THE BASIC STARTER KIT TUTORIAL

FOR UNO

V1.0.19.7.24

序文

Elegoo は 2011 年に設立され、オープンソースのハードウェアの研究開発、生産、マーケティングに特化し、繁栄したテクノロジー企業です。 深セン、中国のシリコンバレーに位置し、我々は 10,763 +平方フィートの工場で、従業員は 150 人以上です。
当社の製品ラインは、デュポンのワイヤー、UNO R3 ボードから Arduino の知識を学ぶためのあらゆるレベルのお客様向けに設計されたスターターキットを完成させました。 また、2.8 インチ TFT タッチと STM32 のようなラズベリーパイアクセサリーの製品も販売しています。 将来的には、3D プリンタ製品などに多くのエネルギーと投資を投入します。 当社の製品はすべて国際的な品質基準に準拠しており、世界中のさまざまな市場で高く評価されています。

公式サイト:http://www.elegoo.com

US Amazon storefront: http://www.amazon.com/shops/A2WWHQ25ENKVJ1

CA Amazon storefront: http://www.amazon.ca/shops/A2WWHQ25ENKVJ1

UK Amazon storefront: http://www.amazon.co.uk/shops/AZF7WYXU5ZANW

DE Amazon storefront: http://www.amazon.de/shops/AZF7WYXU5ZANW

FR Amazon storefront: http://www.amazon.fr/shops/AZF7WYXU5ZANW

ES Amazon storefront: http://www.amazon.es/shops/AZF7WYXU5ZANW

IT Amazon storefront: http://www.amazon.it/shops/AZF7WYXU5ZANW

JP Amazon storefront: http://www.amazon.co.jp/shops/A21X7DQBM2LL85

この教材

このチ教材は初心者のためのものです。 Arduino コントローラボード、センサー、およびコンポーネントの使用方法に関する基本情報をすべて学びます。 Arduino をさらに深く勉強したい場合は、Michael Margolis が書いた Arduino Cookbook を読むことをお勧めします。このチュートリアルの一部のコードは Simon Monk によって編集されています。 Simon Monk はオープンソースハードウェアに関する数多くの書籍の著者です。

Amazon で利用可能です：Arduino のプログラミング、Evil Genius のための 30 の Arduino プロジェクト、Raspberry Pi のプログラミング。

顧客サービス

絶えず急速に成長しているテクノロジー企業として、私たちは お客様の期待に応える優れた製品とた製品と質の高いサービスをご提供しており、service@elegoo.com または EUservice@elegoo.com までご連絡ください。 皆様からのご意見をお待ちしております。批判的なご意見やご提案は、私たちにとって大変貴重なものとなります。

また、製品に関する問題やご質問は、経験豊かなエンジニアが 12 時間以内に（24 時間休暇中）迅速に返信します。

Packing list

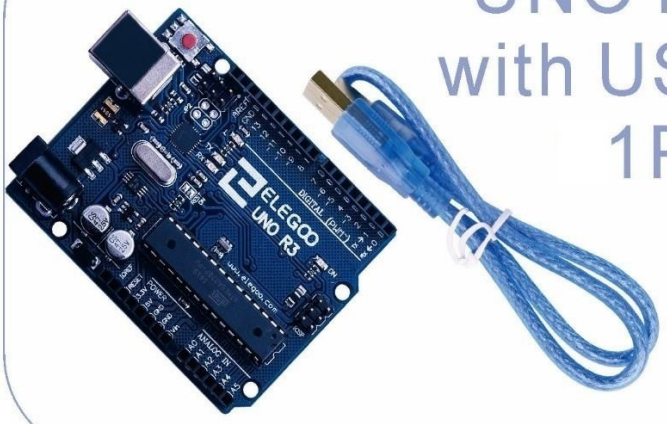
www.elegoo.com

RGB LED
2PCS

Photoresistor
(photocell)
1PC

Resistor
120PCS

UNO R3
with USB
1PC



Tilt Ball Switch
1PC

LED
30PCS

Button(Small)
5PCS

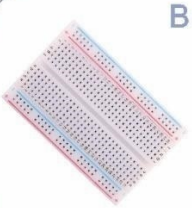
Active Buzzer
1PC

74HC595 IC
1PC

F-M Dupont Wire
5PCS

Breadboard
Jumper Wire
65PCS

Breadboard
1PC



Contact us : service@elegoo.com

Content

Lesson 0 IDE のインストール

前書き

Arduino 統合開発環境（IDE）は、Arduino プラットフォームのソフトウェア側です。

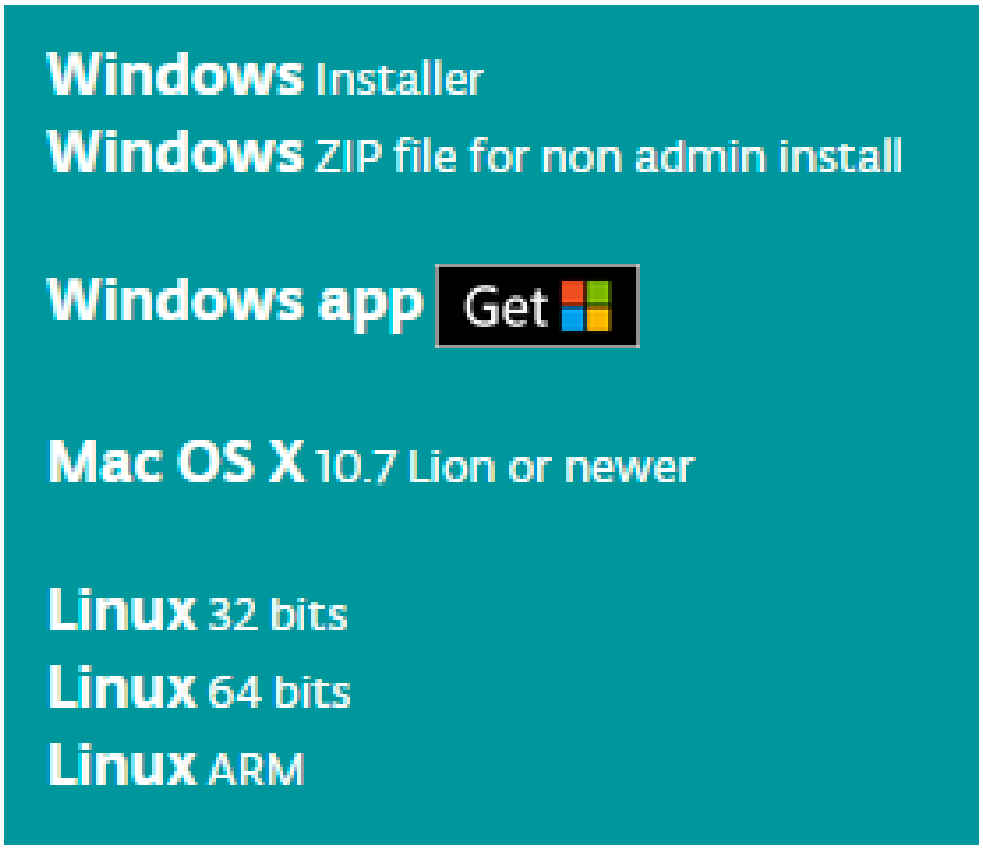
このレッスンでは、Arduino を使用するようにコンピュータを設定する方法と、それに続くレッスンについて設定する方法を学習します。

Arduino をプログラミングするために使用する Arduino ソフトウェアは、Windows、Mac、Linux で使用できます。 インストールプロセスは 3 つのプラットフォームすべてで異なります。残念ながら、ソフトウェアをインストールするには一定の作業が必要です。

STEP 1: Go to https://www.arduino.cc/en/Main/Software and find below page.

このウェブサイトで利用可能なバージョンは通常最新バージョンであり、実際のバージョンは画像のバージョンよりも新しい場合があります。

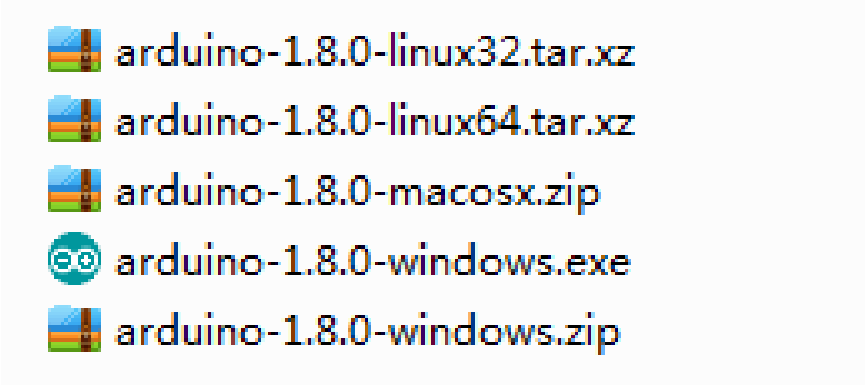
STEP2： コンピュータのオペレーティングシステムと互換性のある開発ソフトウェアをダウンロードします。ここで Windows を例に取る。



Click *Windows Installer*.

Click *JUSTDOWNLOAD*.

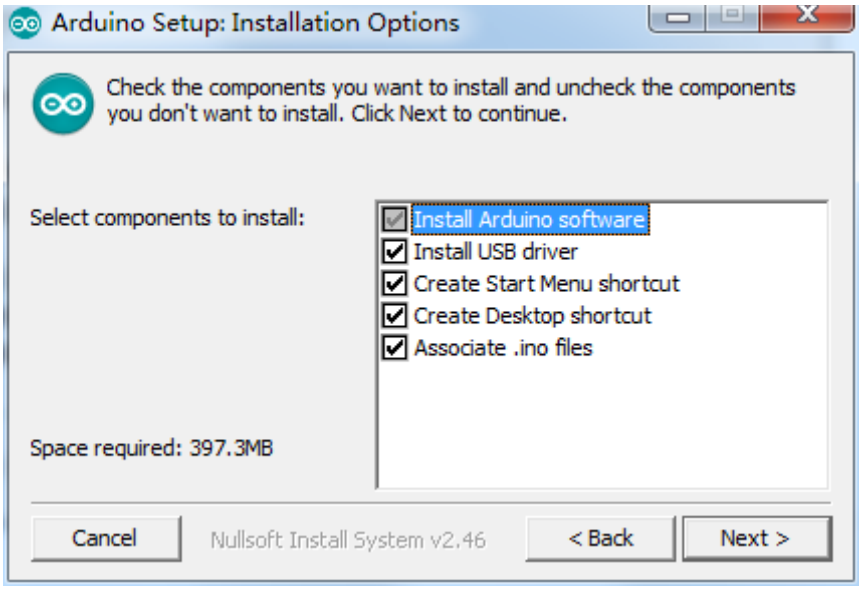
私たちが提供した資料でもバージョン 1.8.0 が利用可能で、この資料のバージョンはこのコースが作成されたときの最新バージョンです。



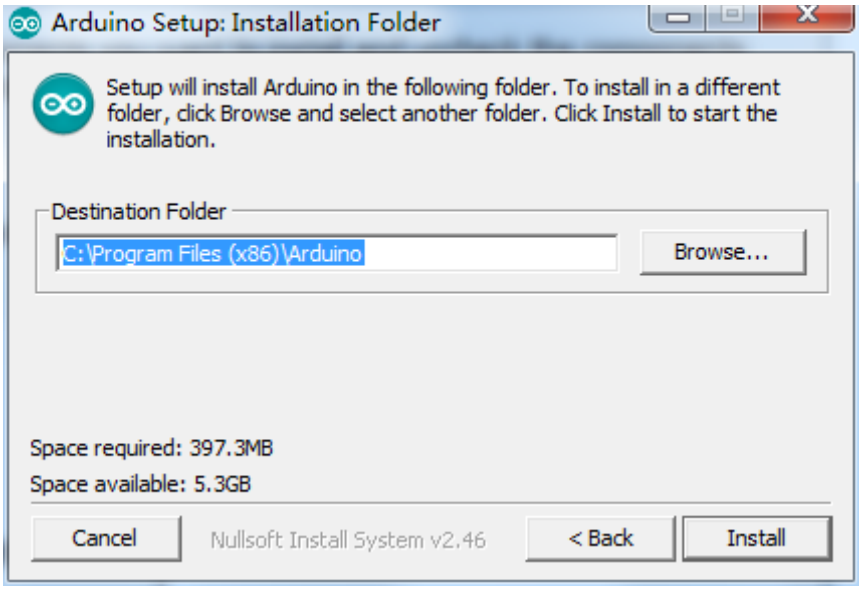
Installing Arduino (Windows)

xe で Arduino をインストールします。 インストールパッケージ.

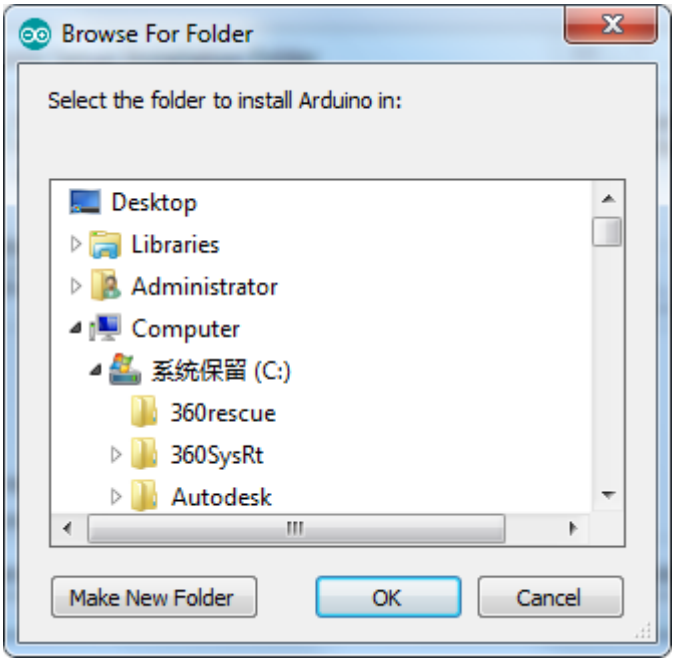
「同意する」をクリックすると、次のインターフェースが表示されます。



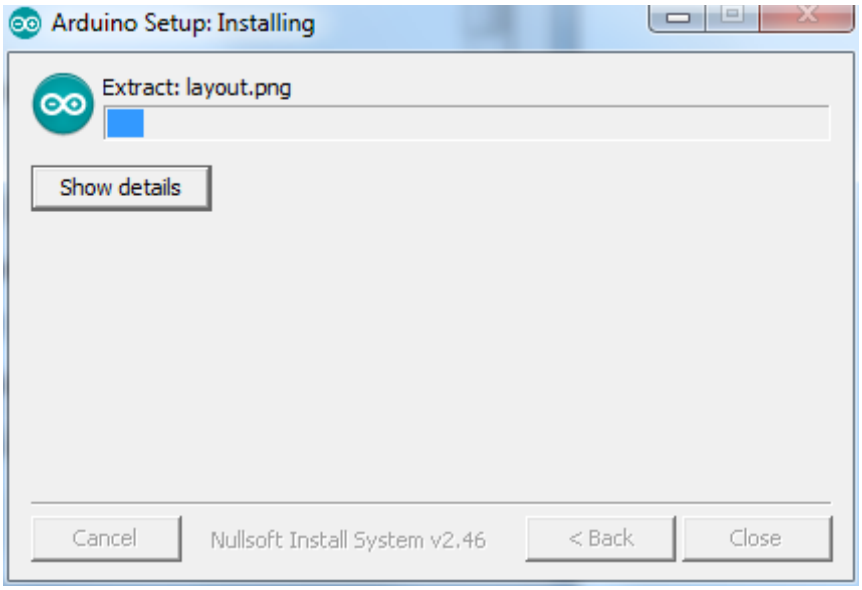
次へをクリックします



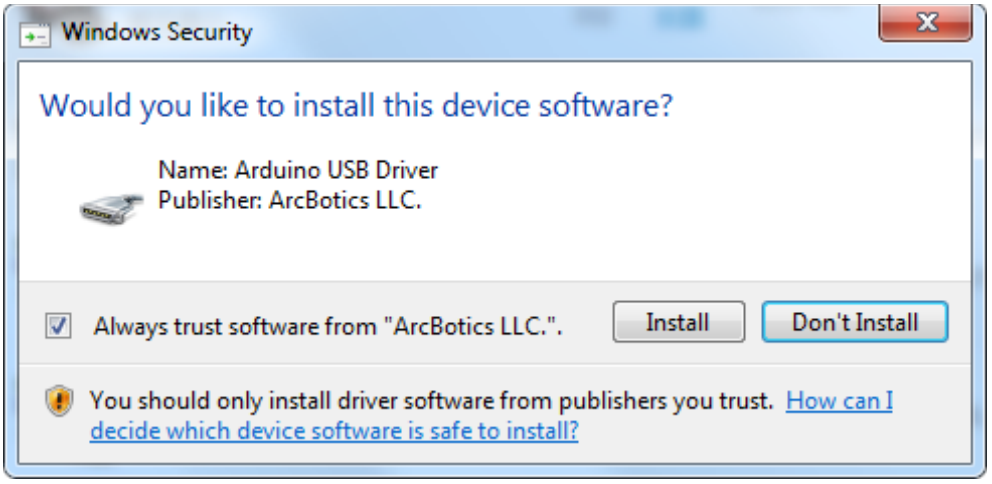
[参照]を押してインストールパスを選択するか、目的のディレクトリに直接入力することができます。



