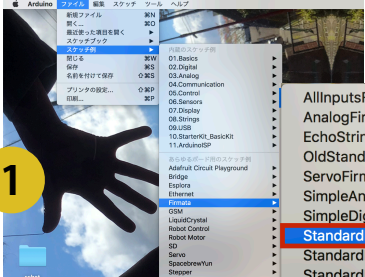
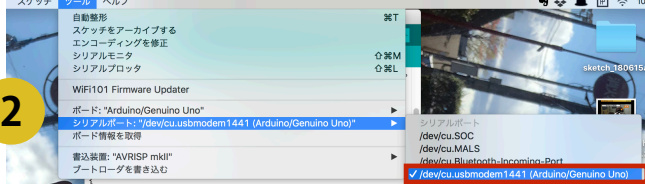


[演習]
Arduino - Processing
アプリケーション間通信 (FIRMATA)


Firmataの導入 (Arduino側)



1 AllInputsFirmata
AnalogFirmata
EchoString
OldStandardFirmata
ServoFirmata
SimpleAnalogFirmata
SimpleDigitalFirmata
StandardFirmata
StandardFirmataBLE
StandardFirmataChinKIT



2 ボード: "Arduino/Genuino Uno"
シリアルポート: "/dev/cu.usbmodem1441 (Arduino/Genuino Uno)"
ポート情報を取得
書き装置: "AVRISP mkII"
ブートローダを書き込む




3

ファイル/スケッチ例より StandardFirmata を開きます。

適切なシリアルポートを選択するとともに、ポート名を覚えておいてください。

USBを介してArduinoを接続し、(ファイルを一切編集することなく) 書き込みを実行してください。

Firmataの導入 (Processing側)

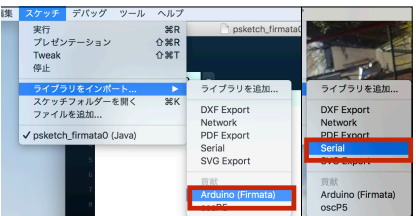


1

Librariesタブで, firmataとタイプして, 対応するライブラリをインストールしてください。

スケッチ/ライブラリをインポート/ライブラリを追加

スケッチ/ライブラリをインポート/Arduino (Firmata)




2 **Arduino (Firmata) とSerialの二つのライブラリをインポートしてください。**

```

1 import processing.serial.*;
2 import cc.arduino.*;
3 import org.firmata.*;
4
    
```

Firmataの動作確認 (Processing側)



```

1 import processing.serial.*;
2 import cc.arduino.*;
3 import org.firmata.*;
4
5 //Arduino arduino;
6 //int ledPin = 13;
7
8 void setup(){
9
10   println(arduino.list());
11
12   //arduino = new Arduino(this, Arduino.list()[0], 5);
13   //arduino.pinMode(ledPin, Arduino.PIN_OUTPUT);
14 }
15
16 void draw(){
17 }
    
```

0 /dev/cu.usbmodem1461
1 /dev/cu.usbmodem1461
2 /dev/cu.usbmodem1461
3 /dev/cu.usbmodem1461
4 /dev/cu.usbmodem1461
5 /dev/cu.usbmodem1461
6 /dev/cu.usbmodem1461
7 /dev/cu.usbmodem1461

まず, Arduinoを接続しているシリアルポートと一致する<添字>を探してください。

このケースでは, 4番目(添字としては3)のシリアルポート名が一致しています。この3という数字を覚えておきます。

Firmataの動作確認 (Processing側)

psketch_LED.pde

```

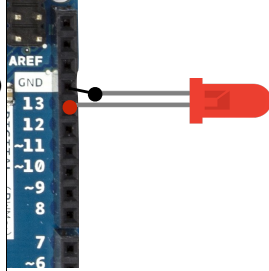
1 import processing.serial.*;
2 import cc.arduino.*;
3 import org.firmata.*;
4
5 Arduino arduino;
6 int ledPin = 13;
7
8 void setup(){
9   //println(arduino.list());
10  arduino = new Arduino(this, arduino.list()[3], 57600);
11  arduino.pinMode(ledPin, Arduino.OUTPUT);
12 }
13
14 void draw(){
15 }
16
17 void mousePressed(){
18   arduino.digitalWrite(ledPin, Arduino.HIGH);
19   background(color(255,0,0));
20 }
21
22 void mouseReleased(){
23   arduino.digitalWrite(ledPin, Arduino.LOW);
24   background(color(0));
25 }

