、変数xが0以下になった場合、画面外にはみ出したと判断し、xにwidth(この場合は470)を代入する
  if(x < 0) x = width;
  //もし、変数yが0以下になった場合、 画面外にはみ出したと判断し、yにheight(この場合は480)を代入する
  if(y < 0) y = height;
  //もし、変数xがwidth以上(470以上)になった場合、 画面外にはみ出したと判断し、xに0を代入する
  if(x > width) x = 0;
  //もし、変数ｙがheight以上(480以上)になった場合、画面外にはみ出したと判断し、yに0を代入する
  if(y > height) y =0 ;
}
```

**LEDを接続してみよう（デジタル出力）

●必要な部品　→　LED、抵抗(330Ω、橙橙茶金)

名称	発光ダイオード、LED(Light Emitting Diode)
部品の写真	
回路記号と接続先	 一般的には足の長い方が、アノードです
用途・機能	照明、信号等
特徴	長寿命、低消費電力、小型
注意点	LEDには極性が有り、逆に接続すると点灯しません。 必ず抵抗を挿入すること

LEDを接続するための回路図



デジタル出力ポートは、ポート3からポート13まで使用可能(ポート1と2は使用出来ません)

LEDを使用したサンプルプログラム1

(led_sample1.pde)　LEDはデジタル出力ポート3に接続
画面をクリックしたときのみポート3に接続したLEDを点灯

```
import processing.serial.*;
import cc.arduino.*;
Arduino arduino;

void setup(){
  size(470, 480);
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  frameRate(30);
  background(255);
  arduino.pinMode(3, arduino.OUTPUT);  //デジタルポート三番ピンをアウトプットとして使用する
}

void draw(){

}

void mousePressed() { //マウスボタンを押したとき
   arduino.digitalWrite(3, arduino.HIGH); //　デジタルポートの三番ピンをHIGH(ON)にする
}

void mouseReleased() {//マウスボタンを離したとき
  arduino.digitalWrite(3, arduino.LOW); //　デジタルポートの三番ピンをLOW(OFF)にする
}
```

LEDを使用したサンプルプログラム2
(led_sample2.pde) LEDはデジタル出力ポート3とポート4に接続
複数のLEDのコントロールを行う

```
import processing.serial.*;
import cc.arduino.*;
Arduino arduino;

void setup(){
  size(470, 480);
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  smooth();
  colorMode(HSB);
  frameRate(30);
  background(255);
  arduino.pinMode(3, arduino.OUTPUT);   //デジタルポート三番ピンをアウトプットとして使用する
  arduino.pinMode(4, arduino.OUTPUT);   //デジタルポート四番ピンをアウトプットとして使用する
}

void draw()
{

}

void mousePressed() {      //マウスボタンを押したとき
   arduino.digitalWrite(3, arduino.HIGH);  //デジタルポートの三番ピンをHIGH(ON)にする
   arduino.digitalWrite(4, arduino.LOW);  //デジタルポートの四番ピンをLOW(OFF)にする
}

void mouseReleased() {   //マウスボタンを離したとき
  arduino.digitalWrite(3, arduino.LOW);   //デジタルポートの三番ピンをLOW(OFF)にする
  arduino.digitalWrite(4, arduino.HIGH);  //デジタルポートの四番ピンを HIGH(ON) にする
```

デジタル出力を行う場合のProcessingのプログラム記述方法

arduino.pinMode(ポート番号, arduino.OUTPUT);

デジタル出力を行うことを宣言する。void setup()内に挿入すること。
（この命令をvoid setup()内に記述しなければ、デジタルポートとして使用できない）。
ポート番号には3から13までの値を記述する（ポート番号1と2はデジタル出力として使用できない）。
ちなみに、デジタル入力を行う場合は、arduino.OUTPUTをarduino.INPUTに変更する。
例）
arduino.pinMode(8, arduino.OUTPUT);
ポート8をデジタル出力ポートとして使用する

arduino.digitalWrite(ポート番号, 値);

ポートのオン・オフを行う。この命令が実行された瞬間、ポートのオンとオフが切り替わる。
ポート番号には3から13までの値を記述する（ポート番号1と2はデジタル出力として使用できない）。
値にはオンの場合、**arduino.HIGH**、オフの場合、**arduino.LOW**を記述する。
例）
arduino.digitalWrite(6, arduino.HIGH);
デジタルポート6をONにする

LEDを使用したサンプルプログラム3

(led_sample3.pde) LEDはデジタル出力ポート3に接続

画面内の特定の場所をクリックした時のみLEDが点灯