[インストール]をクリックしてインストールを開始します

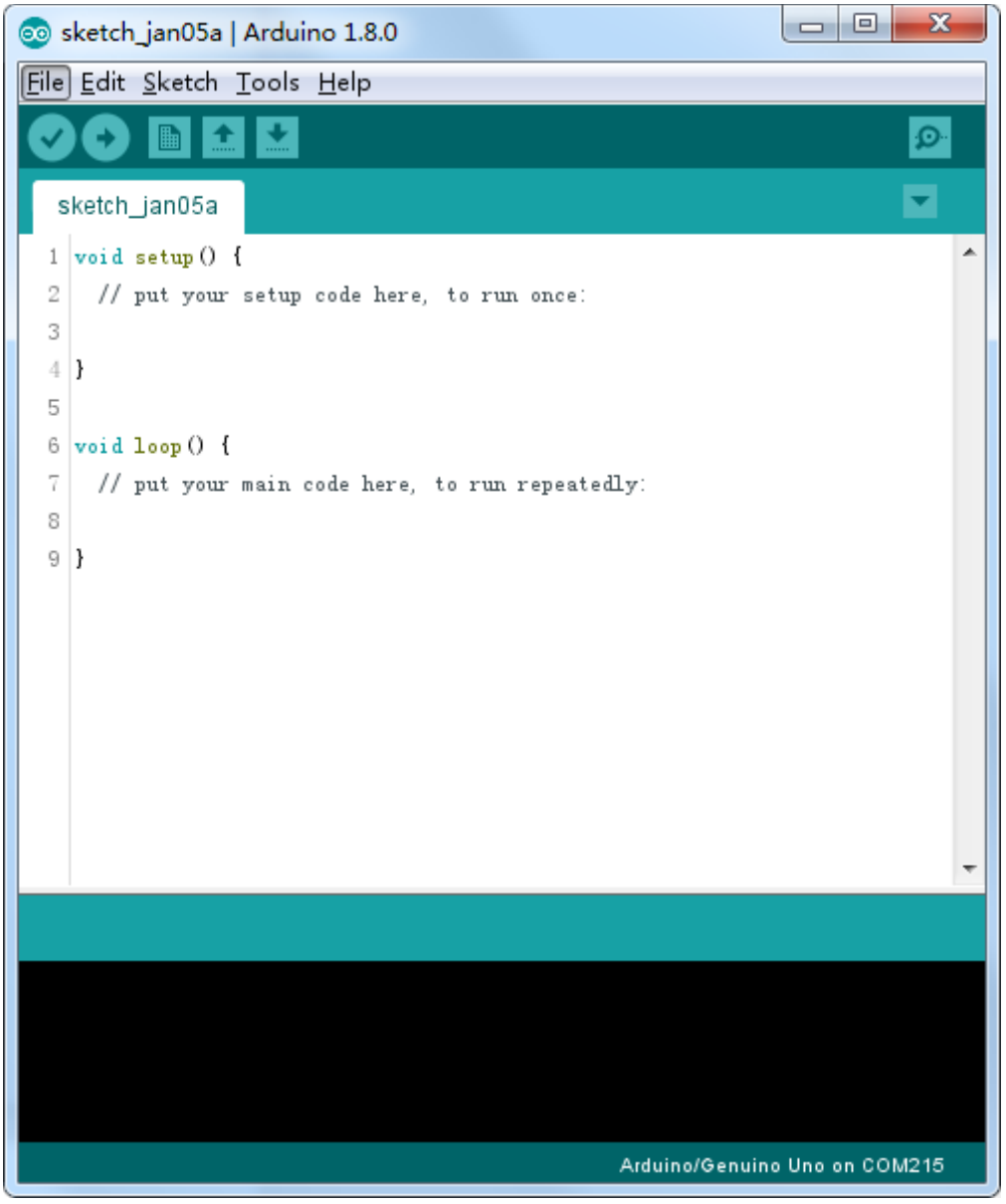


最後に、次のインターフェイスが表示されたら、[インストール]をクリックしてインストールを完了します。



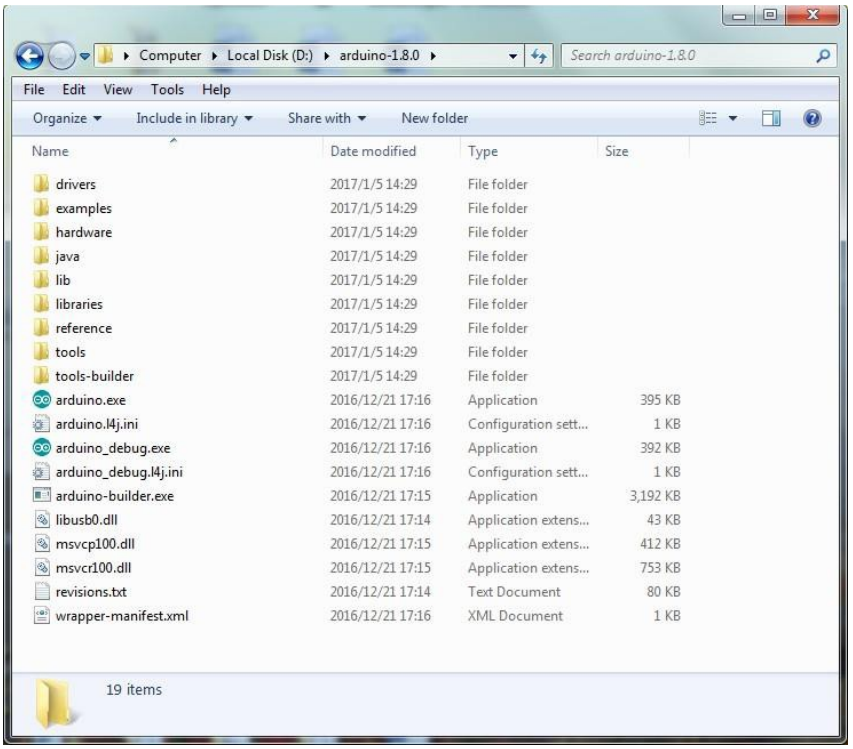
次に、デスクトップに次のアイコンが表示されます。

ダブルクリックして希望の開発環境に入ります

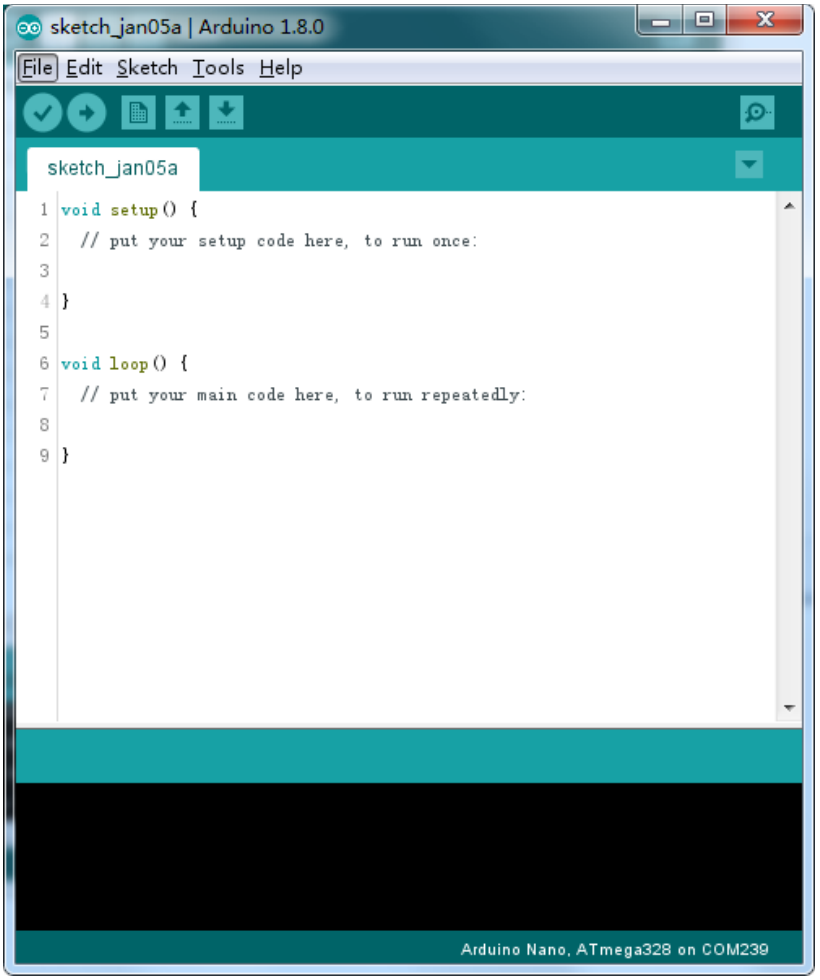


インストール用のインストールパッケージを直接選択して下の内容をスキップし、次のセクションにジャンプすることができます。 しかし、インストールパッケージ以外のいくつかの方法を学びたい場合は、引き続きこのセクションをお読みください。

ダウンロードした zip ファイルを解凍し、ダブルクリックしてプログラムを開き、希望する開発環境に入る。


Computer ▸ Local Disk (D:) ▸ arduino-1.8.0 ▸
Search arduino-1.8.0
File Edit View Tools Help
Organize Include in library Share with New folder
Name Date modified Type Size
drivers 2017/1/5 14:29 File folder
examples 2017/1/5 14:29 File folder
hardware 2017/1/5 14:29 File folder
java 2017/1/5 14:29 File folder
lib 2017/1/5 14:29 File folder
libraries 2017/1/5 14:29 File folder
reference 2017/1/5 14:29 File folder
tools 2017/1/5 14:29 File folder
tools-builder 2017/1/5 14:29 File folder
arduino.exe 2016/12/21 17:16 Application 395 KB
arduino.l4j.ini 2016/12/21 17:16 Configuration sett... 1 KB
arduino_debug.exe 2016/12/21 17:16 Application 392 KB
arduino_debug.l4j.ini 2016/12/21 17:16 Configuration sett... 1 KB
arduino-builder.exe 2016/12/21 17:15 Application 3,192 KB
libusb0.dll 2016/12/21 17:14 Application extens... 43 KB
msvcp100.dll 2016/12/21 17:15 Application extens... 412 KB
msvcr100.dll 2016/12/21 17:15 Application extens... 753 KB
revisions.txt 2016/12/21 17:14 Text Document 80 KB
wrapper-manifest.xml 2016/12/21 17:16 XML Document 1 KB
19 items


arduino.exe

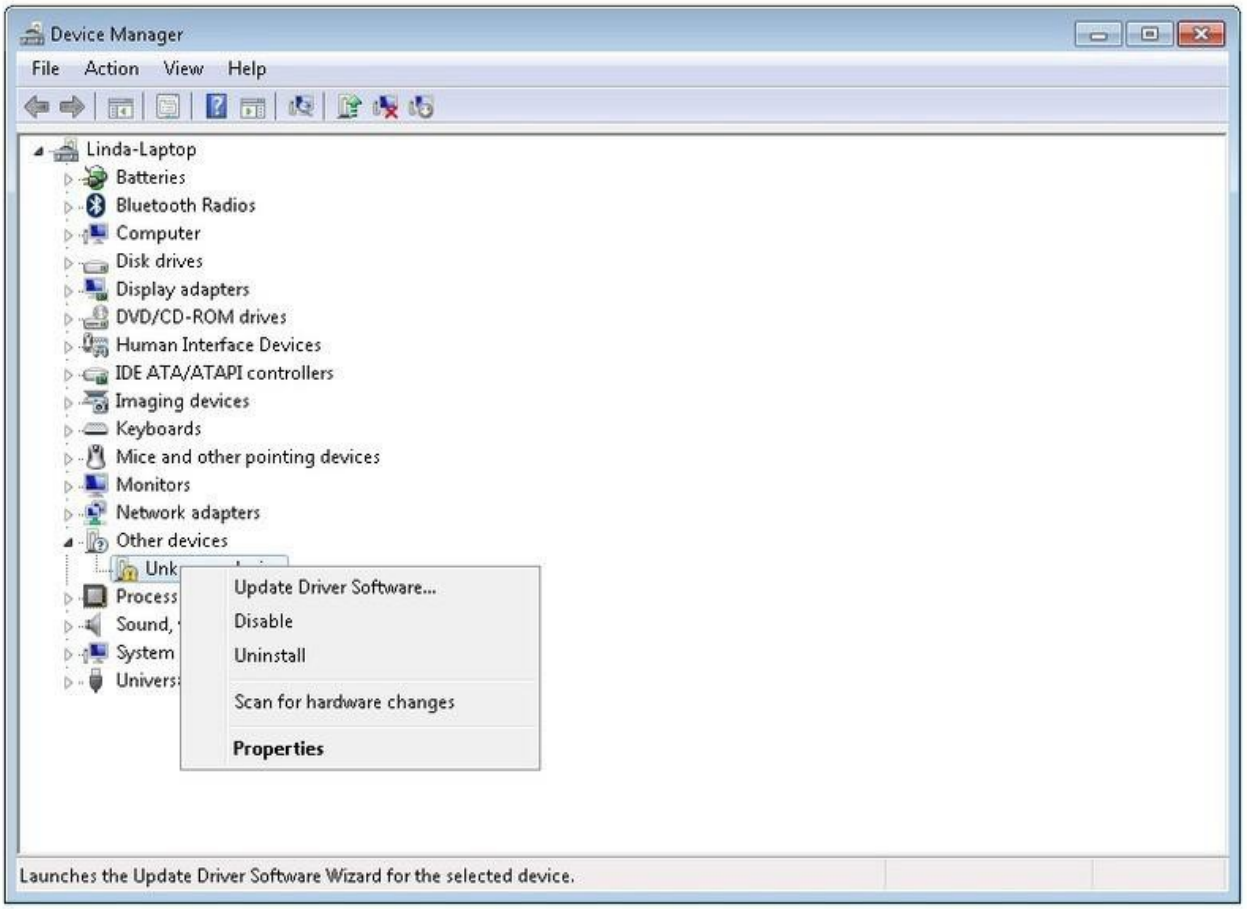

sketch_jan05a | Arduino 1.8.0
File Edit Sketch Tools Help
sketch_jan05a
1 void setup() {
2 // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7 // put your main code here, to run repeatedly:
8
9 }
Arduino Nano, ATmega328 on COM239

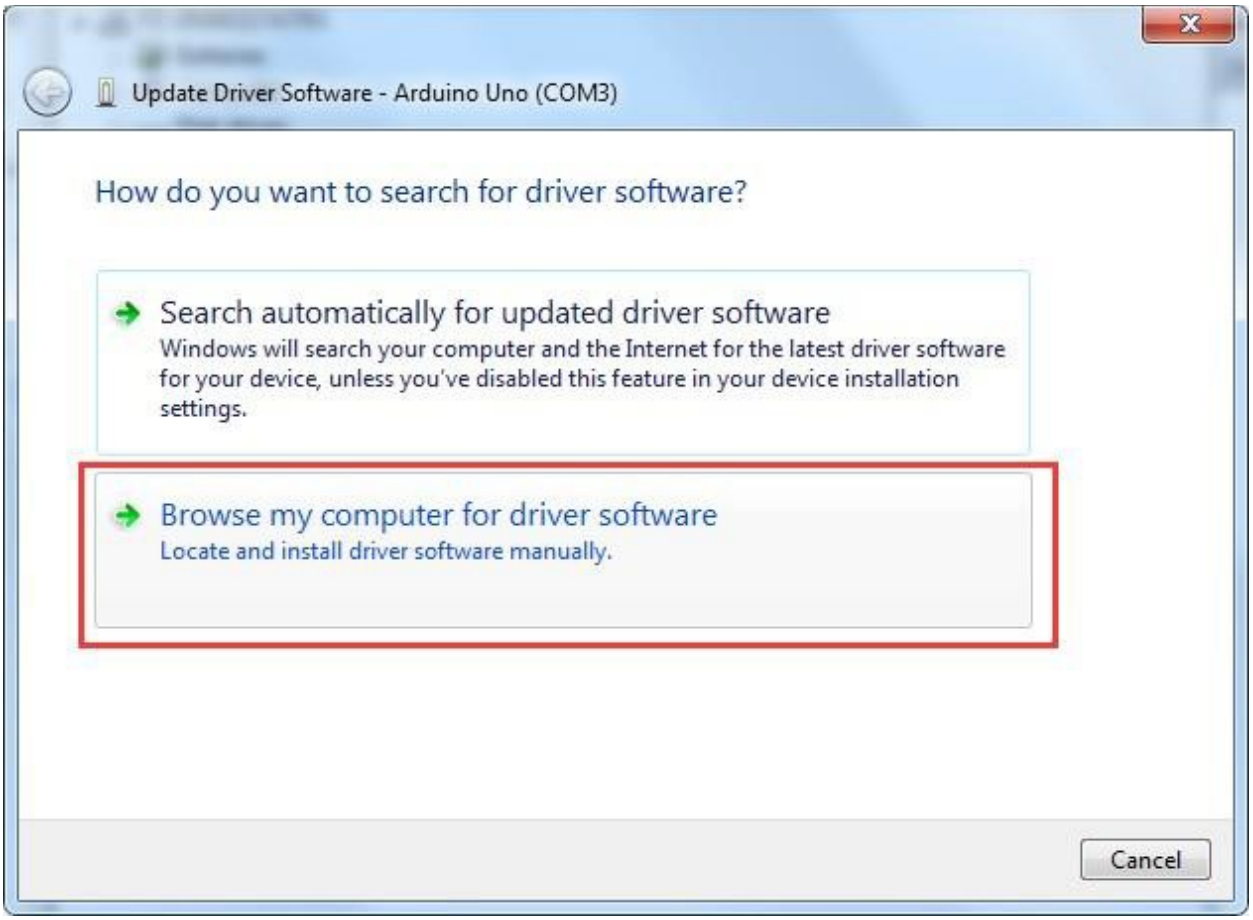
ただし、このインストール方法ではドライバを別途インストールする必要があります。

Arduino フォルダには、Arduino プログラム自体と、Arduino をコンピュータに USB ケーブルで接続するためのドライバが含まれています。 Arduino ソフトウェアを起動する前に、USB ドライバをインストールします。

USB ケーブルの一方の端を Arduino に差し込み、もう一方をコンピュータの USB ソケットに差し込みます。LED の電源ランプが点灯し、Windows から「新しいハードウェアの発見」メッセージが表示される場合があります。 このメッセージを無視し、Windows がドライバを自動的にインストールしようとする試行をすべてキャンセルします。USB ドライバをインストールする最も信頼性の高い方法は、デバイスマネージャを使用することです。 これは、Windows のバージョンによって異なる方法でアクセスされます。 Windows 7 では、最初にコントロールパネルを開き、アイコンを表示するオプションを選択する必要があります。リストでデバイスマネージャを見つけてください。

「その他のデバイス」の下に、「未知のデバイス」のアイコンが表示され、その隣に少し黄色の警告三角が表示されます。 これはあなたの Arduino です。





デバイスを右クリックし、トップメニューオプション（Update Driver Software ...）を選択します。「更新されたドライバソフトウェアを自動的に検索する」または「コンピュータでドライバソフトウェアを参照する」のいずれかのプロンプトが表示されます。 X \ arduino1.8.0 \ drivers をブラウズしてナビゲートするオプションを選択します。

[次へ]をクリックすると、セキュリティ警告が表示されます。インストールされている場合は、ソフトウェアのインストールが許可されます。 ソフトウェアがインストールされると、確認メッセージが表示されます。

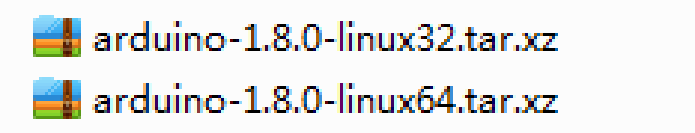
Windows ユーザーは、Mac および Linux システムのインストール手順をスキップして、レッスン 1 にジャンプすることがあります。Mac および Linux のユーザーは引き続きこのセクションを読むことができます。

Installing Arduino (Mac OS X)

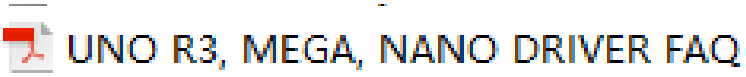
Zip ファイルをダウンロードして解凍し、Arduino.app をダブルクリックして Arduino IDE に入ります。 コンピュータに Java ランタイムライブラリがインストールされていない場合は、Java ランタイムライブラリをインストールするように求められます。 インストールが完了したら、Arduino IDE を実行できます。

Installing Arduino (Linux)

make install コマンドを使用する必要があります。 Ubuntu システムを使用している場合は、Ubuntu のソフトウェアセンターから Arduino IDE をインストールすることをお勧めします。



ヒント：ドライバのインストールに問題がある場合は、UNO R3、MEGA、NANO DRIVER の FAQ を参照してください。



Lesson 1　ライブラリを追加してシリアルモニタを開く

追加 Arduino ライブラリのインストール

Arduino ソフトウェアに慣れていて、組み込み関数を使用すると、Arduino の機能を追加のライブラリで拡張することができます。

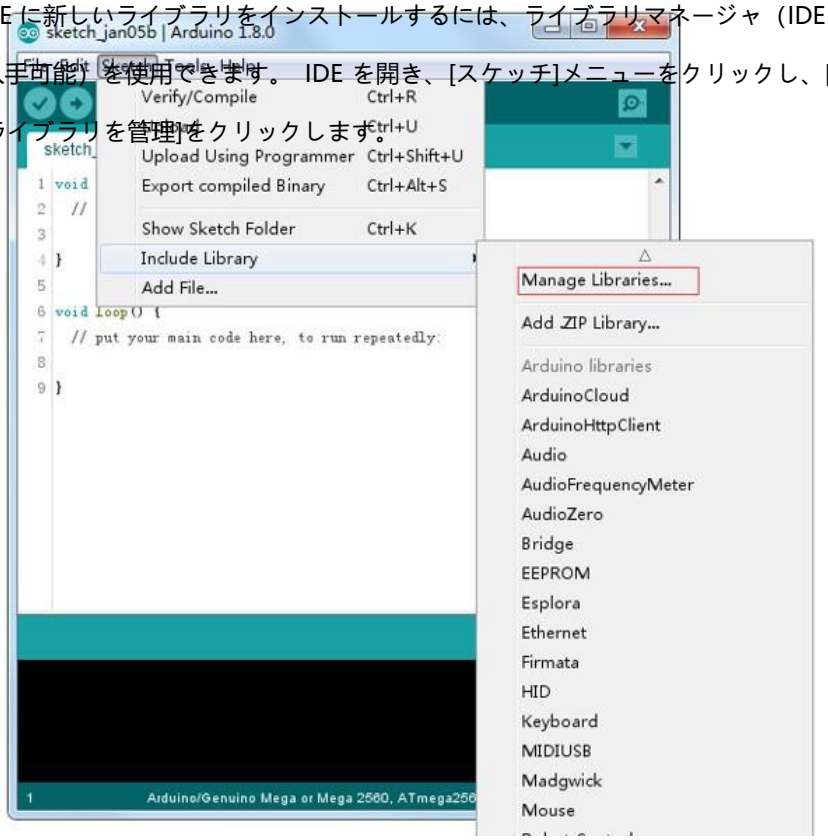
What are Libraries?