```

マウスを押す／離すによって、LEDのON・OFFが切り替わり、同時に画面の背景も黒から赤に変わります。なお、実行の際には、Arduinoは終了しておきません (USBをProcessingのために解放するためです)。

Arduino型のobjectの宣言。以下で、Arduinoのソフトウェアで使用していたメソッドは、このobjectのインスタンスメソッドとして使用することができます。

先に覚えておいた添字に合わせます。



Arduinoのソフトウェアで使用していたシステム変数 (OUTPUT, HIGH, LOW, ...) は、Arduinoクラスのクラス変数として使用できます。

圧力センサの値をビジュアライズする

psketch_Touch.pde

```

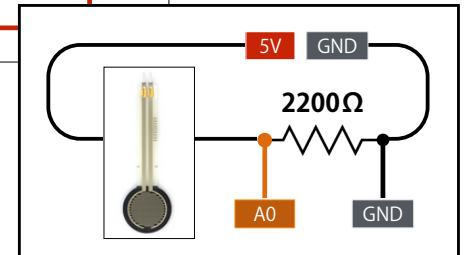
1 import processing.serial.*;
2 import cc.arduino.*;
3 import org.firmata.*;
4
5 Arduino arduino;
6 int sensorPin = 0; //A0に対応
7
8 void setup(){
9   //println(arduino.list());
10  arduino = new Arduino(this, arduino.list()[3], 57600);
11  arduino.pinMode(sensorPin, Arduino.INPUT);
12  size(800,100);
13 }
14
15 void draw(){
16   float val = arduino.analogRead(sensorPin);
17   println(4.63 * val / 1023.);
18
19   background(0); fill(255,100,0); stroke(255);
20   rect(0,0,width*val/1024.,height);
21 }
22 }

```

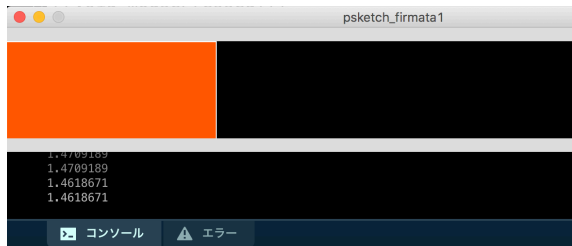
ArduinoのANピンは、Processingでは整数型のNに対応します。

0からMax (V) を0から1023のレンジで読み込みます。この例でのMaxは4.63Vです。

背景は黒、塗りつぶしオレンジ、枠線の色は白で、valに応じて長方形の幅を変化させています。



圧力センサの値をビジュアライズする



実行結果

圧力センサを押すと、バーの長さが伸び縮みます。



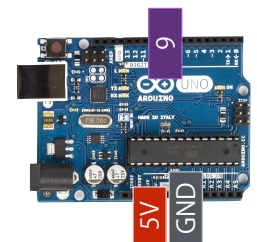
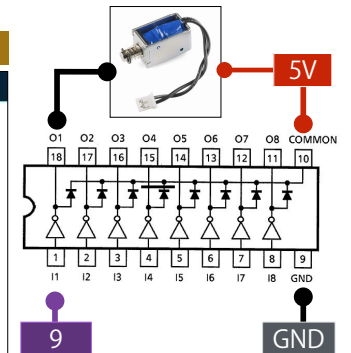
ソレノイドをGUIで動かす