```
import processing.serial.*;
import cc.arduino.*;
Arduino arduino;

void setup(){
  size(470, 480);
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  frameRate(30);
  background(255);
  arduino.pinMode(3, arduino.OUTPUT);
}

void draw()
{
  rect(100,100,100,100); //ボタン用の図形を描写
}

void mousePressed() {
//マウスの現在のX座標が、100以上かつ200以下であり、Y座標が100以上かつ200以下の場合、if文内を実行する
  if(mouseX > 100 && mouseX < 200 && mouseY > 100 && mouseY < 200){
    arduino.digitalWrite(3, arduino.HIGH);
  }
}

void mouseReleased() {
//マウスの現在のX座標が、100以上かつ200以下であり、Y座標が100以上かつ200以下の場合、if文内を実行する
  if(mouseX > 100 && mouseX < 200 && mouseY > 100 && mouseY < 200){
  arduino.digitalWrite(3, arduino.LOW);
  }
}
```

LEDを使用したサンプルプログラム4

(led_sample4.pde) LEDはデジタル出力ポート3に接続

画面内の特定の場所をクリックした時にLEDのオン・オフが切り替わる

```
import processing.serial.*;
import cc.arduino.*;
Arduino arduino;

int button = 0; //現在のボタンの状況を変数で表す。0の場合オフ、1の場合オンとする

void setup(){
  size(470, 480);
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  frameRate(30);
  background(255);
  arduino.pinMode(3, arduino.OUTPUT);
}

void draw()
{
  //buttonが1の場合、塗りつぶし色を黒に、ポート3をオンにする
  //そうではない場合、塗りつぶし色を白に、 ポート3をオフにする
  if(button ==1){
    fill(0);
    arduino.digitalWrite(3, arduino.HIGH);
  }else{
    fill(255);
    arduino.digitalWrite(3, arduino.LOW);
  }

  rect(100,100,100,100); //ボタン用の図形を描写

}

void mousePressed() {
    //マウスの現在のX座標が、100以上かつ200以下であり、Y座標が100以上かつ200以下の場合、if文内を実行する
    if(mouseX > 100 && mouseX < 200 && mouseY > 100 && mouseY < 200){
      //下記のIF文ではbuttonの値が1の時は0に変更、
      //buttonの値が0の時は1に変更している
      //つまり、オンとオフの切り替えを行っている
      if(button ==1){
        button = 0;
      }else{
        button = 1;
      }
    }
  }
```

LEDを使用したサンプルプログラム5

(led_sample5.pde) LEDはデジタル出力ポート3に接続

LEDが1秒ごとにオン・オフを繰り返す

```
import processing.serial.*;
import cc.arduino.*;
Arduino arduino;

void setup(){
  size(470, 480);
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  frameRate(30);
  background(255);
  arduino.pinMode(3, arduino.OUTPUT);
}

void draw()
{
arduino.digitalWrite(3, arduino.HIGH);
delay(1000);    //1000msec（1秒）待つ
arduino.digitalWrite(3, arduino.LOW);
delay(1000);    //1000msec（1秒）待つ
}
```

LEDを使用したサンプルプログラム6

(led_sample6.pde)

LEDはデジタル出力ポート3とポート4に接続

二つのLEDが0.5秒ごとに交互に点灯する

```
import processing.serial.*;
import cc.arduino.*;
Arduino arduino;

void setup(){
  size(470, 480);
  println(Arduino.list());
  arduino = new Arduino(this, Arduino.list()[0], 57600);
  frameRate(30);
  background(255);
  arduino.pinMode(3, arduino.OUTPUT);
  arduino.pinMode(4, arduino.OUTPUT);
}

void draw()
{
arduino.digitalWrite(3, arduino.HIGH);
arduino.digitalWrite(4, arduino.LOW);
delay(500);   //500msec（0.5秒）待つ
arduino.digitalWrite(3, arduino.LOW);
arduino.digitalWrite(4, arduino.HIGH);
delay(500); //500msec（0.5秒）待つ
}
```