ライブラリは、センサ、ディスプレイ、モジュールなどに簡単に接続できるコードの集まりです。たとえば、内蔵の LiquidCrystal ライブラリを使用すると、キャラクタの LCD ディスプレイと簡単に会話することができます。ダウンロードのためにインターネット上で利用可能な数百の追加ライブラリがあります。 組み込みライブラリとこれらの追加ライブラリの一部は、参考文献にリストされています。 追加のライブラリを使用するには、それらをインストールする必要があります。

How to Install a Library

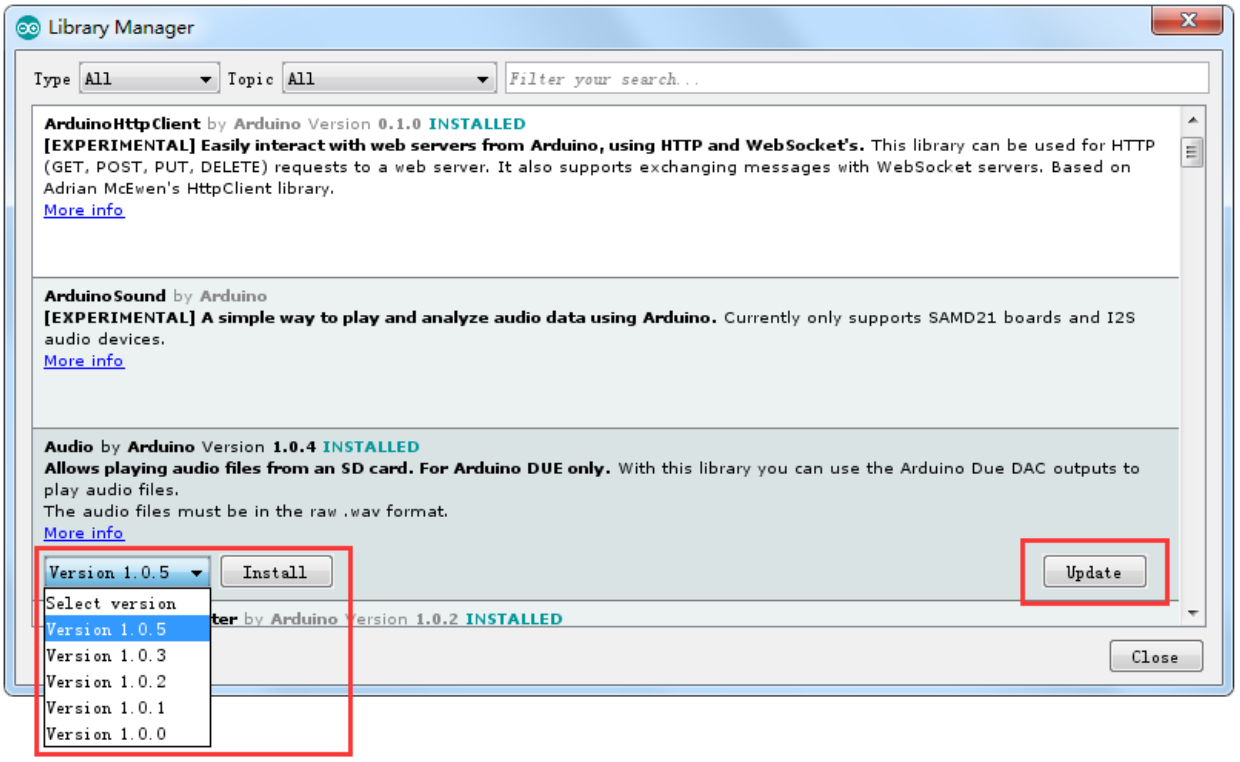
ライブラリマネージャの使用

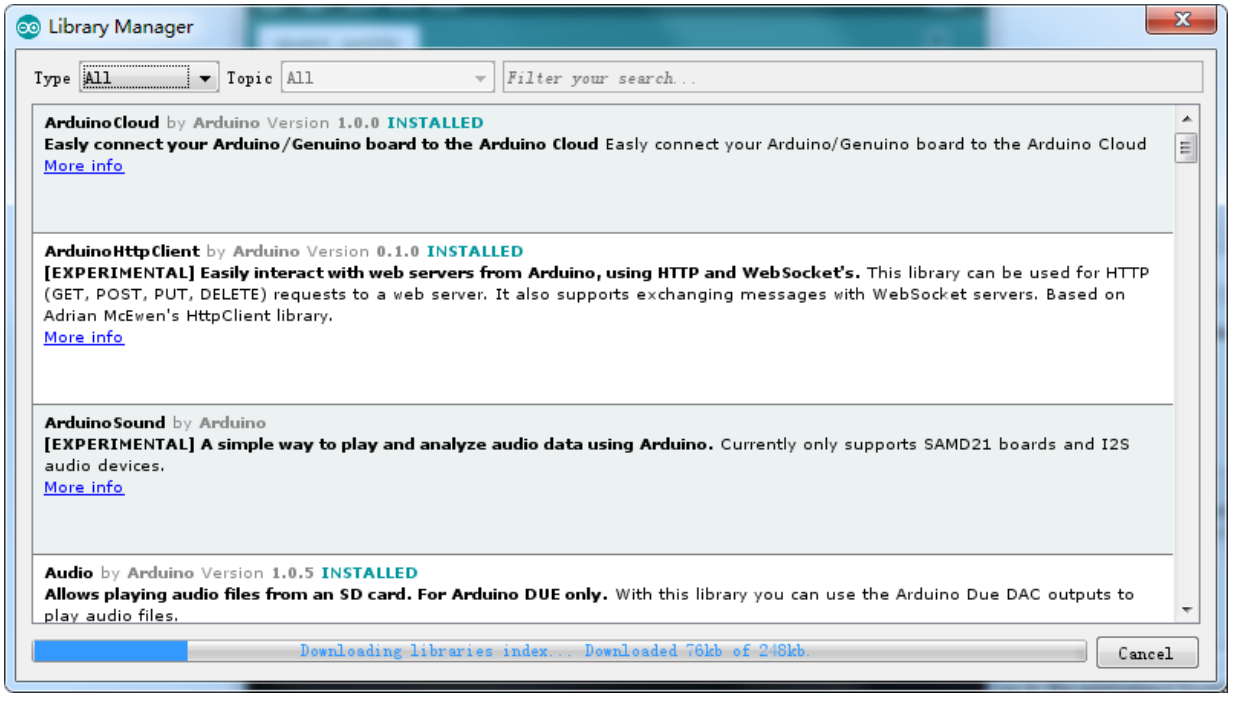
Arduino IDE に新しいライブラリをインストールするには、ライブラリマネージャ（IDE バージョン 1.8.0 から入手可能）を使用できます。 IDE を開き、[スケッチ]メニューをクリックし、[ライブラリを含む]> [ライブラリを管理]をクリックします。



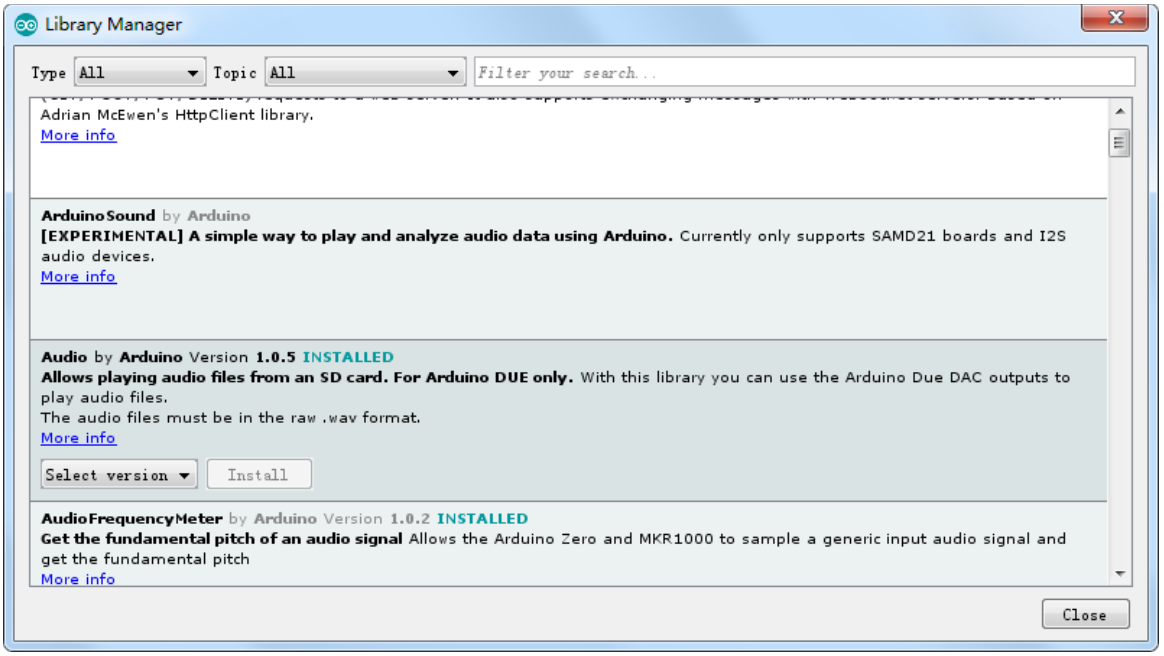
次に、ライブラリマネージャが開き、既にインストールされているか、インストールの準備が整っているライブラリのリストが表示されます。 この例では、Bridge ライブラリをインストールします。 リストをスクロールして検索し、インストールするライブラリのバージョンを選択します。 ライブラリの 1 つのバージョンしか利用できないことがあります。 バージョン選択メニューが表示されない場合は、これは正常です。

忍耐しなければならない時があり、図に示されるように。 それをリフレッシュしてそして待ってください。





最後に、install をクリックし、IDE が新しいライブラリをインストールするのを待ちます。 ダウンロードには接続速度に応じて時間がかかることがあります。 インストールが完了したら、Installed タグが Bridge ライブラリの横に表示されます。 ライブラリマネージャを閉じることができます。

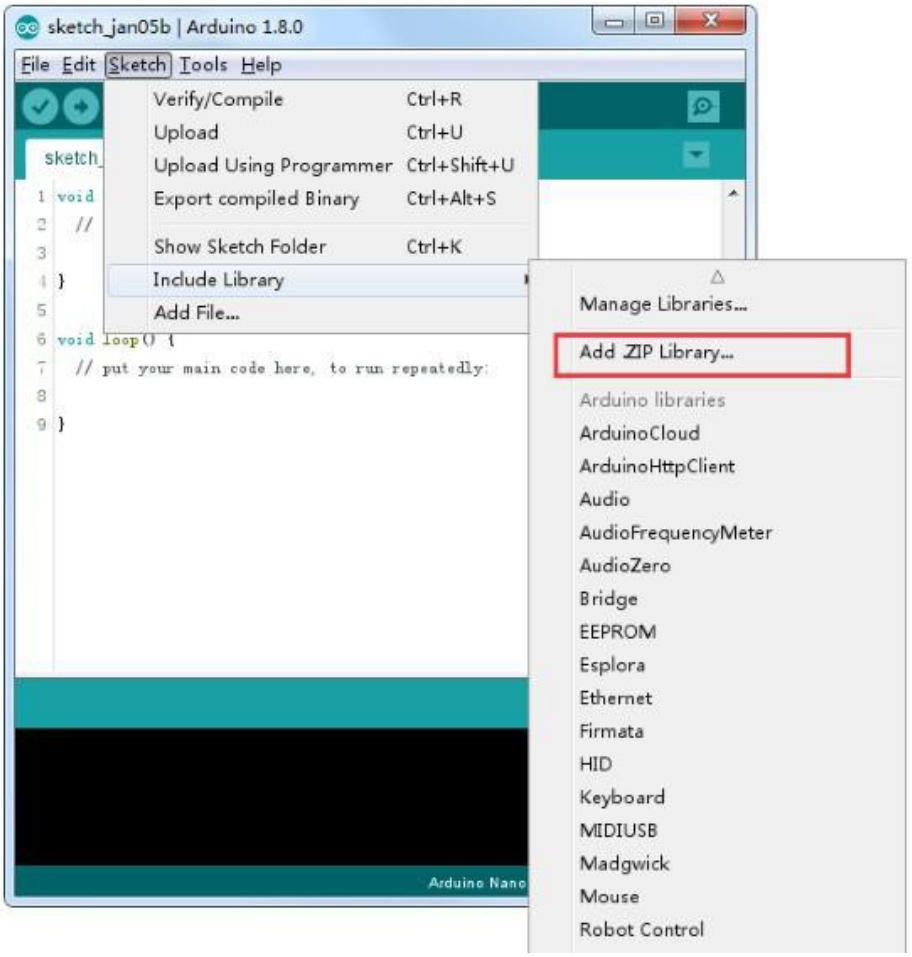


新しいライブラリは[ライブラリを含む]メニューで使用できるようになりました。 独自のライブラリを追加したい場合は、Github で新しい問題を開きます。

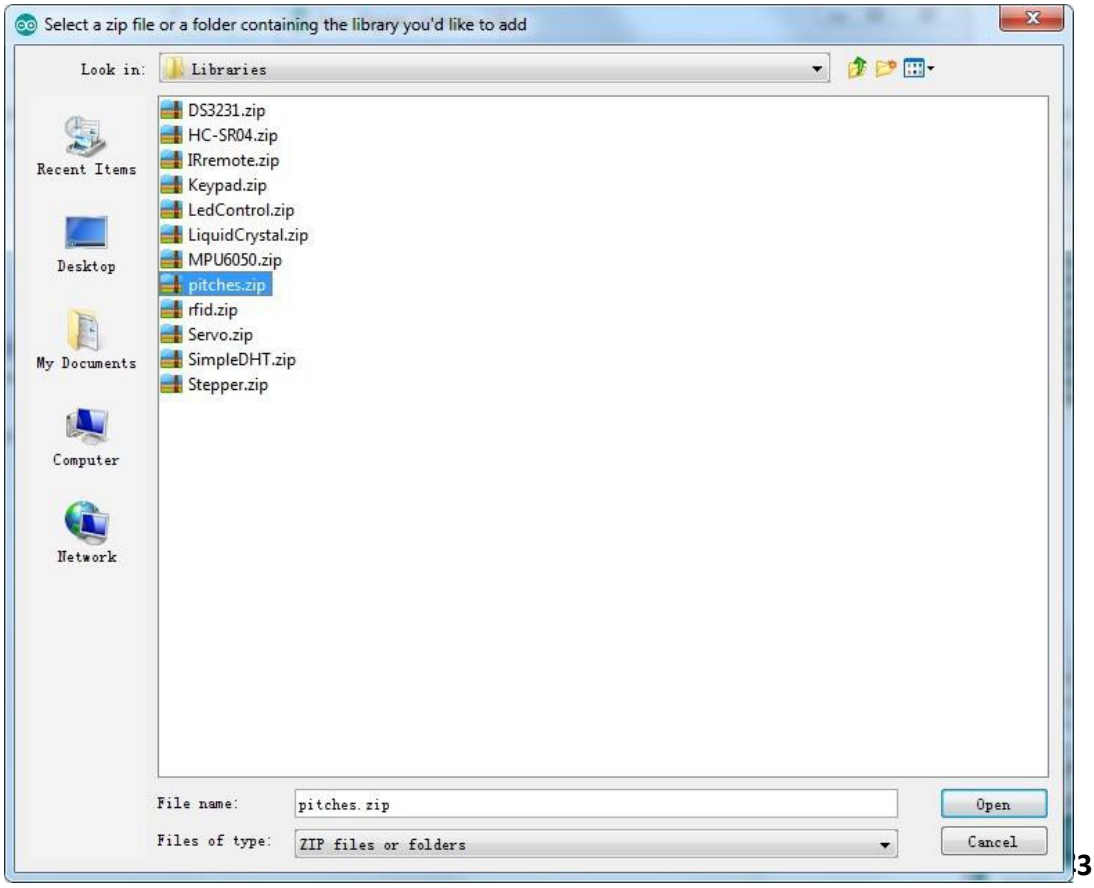
.zip ライブラリのインポート

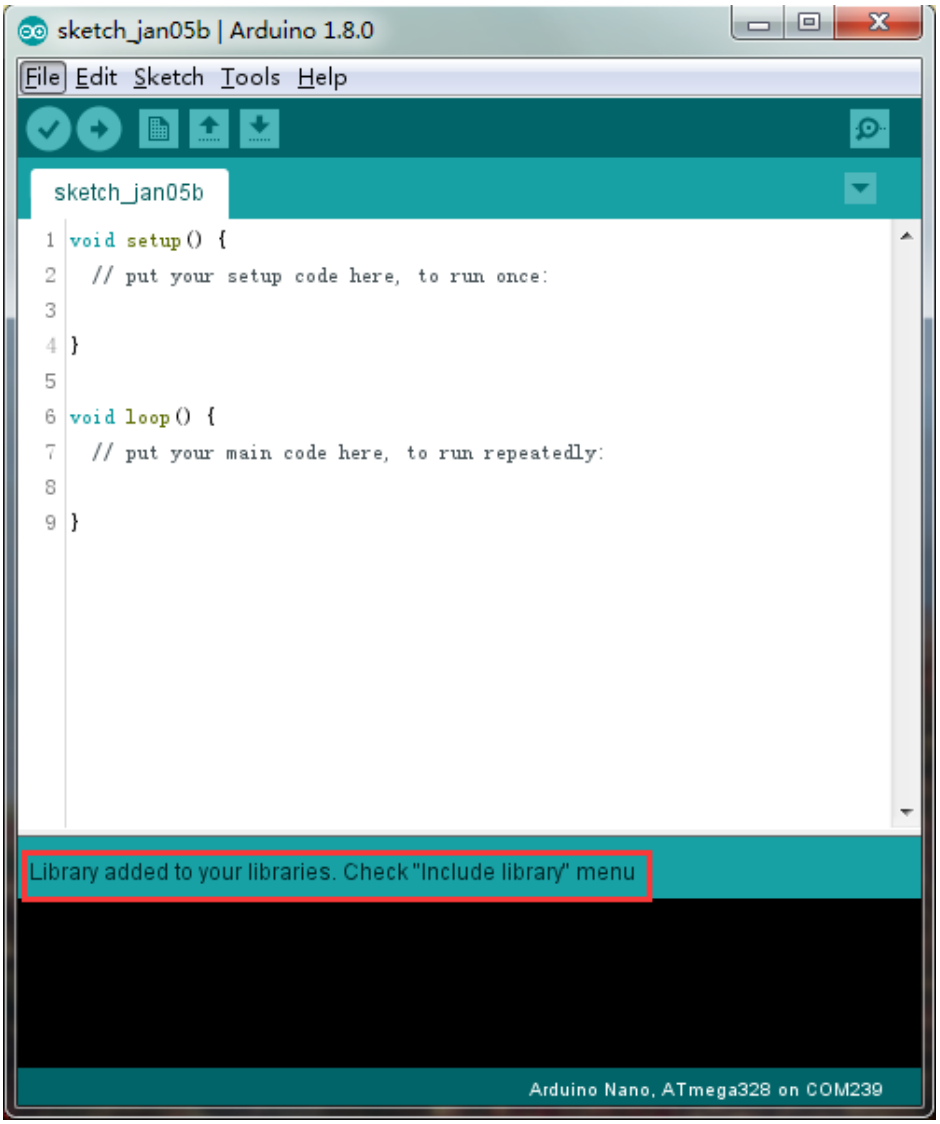
ライブラリは、多くの場合、ZIP ファイルまたはフォルダとして配布されます。 フォルダの名前はライブラリの名前です。 フォルダ内には、.cpp ファイル、.h ファイル、および keywords.txt ファイル、examples フォルダ、およびライブラリに必要なその他のファイルが含まれます。 バージョン 1.0.5 から、サードパーティのライブラリを IDE にインストールすることができます。 ダウンロードしたライブラリを解凍しないでそのままにしてください。