psketch_solenoid.pde (1/2)

```

1 psketch_solenoid
2 import processing.serial.*;
3 import cc.arduino.*;
4 import org.firmata.*;
5
6 Arduino arduino;
7 int sorPin = 9;
8 int val = 255; //ソレノイドの出力値
9 float cx, cy; //中点座標
10
11 float t_last = 0; //ソレノイドの最近の出力時刻 (ms)
12 float space = 200; //ソレノイドの休符時間長 (ms)
13
14 void setup(){
15   arduino = new Arduino(this, arduino.list()[3], 57600);
16   arduino.pinMode(sorPin, Arduino.OUTPUT);
17
18   size(600,600); background(0);
19   cx = 0.5 * width; cy = 0.5 * height;
20 }

```




```

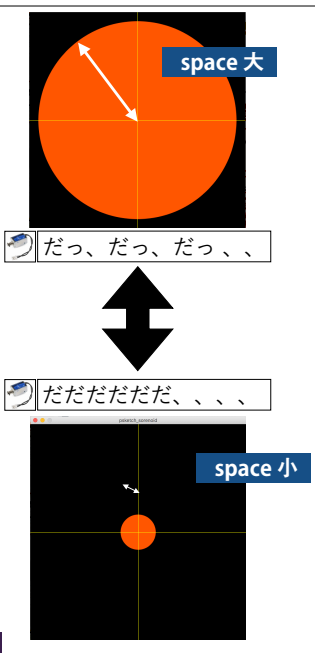
21 void draw(){
22     float t = millis() - t_last; //経過時間
23     //経過時間が休符時間長を超えた場合
24     if(t>space){
25         draw()
26
27         if(val==255){
28             val = 0;
29         }else{
30             val = 255;
31         }
32         arduino.analogWrite(sorPin, val);
33         t_last = millis(); //出力時刻の更新
34     }
35 }
36
37
38
39 if(mousePressed){
40     float rad = dist(mouseX, mouseY, cx, cy);
41     background(0); fill(255,100,0); noStroke();
42     ellipse(cx,cy,2*rad,2*rad);
43     stroke(255,255,0);
44     line(cx,0,cx,height); line(0,cy,width,cy);
45     space = rad;
46 }
47
48
49
50
51

```

前回ソレノイドの値を変えた時刻からの経過時間 t をモニタします。

経過時間が space を超えた場合、ソレノイドの値を反転し、t_last を現在の時間(ms)に更新します。

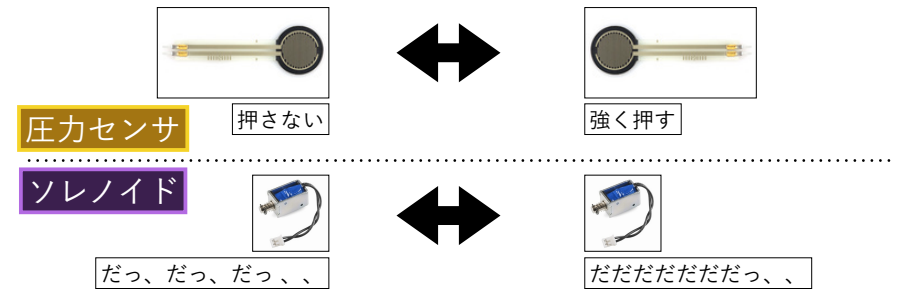
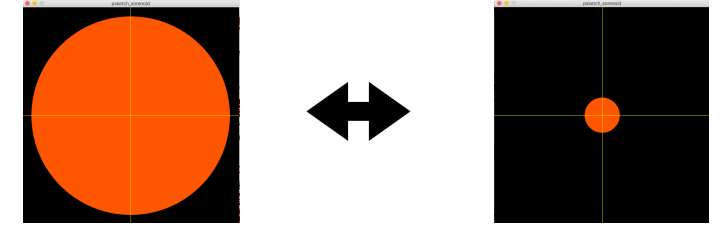
中心からマウスまでの距離に応じて、円の大きさと space を変えます。



psketch_sorenoid.pde (2/2)