Arduino IDE で、[スケッチ]> [ライブラリを含める]に移動します。 ドロップダウンリストの一番上にある[ZIP ライブラリを追加]オプションを選択します。



追加したいライブラリを選択し、.zip ファイルの場所に移動して開きます。





スケッチ]> [ライブラリの読み込み]メニューに戻ります。 ドロップダウンメニューの一番下にライブラリが表示されます。 あなたのスケッチで使用する準備が整いました。 zip ファイルは Arduino スケッチディレクトリの libraries フォルダに展開されます。

注：ライブラリはスケッチで使用できるようになりますが、IDE の再起動後までライブラリの例は File> Examples で公開されません。

これらの 2 つが最も一般的なアプローチです。 MAC および Linux システムも同様に処理できます。下記で紹介するマニュアルインストールはほとんど使用されないかもしれませんし、必要のないユーザーはそれをスキップするかもしれません。

手動インストール

ライブラリをインストールするには、まず Arduino アプリケーションを終了します。 その後、ライブラリを含む ZIP ファイルを解凍します。 たとえば、「ArduinoParty」というライブラリをインス

トールする場合は、ArduinoParty.zip を解凍します。

ArduinoParty.cpp や ArduinoParty.h のようなファイルが入った ArduinoParty という名前のフォルダがあります。 （.cpp ファイルと.h ファイルがフォルダ内にない場合は、ファイルを作成する必要があります。この場合、 "ArduinoParty"というフォルダを作成し、ZIP 内にあるすべてのファイルに移動します ArduinoParty.cpp や ArduinoParty.h のようなファイルです。）

ArduinoParty フォルダをこのフォルダ（ライブラリフォルダ）にドラッグします。 Windows では、おそらく "My Documents \ Arduino \ libraries"です。

Mac ユーザーにとっては、おそらく "Documents / Arduino / libraries"です。 Linux では、スケッチブックの "libraries"フォルダになります。

あなたの Arduino ライブラリのフォルダは次のようになります（Windows）：

My Documents\Arduino\libraries\ArduinoParty\ArduinoParty.cpp
My Documents\Arduino\libraries\ArduinoParty\ArduinoParty.h My Documents\Arduino\libraries\ArduinoParty\examples

(Mac と Linux):

Documents/Arduino/libraries/ArduinoParty/ArduinoParty.cpp
Documents/Arduino/libraries/ArduinoParty/ArduinoParty.h
Documents/Arduino/libraries/ArduinoParty/examples

....

cpp ファイルと.h ファイルだけでなく、もっと多くのファイルがあるかもしれません。すべてがそこにあることを確認してください。

（.cpp ファイルと.h ファイルをライブラリフォルダに直接置くか、余分なフォルダに入れた場合、ライブラリは機能しません。例えば： Documents¥Arduino¥libraries¥ArduinoParty.cpp and Documents¥Arduino¥libraries¥ArduinoParty¥ArduinoParty¥ArduinoParty.cpp won't work.)

Arduino アプリケーションを再起動します。 新しいライブラリが Sketch-> [ライブラリのインポート]メニュー項目を選択します。 それでおしまい！ ライブラリをインストールしました！

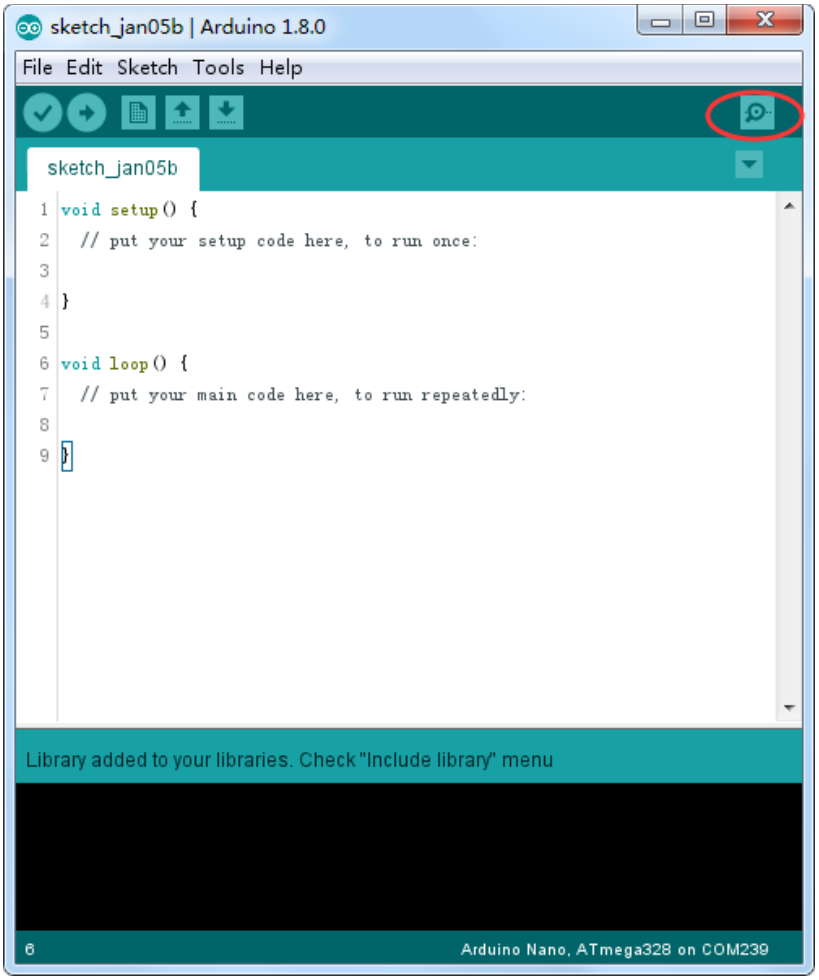
Arduino Serial Monitor (Windows, Mac, Linux)

Arduino 統合開発環境（IDE）は、Arduino プラットフォームのソフトウェア側です。 そして、端末を

使用することは、Arduinos と他のマイクロコントローラ、彼らはソフトウェアとシリアル端子を含めることにしました。 Arduino 環境では、これはシリアルモニタと呼ばれます。

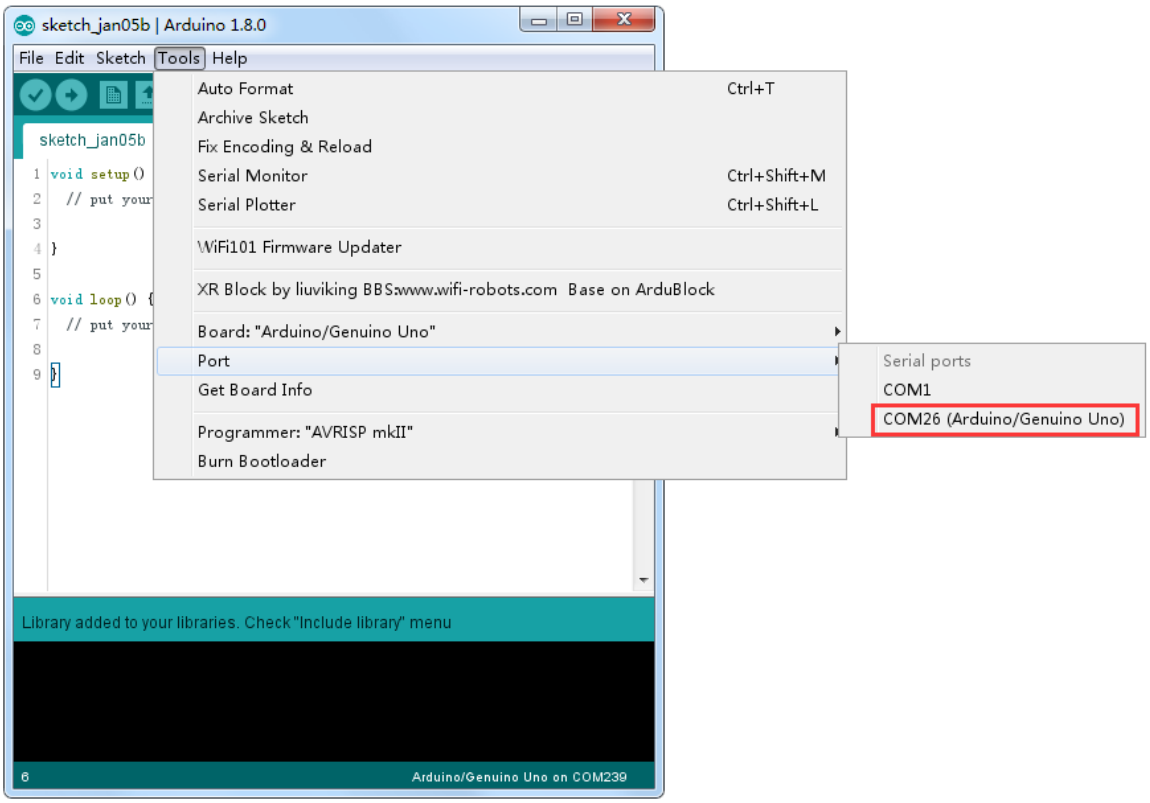
接続

シリアルモニタには、Arduino IDE のすべてのバージョンが付属しています。 それを開くには、シリアルモニタアイコンをクリックするだけです。

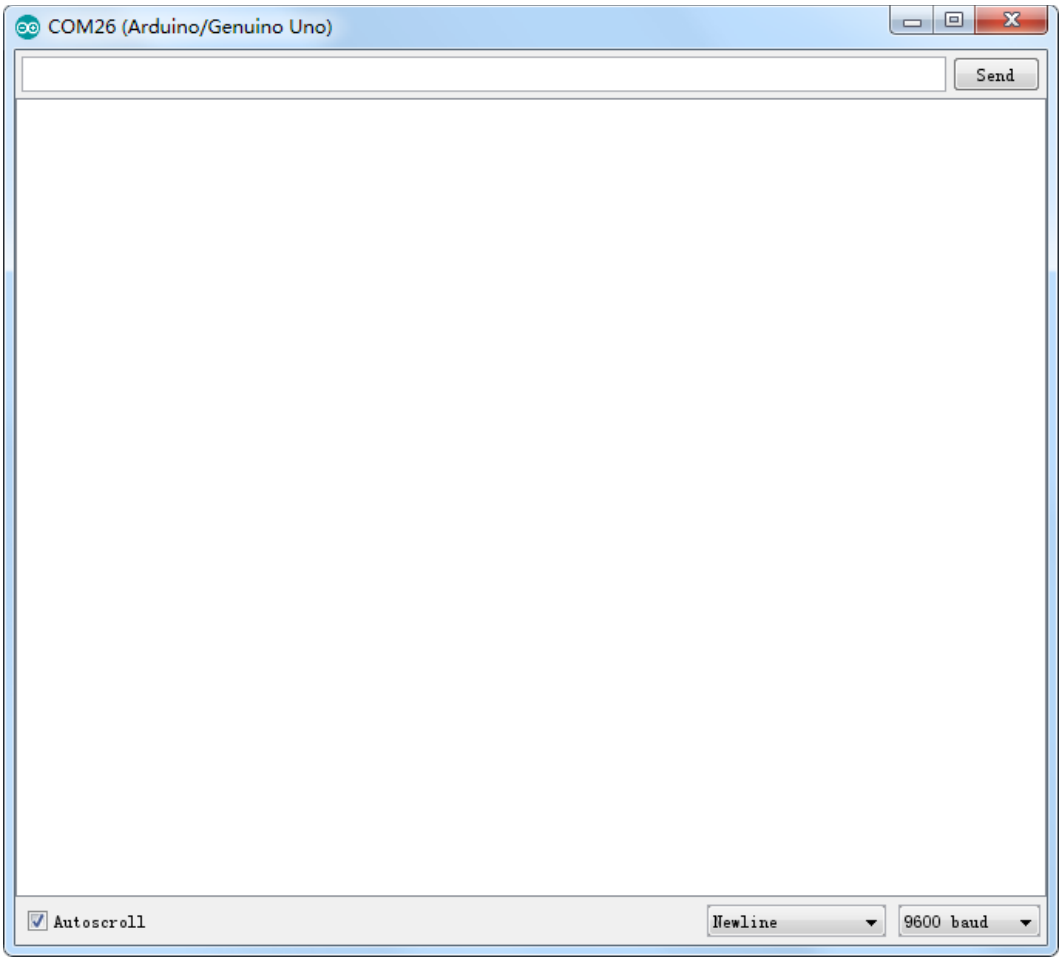


シリアルモニタで開くポートを選択することは、Arduino コードをアップロードするポートを選択することと同じです。 「ツール」 - >「シリアルポート」に移動し、正しいポートを選択します。

ヒント：デバイスマネージャーと同じ COM ポートを選択してください。

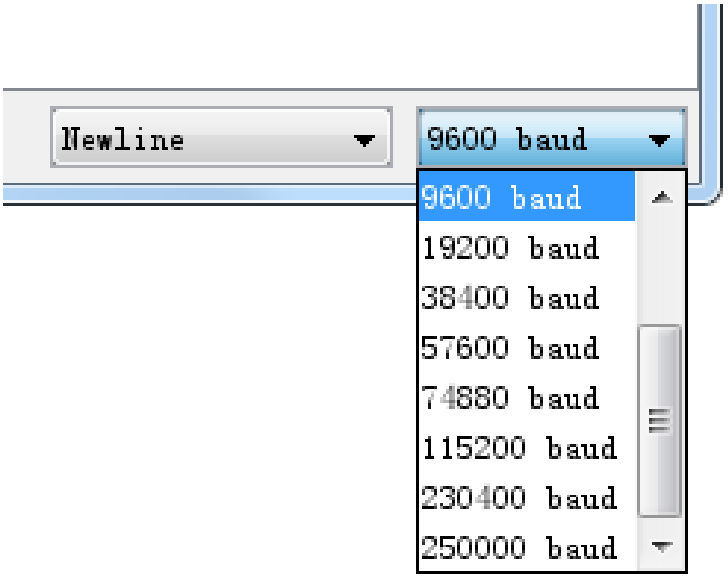


開いたら、次のように表示されます:

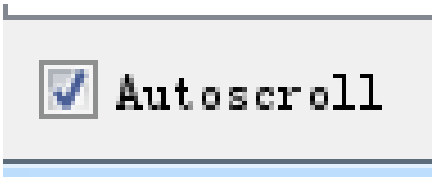


設定

アルモニタには設定が限られていますが、ほとんどのシリアル通信のニーズに対応できます。 変更可能な最初の設定はボーレートです。 ボーレートドロップダウンメニューをクリックして正しいボーレートを選択します。 （9600 ボー）



最後に、左下隅のチェックボックスをオンにすることで、端末を自動スクロールするかどうかを設定できます。



長所

シリアルモニタは Arduino とのシリアル接続を確立するのに素早く簡単な方法です。 既に Arduino IDE で作業している場合は、別の端末を開いてデータを表示する必要はありません。

短所

シリアルモニタでは設定の欠如が望ましく、高度なシリアル通信の場合は、このトリックはできません。

Lesson 2 点滅

概要

このレッスンでは、Arduino の内蔵 LED を点滅させられる UNO R3 コントロール方法と、プログラムをダウンロードする手順を学習します。。

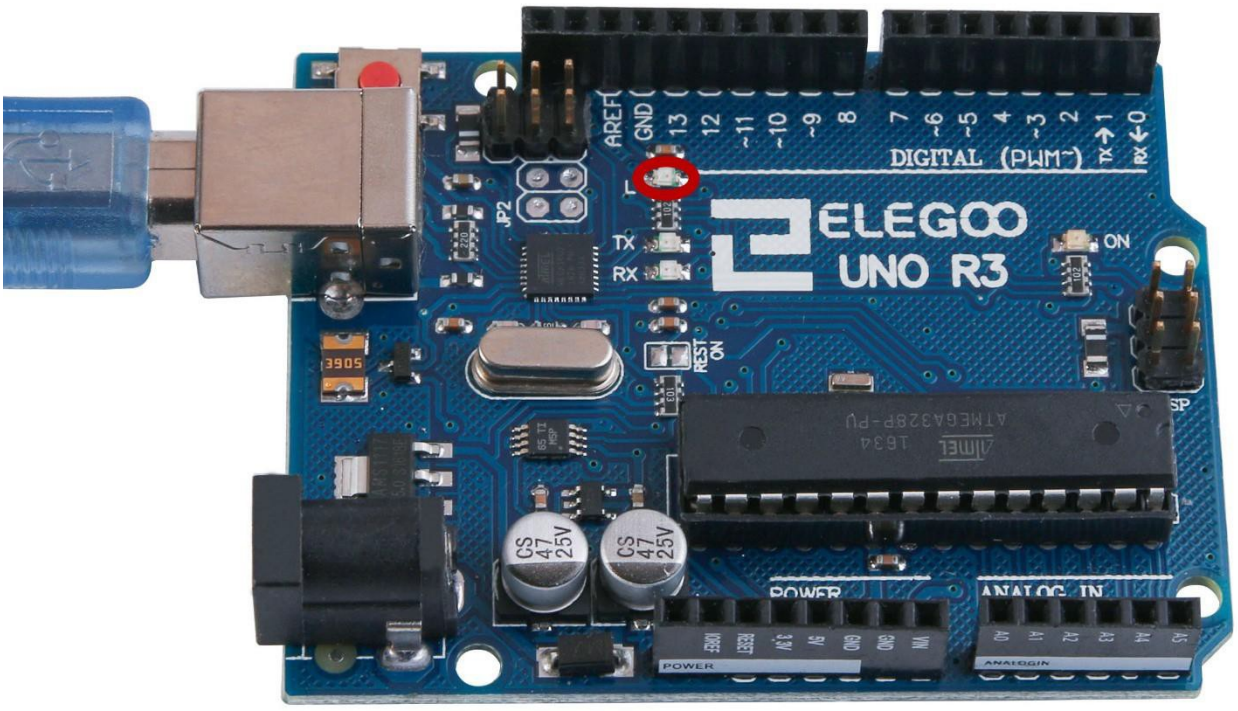
必要な構成部品:

(1)　x Elegoo Uno R3

原理

UNO R3 ボードには、複数の電子デバイスとその機能を拡張するプラグインのシールドに接続するために使用される、両側に沿ったコネクタ列があります。

また、スケッチから制御できる単一の LED もあります。 この LED は UNO R3 ボードに内蔵されており、ボード上にラベル付けされているので、しばしば「L」LED と呼ばれます。



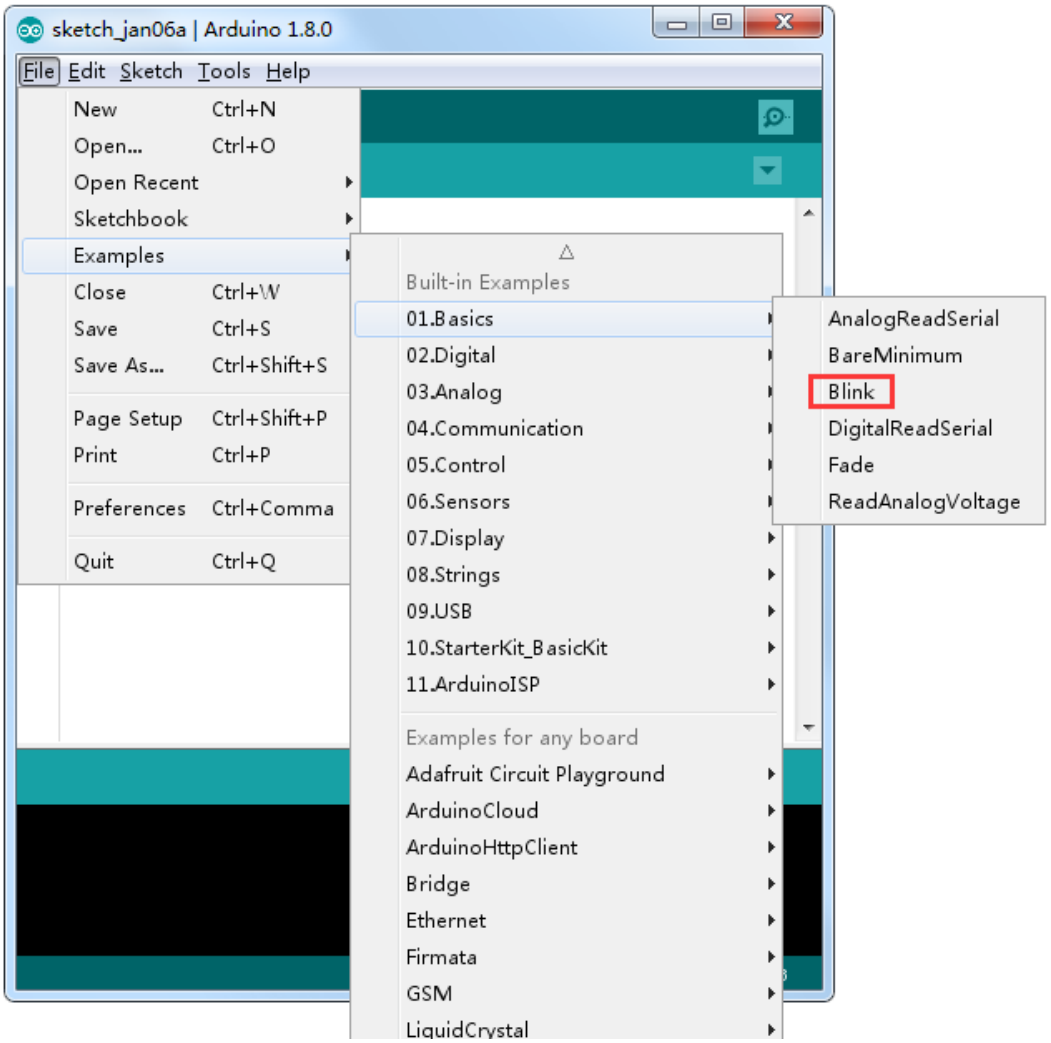
UNO R3 ボードの「L」LED は、USB プラグに接続すると既に点滅しています。 これは、ボードには一般に「Blink」スケッチがプリインストールされて出荷されるためです。

このレッスンでは、UNO R3 ボードを独自の Blink スケッチで再プログラムし、点滅するレートを変更します。

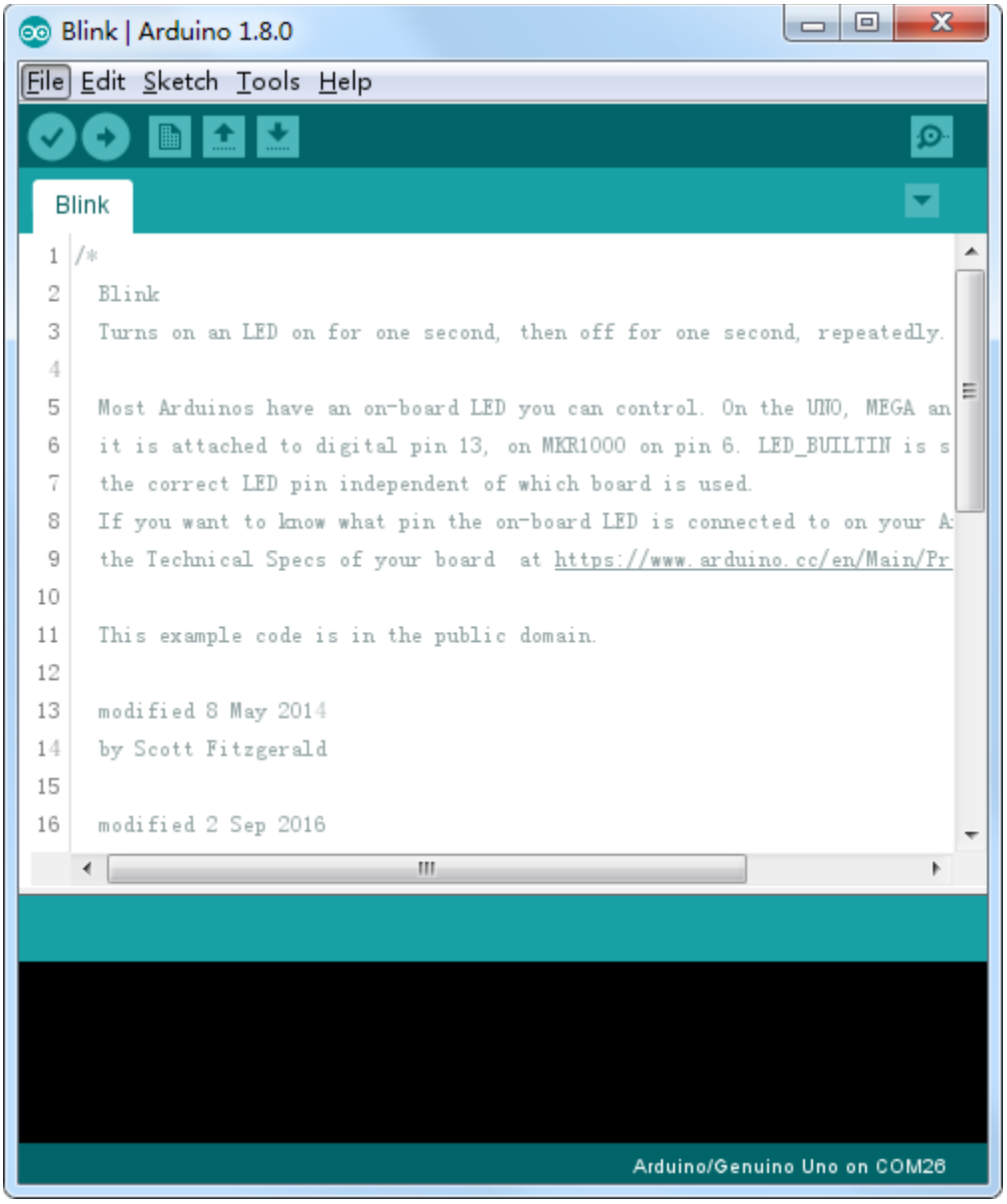
レッスン 0 では、Arduino IDE をセットアップし、UNO R3 ボードに接続するための正しいシリアルポートを見つけることができることを確認しました。 今、あなたの UNO R3 ボードをテストしてプログラミングできます。

Arduino IDE には、ロードして使用できるサンプルスケッチの大きなコレクションが含まれています。これには、 'L' LED を点滅させるためのスケッチ例が含まれています。

IDE のメニューシステムで[ファイル]> [サンプル]> [01.Basics]の[Blink]スケッチをロードします。

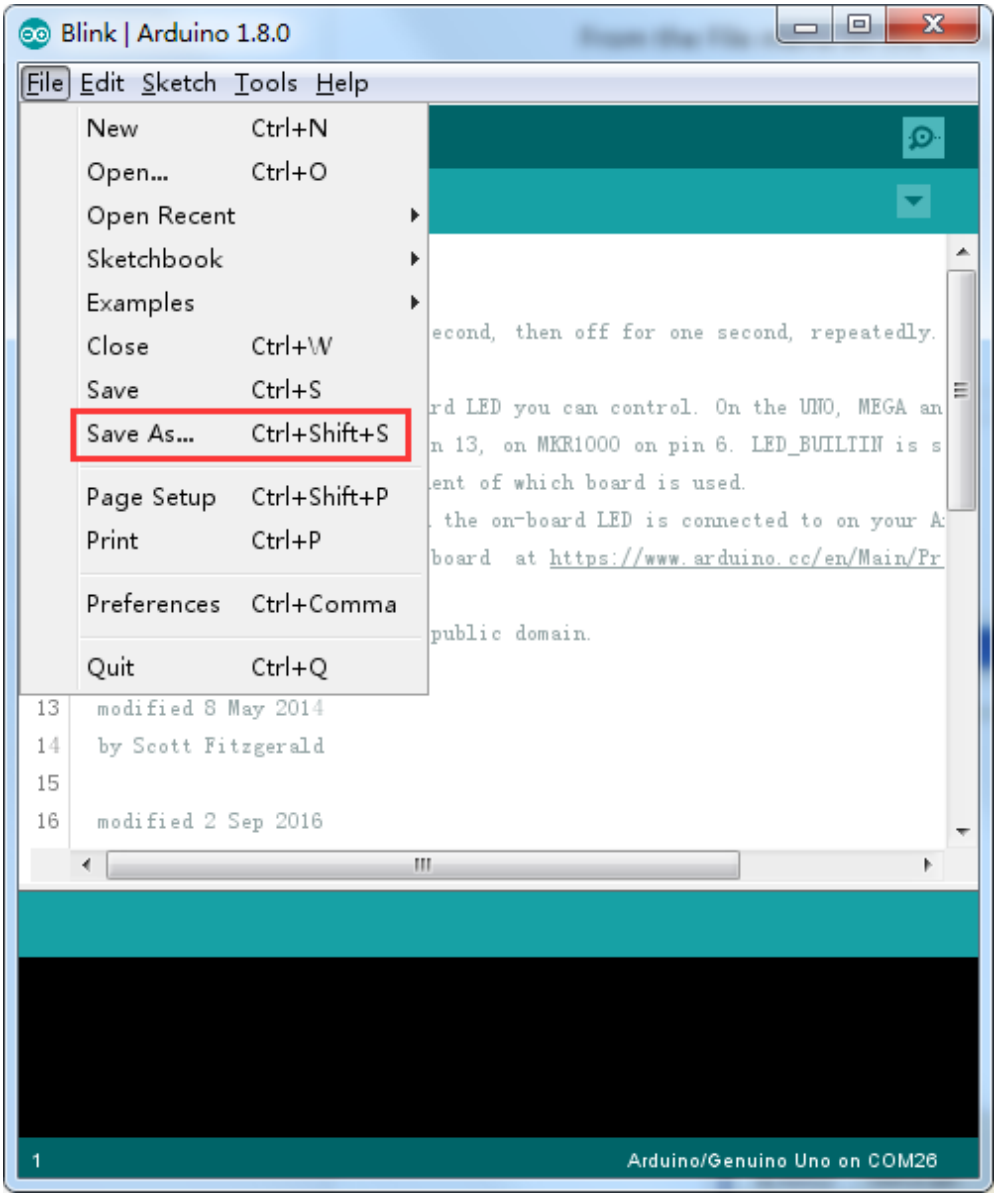


スケッチウィンドウが開いたら、拡大してスケッチ全体をウィンドウで見ることができます。

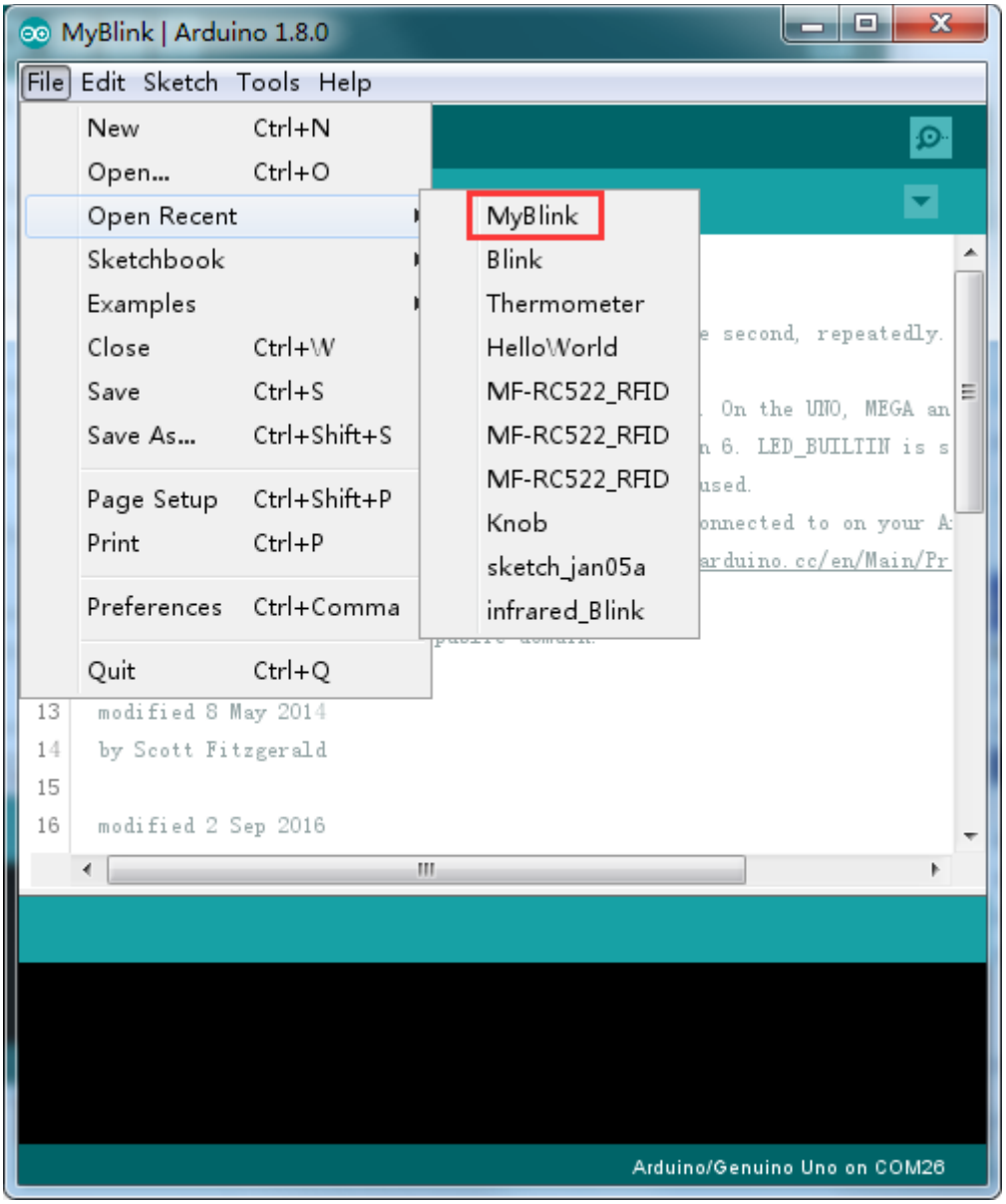


Arduino IDE に含まれているスケッチの例は「読み取り専用」です。 つまり、それらを UNO R3 ボードにアップロードできますが、変更した場合は同じファイルとして保存することはできません。

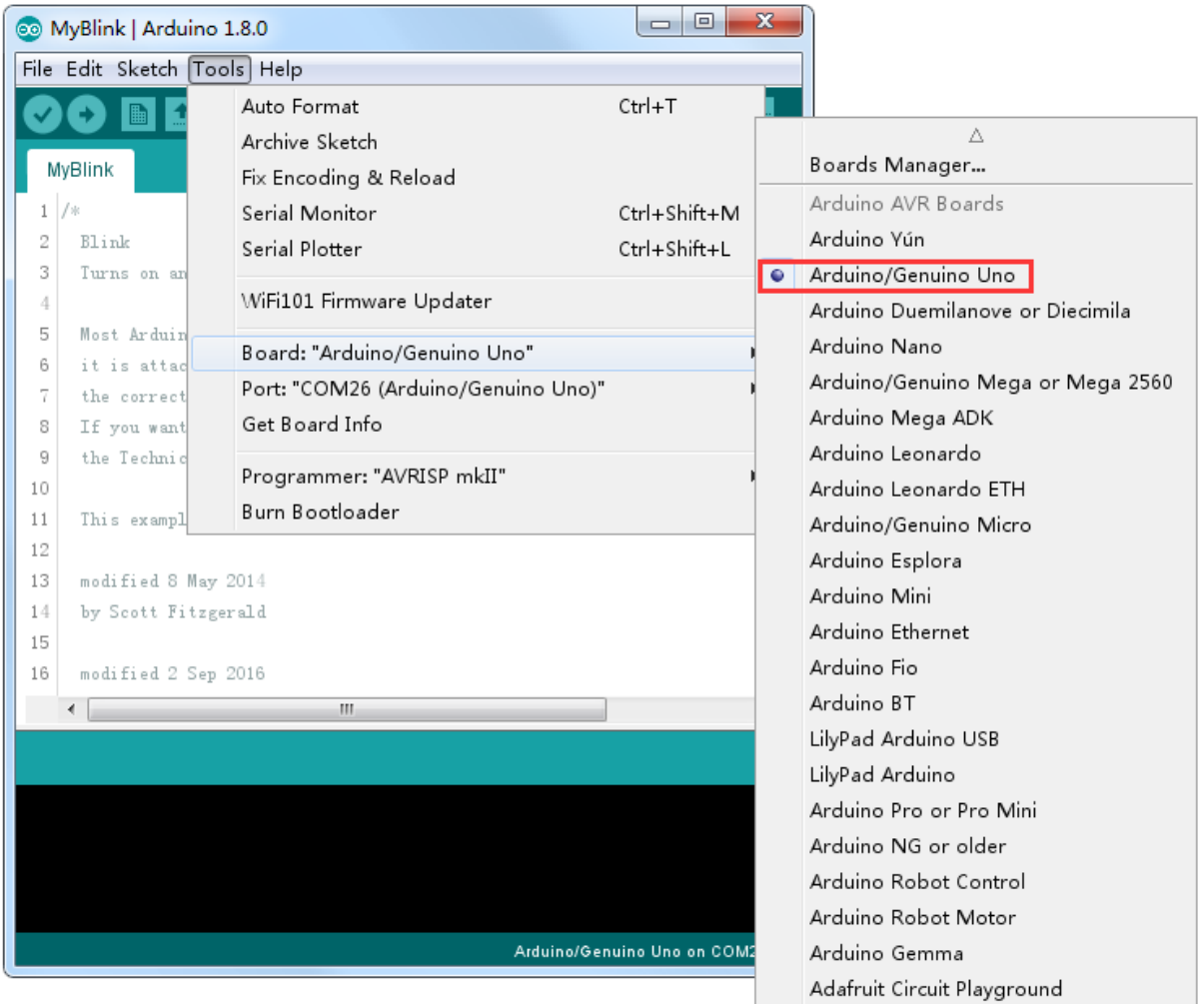
このスケッチを変更する予定なので、最初に行う必要があるのは、自分のコピーを保存することです。Arduino IDE のファイルメニューから「名前を付けて保存」を選択し、スケッチを「MyBlink」という名前で保存します。