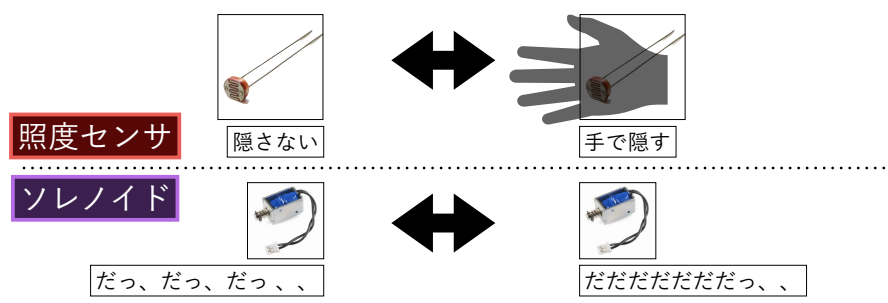
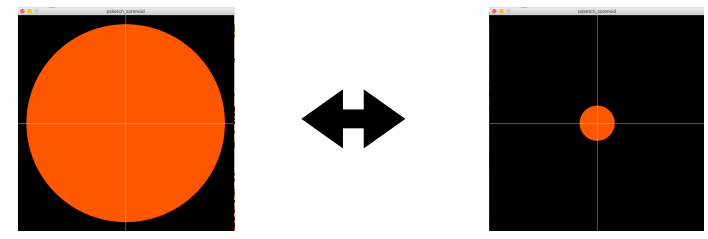
練習 1

圧力センサを押すと、GUI上の円が収縮し、それに伴い、ソレノイドの打撃のスピードが高まるようなプログラムを作成してください。



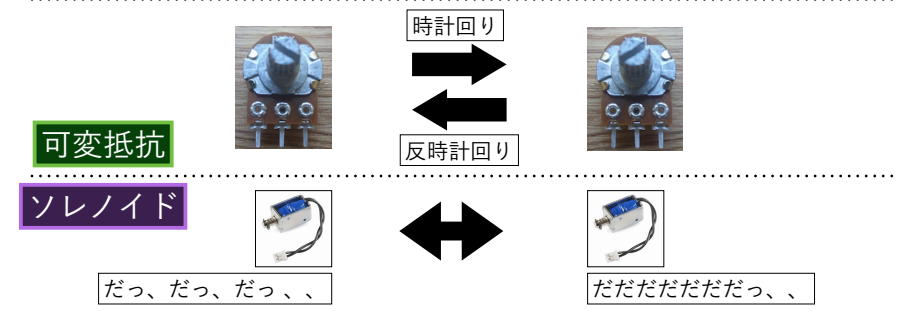
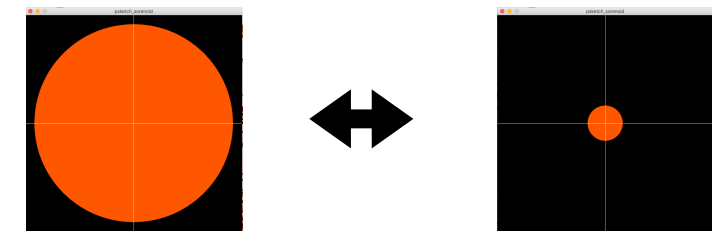
練習 2