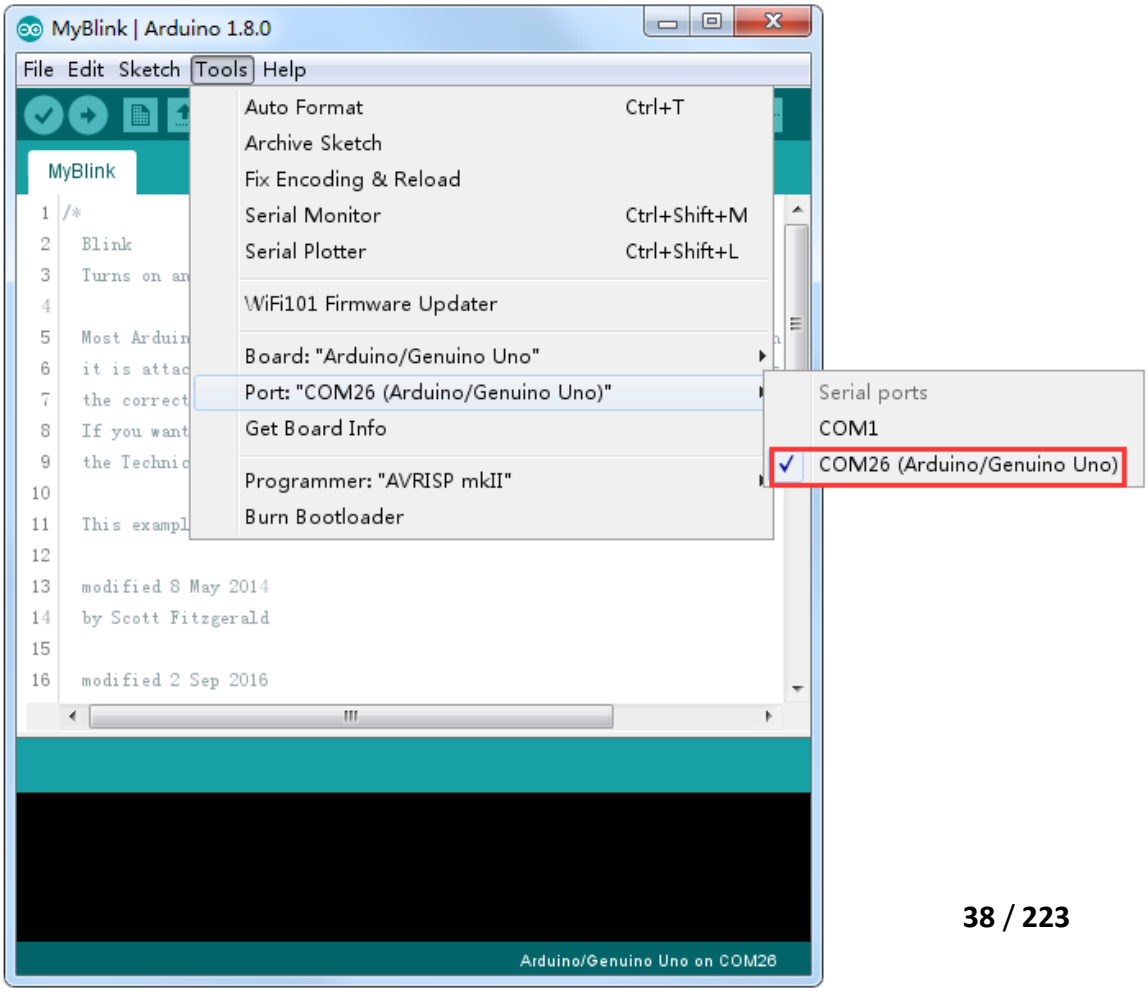

Blink | Arduino 1.8.0
File Edit Sketch Tools Help
New Ctrl+N
Open... Ctrl+O
Open Recent
Sketchbook
Examples
Close Ctrl+W
Save Ctrl+S
Save As... Ctrl+Shift+S
Page Setup Ctrl+Shift+P
Print Ctrl+P
Preferences Ctrl+Comma
Quit Ctrl+Q
modified 8 May 2014
by Scott Fitzgerald
modified 2 Sep 2016
Arduino/Genuino Uno on COM26

スケッチブックにあなたのコピーを保存しました。 つまり、これを再度検索したい場合は、「ファイル」>「スケッチブック」メニューオプションを使用して開くことができます。



コンピュータに Arduino ボードを USB ケーブルで接続し、 'Board Type'と 'Serial Port'が正しく設定されていることを確認します。


MyBlink | Arduino 1.8.0
File Edit Sketch Tools Help
MyBlink
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 Firmware Updater
Board: "Arduino/Genuino Uno"
Port: "COM26 (Arduino/Genuino Uno)"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Boards Manager...
Arduino AVR Boards
Arduino Yún
Arduino/Genuino Uno
Arduino Duemilanove or Diecimila
Arduino Nano
Arduino/Genuino Mega or Mega 2560
Arduino Mega ADK
Arduino Leonardo
Arduino Leonardo ETH
Arduino/Genuino Micro
Arduino Esplora
Arduino Mini
Arduino Ethernet
Arduino Fio
Arduino BT
LilyPad Arduino USB
LilyPad Arduino
Arduino Pro or Pro Mini
Arduino NG or older
Arduino Robot Control
Arduino Robot Motor
Arduino Gemma
Adafruit Circuit Playground
Arduino/Genuino Uno on COM


MyBlink | Arduino 1.8.0
File Edit Sketch Tools Help
MyBlink
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 Firmware Updater
Board: "Arduino/Genuino Uno"
Port: "COM26 (Arduino/Genuino Uno)"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Serial ports
COM1
COM26 (Arduino/Genuino Uno)
Arduino/Genuino Uno on COM26

注: ボードタイプとシリアルポートは、ここに示したものと必ずしも同じではありません。 2560 を使用している場合は、Board Type として Mega 2560 を選択する必要があります。他の選択肢も同じ方法で行うことができます。 そして、すべての人に表示されるシリアルポートは、ここで選択された COM 26 にもかかわらず、あなたのコンピュータ上の COM3 または COM4 である可能性があります。 正しい COM ポートは、認証基準である COMX（arduino XXX）であるはずです。

Arduino IDE は、ウィンドウの下部にあるボードの現在の設定を表示します。

[アップロード]ボタンをクリックします。 ツールバーの左から 2 番目のボタン。

IDE のステータス領域を見ると、進行状況バーと一連のメッセージが表示されます。 最初は、「スケッチの編集...」と表示されます。 これは、スケッチをボードにアップロードするのに適したフォーマットに変換します。

次に、ステータスが「アップロード中」に変わります。 この時点で、Arduino の LED は、スケッチが転送されるときにちらつき始めるはずです。

最後にステータスが「完了」に変わります。

他のメッセージは、スケッチが利用可能な 32,256 バイトの 928 バイトを使用していることを示しています。「スケッチをコンパイルしています」段階の後、次のエラーメッセージが表示されます:

ボードがまったく接続されていないか、ドライバがインストールされていないか（必要な場合）、または間違ったシリアルポートが選択されている可能性があります。

これに遭遇した場合は、レッスン 0 に戻ってインストールを確認してください。

アップロードが完了したら、ボードを再起動して点滅を開始する必要があります。 コードを開く

このスケッチの大部分はコメントで構成されていることに注意してください。これらは実際のプログラム命令ではありません。 むしろ、プログラムの仕組みを説明するだけです。 彼らはあなたの利益のためにそこにいます。

スケッチの最上部にある/ *と* /の間のすべてがブロックコメントです。 それはスケッチのためのものを説明します。

一行のコメントは、//で始まり、その行の終わりがコメントとみなされるまですべて終わります。

コードの最初の行は次のとおりです:

```
int led = 13;
```

上のコメントが説明するように、これは LED が付いているピンの名前を与えています。 UNO や Leonardo を含む、ほとんどの Arduinos では 13 です。

次に、「設定」機能があります。 再度、コメントが示すように、リセットボタンが押されたときに実行されます。 電源投入時やスケッチがアップロードされた後など、何らかの理由でボードがリセットされたときにも実行されます。

```
void setup() {
// initialize the digital pin as an output.
pinMode(led, OUTPUT);
}
```

すべての Arduino スケッチには 'setup'機能が必要です。自分で指示を追加したい場所は{と}の間にあります。

この場合、そこにコマンドが 1 つしかありません。コメント状態が Arduino ボードに LED ピンを出力として使用するように指示しています。

スケッチが 'ループ'機能を持つことも必須です。 一度しか動作しない「セットアップ」機能とは異なり、リセット後、「ループ」機能はコマンドの実行が終了した後すぐに再び開始されます。

```
void loop() {
   digitalWrite(led, HIGH);      // turn the LED on (HIGH is the voltage level)
   delay(1000);                        // wait for a second
   digitalWrite(led, LOW);       // turn the LED off by making the voltage LOW
   delay(1000);                        // wait for a second
}
```

ループ機能の中で、コマンドはまず LED ピンを（HIGH）にしてから、1000 ミリ秒（1 秒）だけ遅延させてから、LED ピンをオフにしてもう 1 秒間ポーズします。

あなたは今あなたの LED を速く点滅させるつもりです。 あなたが推測したように、これの鍵は、'delay'コマンドのための（）のパラメータを変更することにあります。

```
30 // the loop function runs over and over again forever
31 void loop() {
32   digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the volt
33   delay(500)                         // wait for a second
34   digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the vo
35   delay(500)                         // wait for a second
36 }
```

この遅延時間はミリ秒単位であるため、LED を 2 倍速く点滅させたい場合は、値を 1000 から 500 に変更します。その後、1 秒間ではなく、0.5 秒ごとに一時停止します。

スケッチをもう一度アップロードすると、LED がすぐに点滅し始めます。

Lesson 3 LED

概要

このレッスンでは、異なる値の抵抗を使用して LED の輝度を変更する方法を学習します。

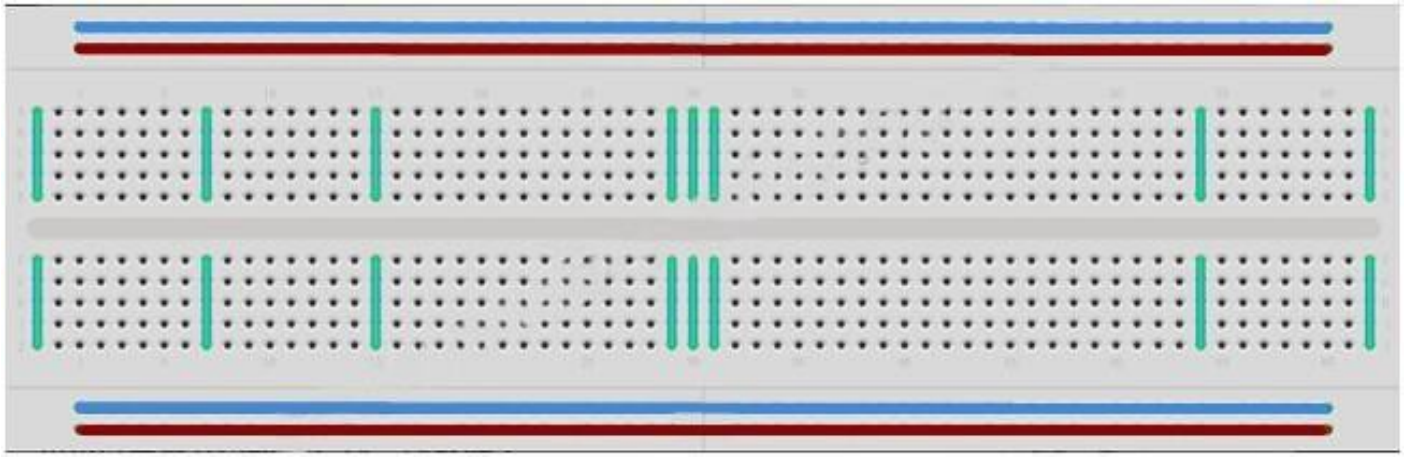
必要な構成部品:

(1) x Elegoo Uno R3
(1) x 5mm red LED
(1) x 220 ohm resistor
(1) x 1k ohm resistor
(1) x 10k ohm resistor
(2) x M-M wires (Male to Male jumper wires)

部品の紹介

BREADBOARD MB-102：

ブレッドボードを使用すると、接続をハンダ付けすることなく、素早く回路を試作できます。 以下は例です。



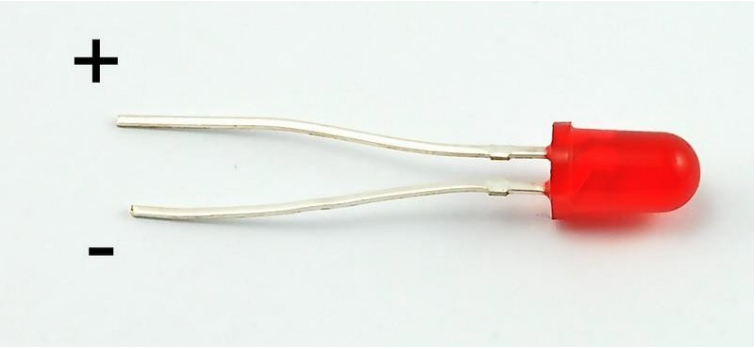
パンフレットには様々なサイズと構成があります。最も単純な種類は、プラスチックブロックの穴のグリッドです。内部には短い列の穴の間を電気的に接続するための金属ストリップがあります。 2 つの異なるコンポーネントの脚部を同じ列に押し込むと、それらを電気的に結合します。中央の深いチャンネルは、そこに接続が途切れていることを示します。つまり、チャンネルの両側にある脚をチップを一緒に接続せずに差し込むことができます。一部のブレッドボードには、ボードの長辺に沿ってメイングリッドから離れた 2 つのストリップの穴があります。これらのストリップは内部のボードを通り、共通の電圧を接続する手段を提供します。これらは通常+5 ボルトとグランドのペアになっています。これらのストリップはレールと呼ばれ、ボード内の多くのコンポーネントやポイントに電源を接続することができます。

ブレッドボードはプロトタイピングには最適ですが、いくつかの制限があります。接続はプッシュフィットで一時的なので、はんだ付けされた接続ほど信頼性がありません。断続的な回路に問題がある場合は、ブレッドボードの接続不良の可能性があります。

LED:

LED が優れたインジケータライトを作ります。 彼らは電気をほとんど使わず、長寿命です。

このレッスンでは、最も一般的なすべての LED（5mm 赤色 LED）を使用します。 5mm は LED の直径を意味します。 他の一般的なサイズは 3mm と 10mm です。 LED をバッテリまたは電圧源に直接接続することはできません。その理由は、1) LED は正と負のリード線を持ち、誤った方向に置かれた場合は点灯しません。2) LED は抵抗を使って電流を制限するか「チョーク」する必要があります。さもなければ、それは燃え尽きるでしょう!



LED を付属の抵抗器を使用しないと、あまりにも多くの電流が流れ、加熱され、光が発生する「接合部」を破壊するため、ほとんど直ちに破壊される可能性があります。

どちらが LED の正のリードであり、負のリードであるかを知るには 2 つの方法があります。

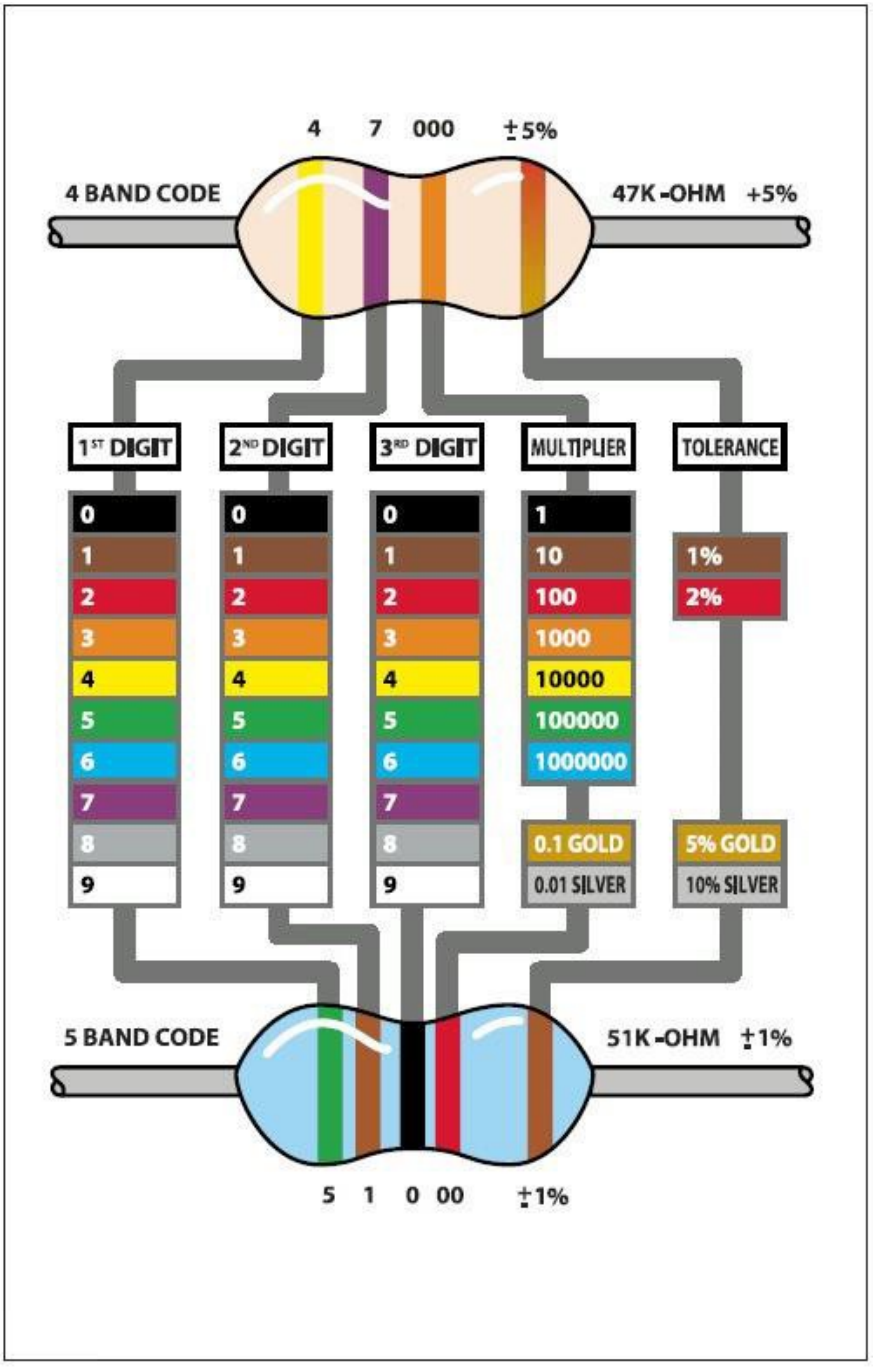
まず、陽性リードは長くなります。

第 2 に、負のリードが LED の本体に入る場合、LED のケースに対して平坦な縁がある。
長いリードの横に平坦な側面を持つ LED がある場合は、長いリードが陽性であると認識する必要があります。

抵抗器:

名前が示すように、抵抗器は電気の流れに抵抗します。 抵抗の値が高いほど、抵抗が大きくなり、流れる電流は少なくなります。 LED をどれだけの電流が流れているかを制御するために、これを使用して、それがどれだけ明るく輝いているかを制御します。

しかし、まず、さらに抵抗について...
抵抗の単位はオームと呼ばれ、通常はオーム語のギリシャ文字 Ω に短縮されます。 オームは抵抗値が低いため（小さい抵抗です）、kΩ（1,000Ω）と MΩ（1,000,000Ω）も抵抗値を示します。 これらはキロオームとメガオームと呼ばれます。
このレッスンでは、220Ω、1kΩ、10kΩの 3 種類の抵抗器を使用します。 これらの抵抗器はすべて、同じ色をしていますが、異なる色の縞模様があります。 これらのストライプは抵抗の値を示します。
レジスタのカラーコードには 3 つの色のストライプがあり、一方の端に金色のストライプがあります。