照度センサを手で隠すと、GUI上の円が収縮し、それに伴い、ソレノイドの打撃のスピードが高まるようなプログラムを作成してください。



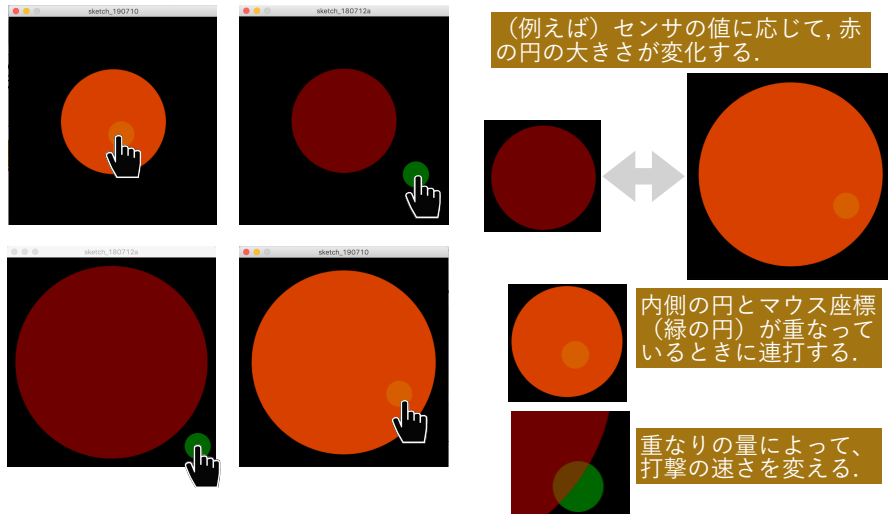
練習 3

可変抵抗を時計回りに回すと、GUI上の円の半径が収縮し、それに伴い、ソレノイドの打撃のスピードが高まるようなプログラムを作成してください。



課題

マウスの座標とセンサの値に応じてソレノイドの打撃パターンが変化するリズムマシンを作って、演奏してください。また、この際、Processing上で連打している様子がビジュアルとしても伝わるように工夫してください。以下はあくまでも例です。アニメーション表現は各自で工夫すること。



参考資料

正負の電圧を扱う場合 (DCモータ)

モーターフェーダー (正回転・逆回転)



スイッチサイエンス：2253円

型番	SFE-COM-10976
標準電圧	6V~11V
最大電流	400mA~800mA

<http://www.switch-science.com/catalog/1285/>

リニアアクチュエータ (伸長・収縮)



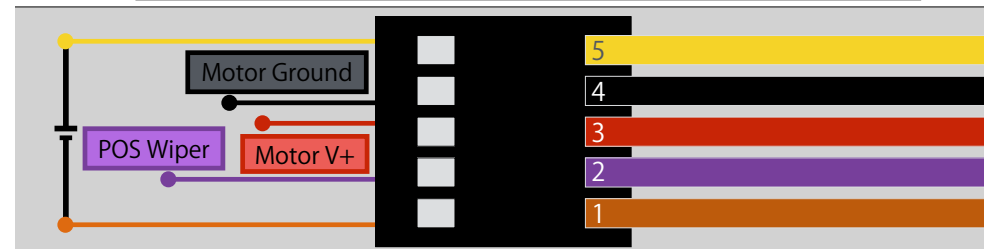
Firgelli：80\$

型番	L16-P Linear Actuator
Input Volatage	12V
Stroke	50 / 100 / 140 mm
Gear	35:1 / 63:1 / 150:1

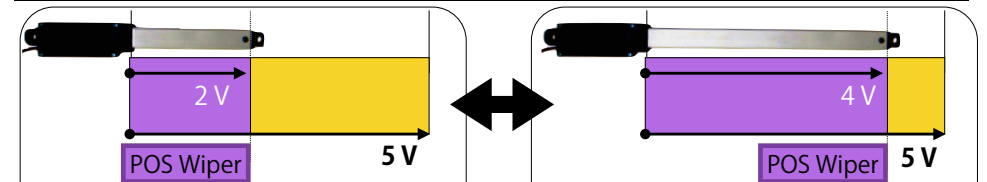
<https://www.firgelliauto.com/>

WIRING: (see last page for pin numbering)

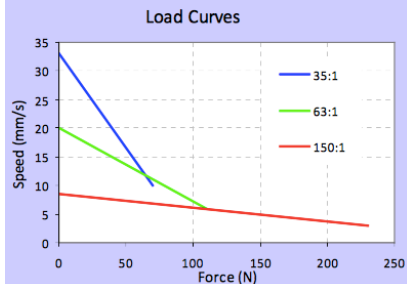
- 1 - Orange – Feedback Potentiometer negative reference rail
- 2 - Purple – Feedback Potentiometer wiper
- 3 - Red – Motor V+ (6V or 12V)
- 4 - Black – Motor V- (Ground)
- 5 - Yellow – Feedback Potentiometer positive reference rail