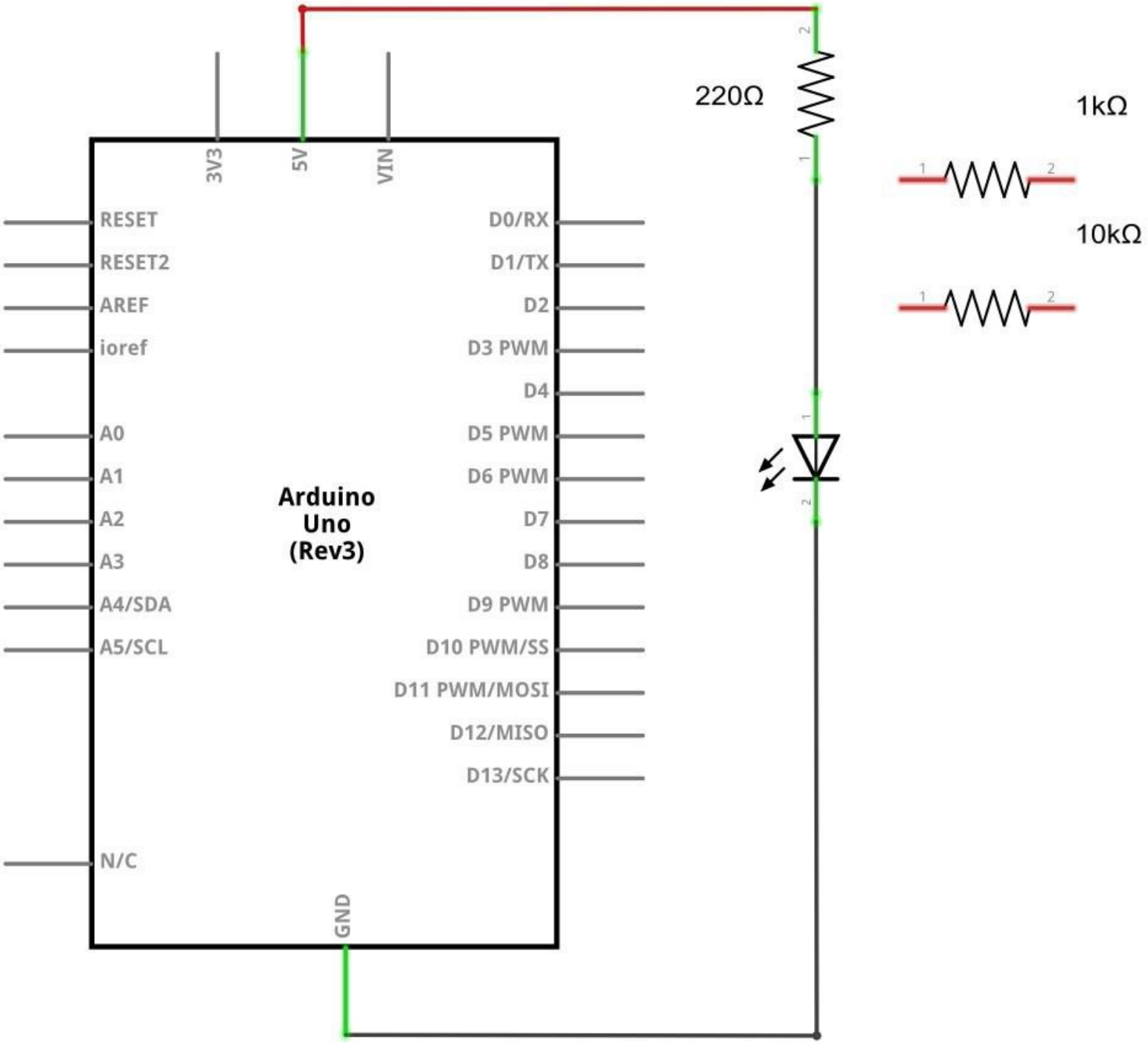


LED とは異なり、抵抗には正と負のリードがありません。 それらはいずれかの方法で接続することができます。

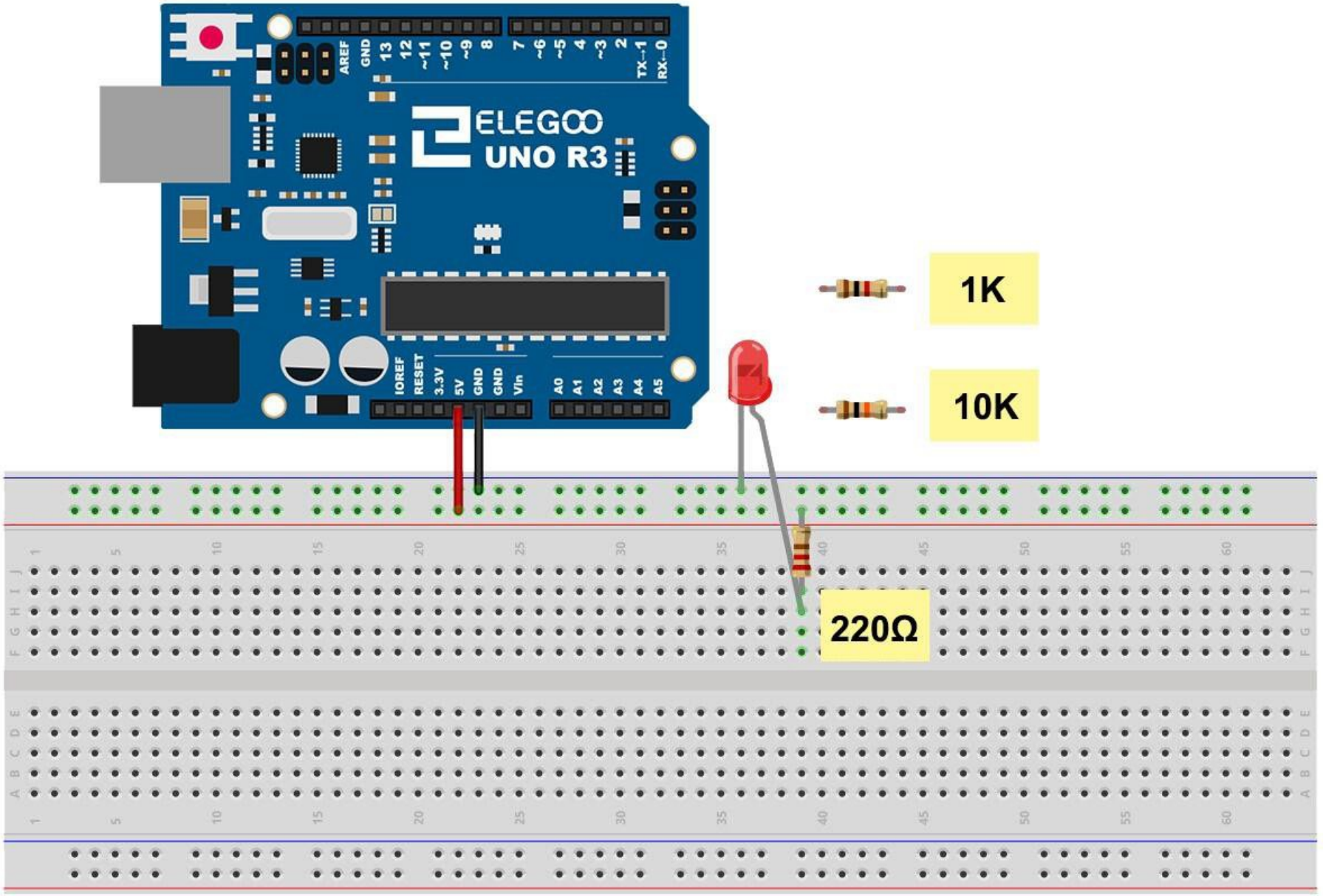
このアプローチ方法が複雑すぎるとわかったら、私たちの抵抗のカラーリングフラグを直接読んでその抵抗値を決定することができます。 または、代わりにデジタルマルチメータを使用することもできます。

Connection

Schematic



Wiring diagram



UNO は、LED と抵抗に電力を供給するために使用する 5 ボルトの便利な電源です。 UNO を USB ケーブルに差し込む以外は何もする必要はありません。

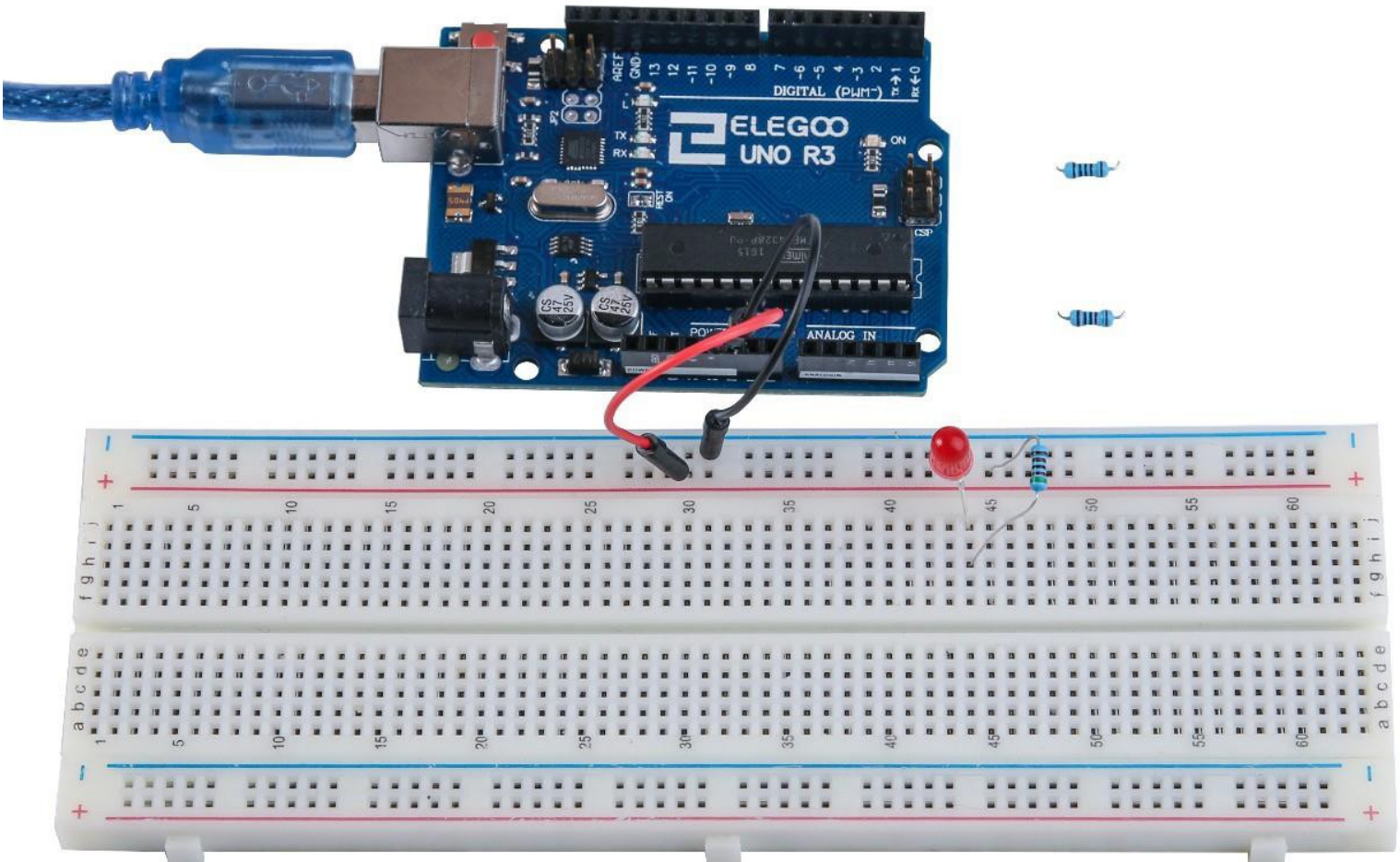
220Ωの抵抗を取り付けた場合、LED は非常に明るいはずです。 1kΩ抵抗の 220Ω抵抗を交換すると、LED が少し減光します。 最後に、10kΩの抵抗を配置すると、LED が目に見えます。 赤いジャンパーリードをブレッドボードから引き出し、穴の中に触れて取り除き、スイッチのように動作させます。 あなたは違いを気付くことができます。

現時点では、抵抗器の一方の脚に 5V が流れ、抵抗器のもう一方の脚は LED のプラス側に、LED のもう一方の側は GND に向かって流れます。 ただし、LED を点灯させるように抵抗を移動すると、以下のように LED が点灯します。

おそらく 220Ωの抵抗を元の場所に戻したいと思うでしょう。

どこかにある限り、LED のどちら側に抵抗を置いても問題ありません。

Example picture



Lesson 4 RGB LED

概要

RGB LED は、プロジェクトに色を追加する楽しく簡単な方法です。 それらは 1 つは 3 つの LED で、それらを使用して接続する方法はそれほど違いはありません。

それらは主に 2 つのバージョンで提供されます：Common Anode または Common Cathode。

共通アノードは共通ピンに 5V を使用し、共通カソードはグランドに接続します。

他の LED と同様に、いくつかの抵抗をインライン（3 つの合計）に接続する必要があります。

私たちのスケッチでは、赤色の状態の LED で始まり、次に緑色にフェードアウトし、次に青色にフェードアウトし、最後に赤色に戻ります。これを行うことで、達成可能な色のほとんどを循環させます。

必要な構成部品:

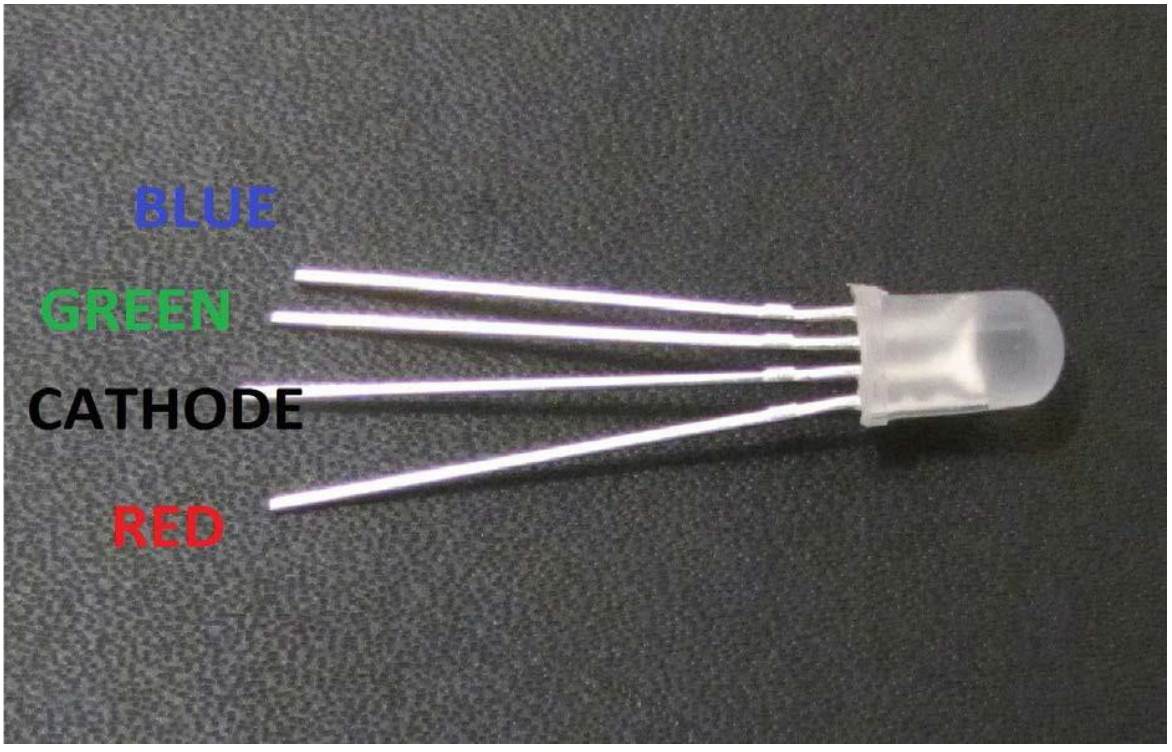
(1) x Elegoo Uno R3
(1) x 830 Tie Points Breadboard
(4) x M-M wires (Male to Male jumper wires)
(1) x RGB LED
(3) x 220 ohm resistors

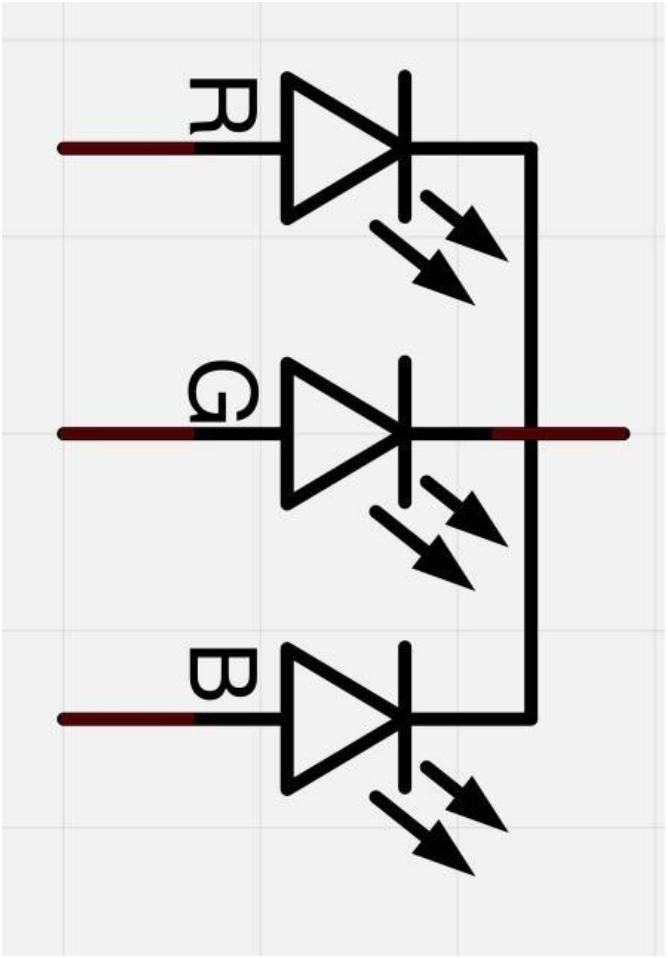
部品の紹介

RGB:

一見すると、RGB（赤、緑、青）の LED は通常の LED のように見えます。 しかし、通常の LED パッケージの内部には、実際には赤色、緑色、はい、青色の 3 つの LED があります。 個々の LED の明るさを制御することで、あなたが望むどんな色でもかなり混在させることができます。

3 つの LED のそれぞれの明るさを調整することで、パレットにペイントを混ぜるのと同じ方法で色をミックスします。 これを行う難しい方法は、レッスン 2 で行ったように異なる値の抵抗（または可変抵抗）を使用することですが、それは多くの作業です！ UNO R3 ボードには analogWrite 機能があり、∽でマークされたピンで使用して、適切な LED にさまざまな電力を出力できます。
RGB LED には 4 本のリード線があります。 パッケージ内の単一の LED のそれぞれの正の接続に向かう 1 つのリードと、LED の 3 つの負の側面のすべてに接続された単一のリードがあります。





ここで写真には 4 つの電極 LED が見えます。 緑色、青色、赤色の各ピンはアノードと呼ばれます。 あなたは常にそれに "+"を接続します。 陰極は " - "（地面）に行きます。 それ以外の方法で接続すると、LED は点灯しません。

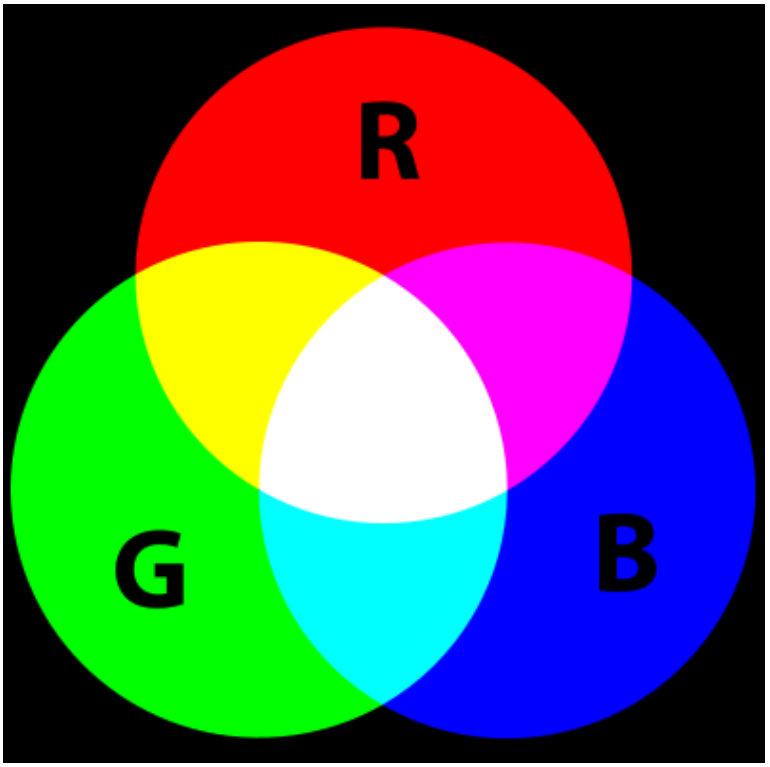
LED パッケージの共通の負の接続は、平らな側から 2 番目のピンです。 それはまた、4 つのリード線のうち最長であり、接地されます。

パッケージ内の各 LED は、あまりにも多くの電流が流れないように、独自の 220Ω抵抗を必要とします。 LED の 3 つの正のリード線（赤色、緑色、青色）は、これらの抵抗を使用して UNO 出力ピンに接続されています。