端子5に5Vの接続がある場合、アクチュエータの位置に応じて端子2の電圧が0-5Vで変化する。



L16 Specifications



Firgelli L16のデータシート

L16 Specifications			
Gearing Option	35:1	63:1	150:1
Peak Power Point	50N @16mm/s	75N @10mm/s	175N @4mm/s
Peak Efficiency Point	24N @24mm/s	38N @15mm/s	75N @7mm/s
Max Speed (no load)	32mm/s	20mm/s	8mm/s
Max Force (lifted)	50N	100N	200N
Back Drive Force	31N	46N	102N
Stroke Option	50mm	100mm	140mm
Mass	56g	74g	84g
Positional Accuracy	0.3mm	0.4mm	0.5mm
Max Side Load (extended)	40N	30N	20N
Feedback Potentiometer	9kΩ±30%	18kΩ±30%	25kΩ±30%
Electrical Stroke	48mm	98mm	138mm
Input Voltage	0-15 VDC. Rated at 12VDC.		
Stall Current	650mA @ 12V		
Operating Temperature	-10°C to +50°C		
Audible Noise	60dB @ 45cm		
Ingress Protection	IP-54		
Mechanical Backlash	0.2mm		
Limit Switches	Max. Current Leakage: 8uA		

L16-SS-GG-VV-C

Feature	Options
SS: Stroke	50, 100, 140 (mm)
GG: Gear reduction ratio (refer to load curves above)	35, 63, 150 (lower ratios are faster but push less force, and vice versa)
VV: Voltage	12 vdc or 6 vdc (-R only)
C: Controller	P Potentiometer Feedback S Limit Switches R RC Linear Servo

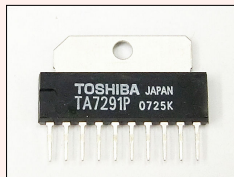
モータドライバ

■ モータは通常、一方の端子に、正の電圧を与えると正回転（伸長）、負の電圧を与えると逆回転（収縮）する。モータドライバは、これを二つの入力端子に対する正の電圧の増減によって制御することを可能とします。

■ モータドライバにはトランジスタが内蔵されており、一般的なモータを動かすのに十分な電流を扱うことができます。

TA7291P 正逆切替モータドライバ（東芝）

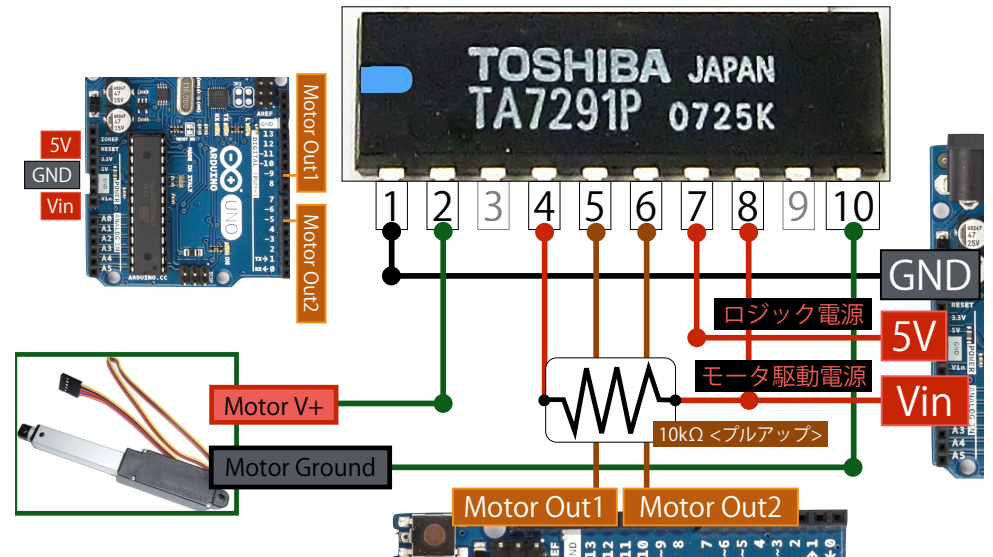
2個300円（秋月）



モータ用電源	0 ~ 20V
ロジック用電源	4.5V ~ 20V
出力電流	1A (max 2A)

モータドライバとArduinoの接続例

- ArduinoからPWMの出力ピンを2つ使って、一方の電圧で正回転の回転量、もう一方の電圧で逆方向の回転量を制御します。
- モータ駆動電源は、使用するモータの入力電圧と合わせます。



Arduinoへの書き込みの例（スタンドアロン）

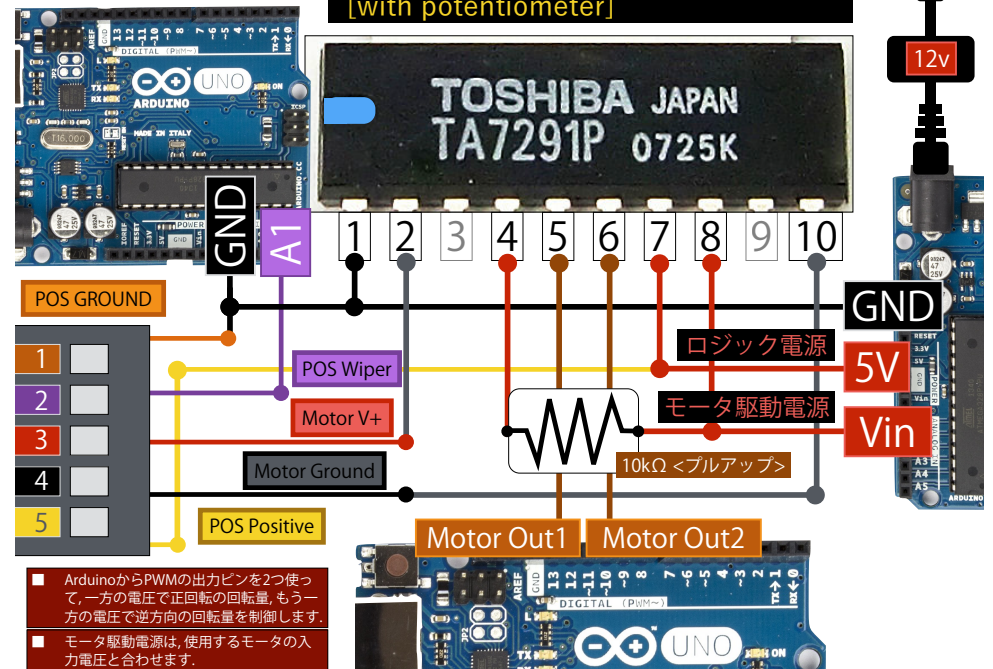
```

asketch_SimpleMotor | Arduino 1.0.5
asketch_SimpleMotor
//モータをコントロールするピン
一方が正回転用・もう一方が逆回転用（PWM）
int motorPin1 = 5; int motorPin2 = 6;

void setup(){
  //vibePinを出力モードとします。
  pinMode(motorPin1, OUTPUT);
  pinMode(motorPin2, OUTPUT);
}

void loop(){
  正回転と逆回転を2秒おきに繰り返します。
  analogWrite(motorPin1,255); analogWrite(motorPin2,0);
  delay(2000);
  analogWrite(motorPin1,0); analogWrite(motorPin2,255);
  delay(2000);
}
    
```

Firgelli L16とArduinoの接続例 [with potentiometer]



- ArduinoからPWMの出力ピンを2つ使って、一方の電圧で正回転の回転量、もう一方の電圧で逆方向の回転量を制御します。
- モータ駆動電源は、使用するモータの入力電圧と合わせます。