COLOR:

赤、緑、青の光の量を変えることで好きな色を混ぜることができるのは、目には 3 種類の光レセプター（赤、緑、青）があるからです。 あなたの目と脳は、赤、緑、青の量を処理し、それをスペクトルの色に変換します。

ある意味では、3 つの LED を使用することで、私たちは目は錯覚します。 この同じ考え方は、LCD に赤、緑、青の色のドットが並び、各ピクセルを構成しているテレビで使用されています。

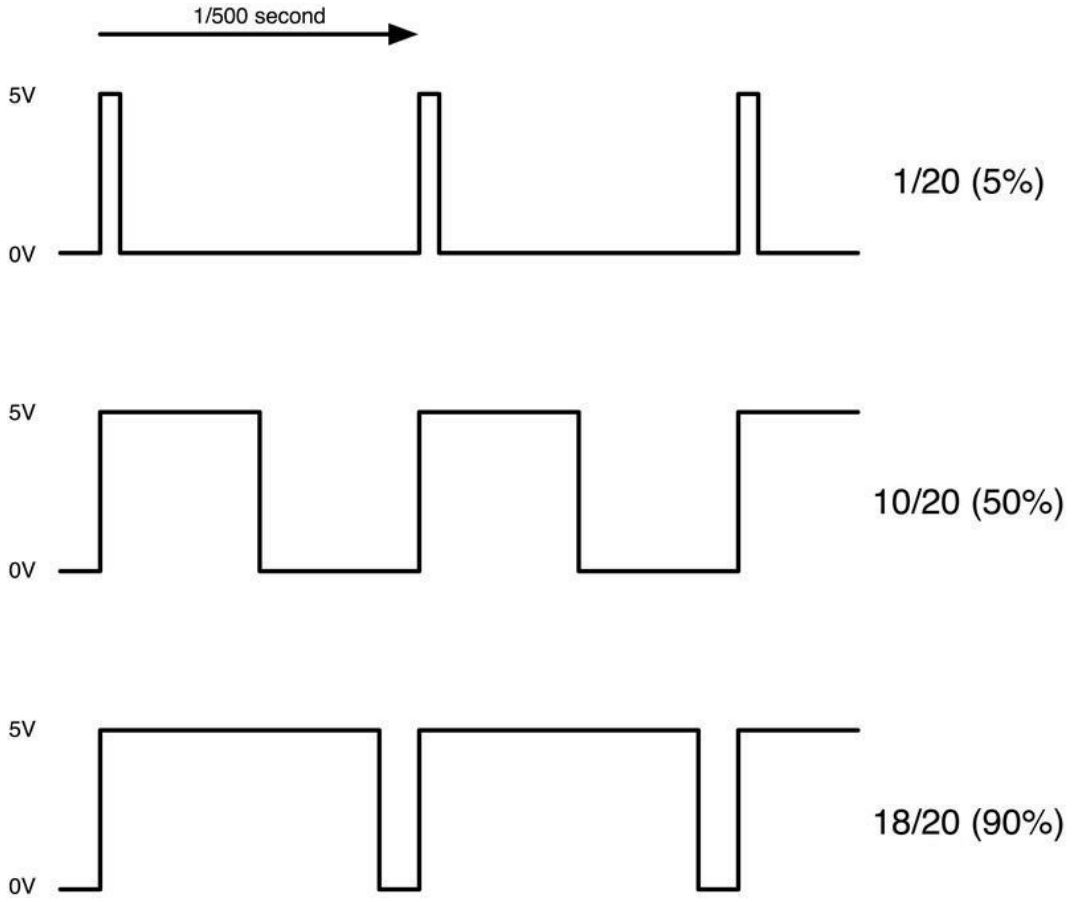


3 つすべての LED の輝度を同じに設定すると、光の全体的な色は白色になります。 赤と緑の LED だけが同じ明るさになるように青の LED を消すと、ライトは黄色に見えます。

LED の赤、緑、青の各部分の明るさを個別に制御できるので、好きな色を混在させることができます。黒はそれほど光のない色ではありません。 したがって、我々の LED で最も近いのは 3 色すべてを消すことです。

理論 (PWM)

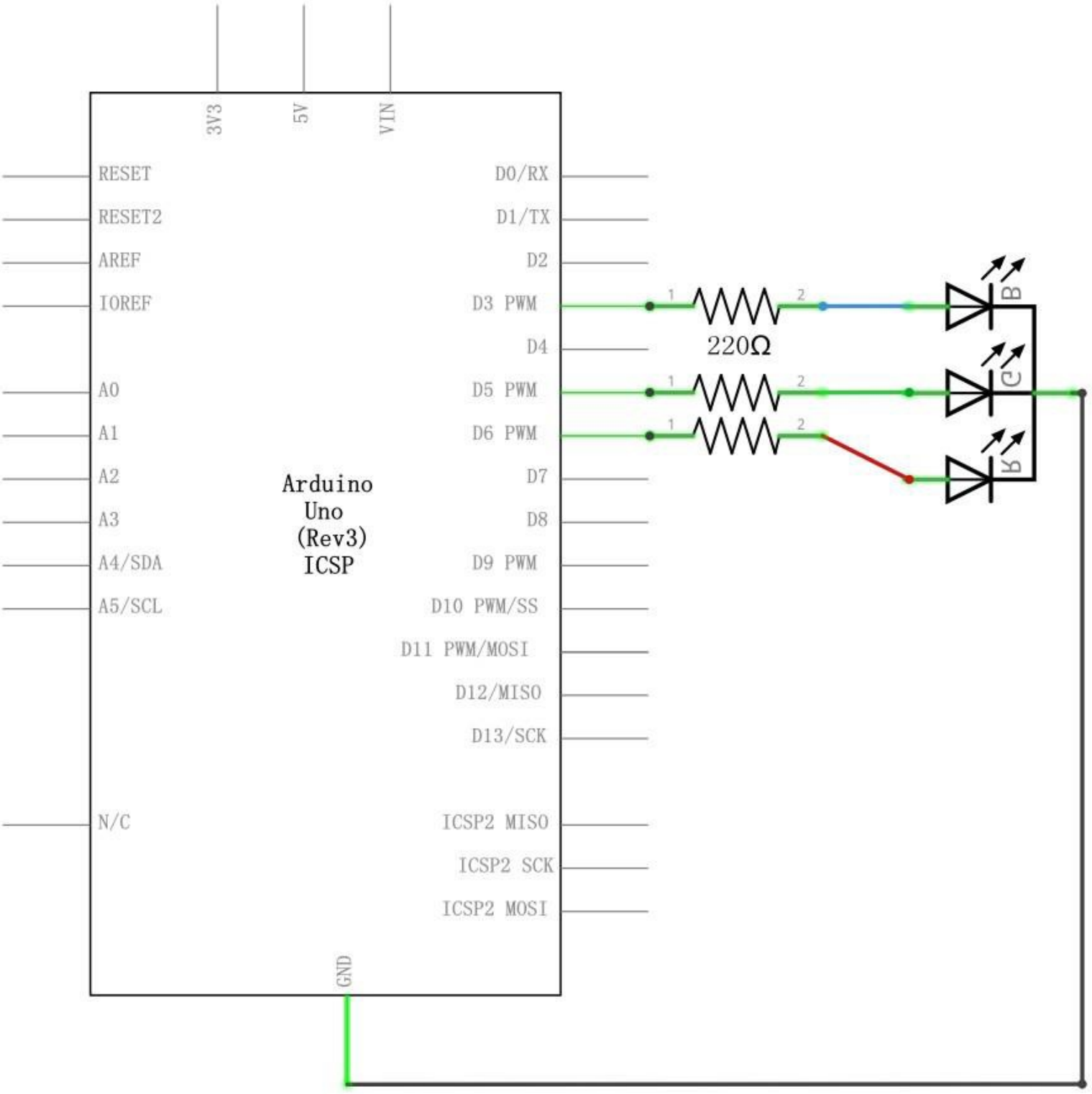
パルス幅変調（PWM）は電力を制御する技術です。 また、各 LED の輝度を制御するためにここで使用します。
下の図は、UNO の PWM ピンの 1 つからの信号を示しています。



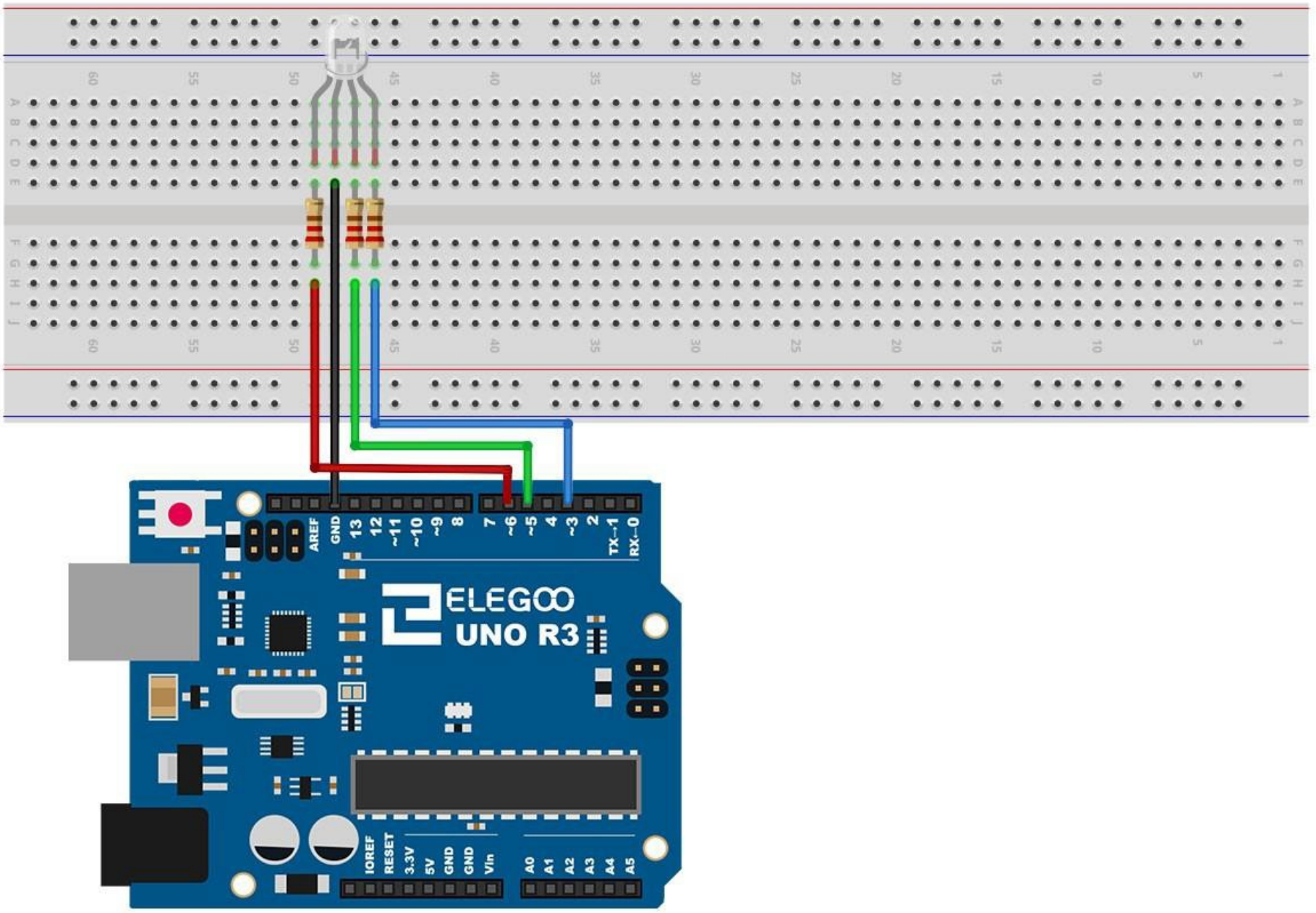
およそ 1/500 秒ごとに、PWM 出力はパルスを生成します。
このパルスの長さは、 'analogWrite'関数によって制御されます。 したがって、 'analogWrite（0）'はパルスをまったく生成せず、 'analogWrite（255）'は次のパルスが到着するまですべての時間持続するパルスを生成します。その結果、実際には常に出力されます。
analogWrite に 0 から 255 の間の値を指定すると、パルスが生成されます。 出力パルスが時間の 5%だけ高い場合、駆動しているものは全出力の 5%しか受け取りません。
しかし、出力が時間の 90%の間 5V であれば、負荷はそれに供給される電力の 90%を得ます。 そのスピードで LED の点滅が見えないので、明るさが変わっているように見えます。

Connection

Schematic



Wiring diagram



Code

配線後、コードフォルダのレッスン 4 の RGB LED でプログラムを開き、UPLOAD をクリックしてプログラムをアップロードしてください。エラーがある場合のプログラムアップロードの詳細については、レッスン 2 を参照してください。

私たちのコードは FOR ループを使って色を循環させます。 最初の FOR ループは RED から GREEN になります。
2 番目の FOR ループは GREEN から BLUE になります。 最後の FOR ループは BLUE から RED になります。
スケッチを試してみると、それを詳細に解剖する.....

スケッチは、各色にどのピンを使用するかを指定することから始まります:

```
// Define Pins
#define BLUE 3
#define GREEN 5
#define RED 6
```

次のステップは 'setup'関数を書くことです。 以前のレッスンで学んだように、セットアップ機能は Arduino がリセットされた後に 1 回だけ実行されます。 この場合、出力として使用している 3 つのピンを定義するだけです。

```
void setup()
{
pinMode(RED, OUTPUT);
pinMode(GREEN, OUTPUT);
pinMode(BLUE, OUTPUT);
digitalWrite(RED, HIGH);
digitalWrite(GREEN, LOW);
digitalWrite(BLUE, LOW);
}
```

'loop'関数を見る前に、スケッチの最後の関数を見てみましょう。
定義変数

```
redValue = 255; // choose a value between 1 and 255 to change thecolor.
greenValue = 0;
```

blueValue = 0;

この関数は 3 つの引数を取ります.1 つは赤、緑、青の LED の明るさです。 いずれの場合も、数値は 0∽255 の範囲内にあり、0 はオフ、255 は最大輝度を意味します。 次に、この関数は 'analogWrite' を呼び出して各 LED の輝度を設定します。

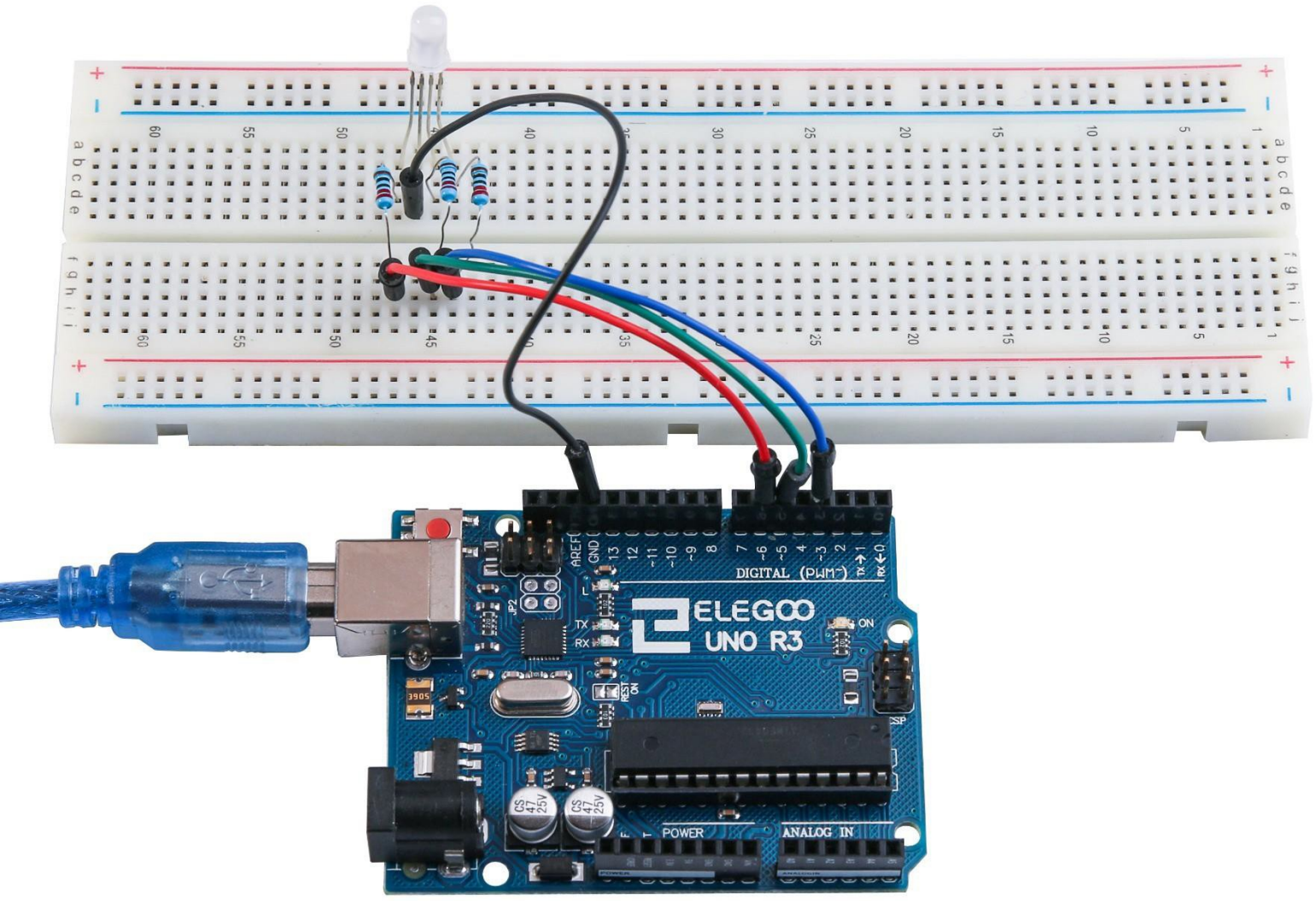
'ループ'機能を見ると、赤色、緑色、青色の光量を表示してから 1 秒間一時停止してから次の色に移動することがわかります。

#define delayTime 10 // fading time between colors

Delay(delayTime);

スケッチにあなた自身のいくつかの色を追加し、あなたの LED の効果を見てみてください。

Example picture



Lesson 5 デジタル入力

概要

このレッスンでは、LED をオン/オフするためにデジタル入力付きのプッシュボタンを使用する方法を学習します。
ボタンを押すと LED が点灯します。 もう一方のボタンを押すと LED が消灯します。

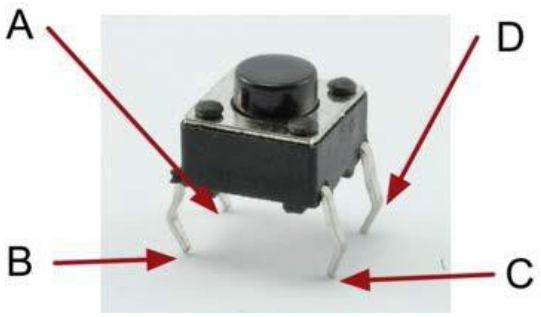
必要な構成部品:

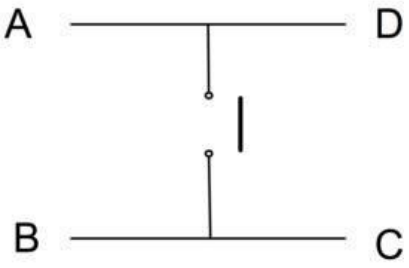
(1) x Elegoo Uno R3
(1) x 830 Tie-points Breadboard
(1) x 5mm red LED
(1) x 220 ohm resistor
(2) x push switches
(7) x M-M wires (Male to Male jumper wires)

部品の紹介

PUSH SWITCHES:

スイッチは本当にシンプルなコンポーネントです。 ボタンを押すかレバーを回すと、2 つの接点が電気的に接続されます。
このレッスンで使用されているソフトタッチスイッチには 4 つの足があり、少し紛らわしいものです。

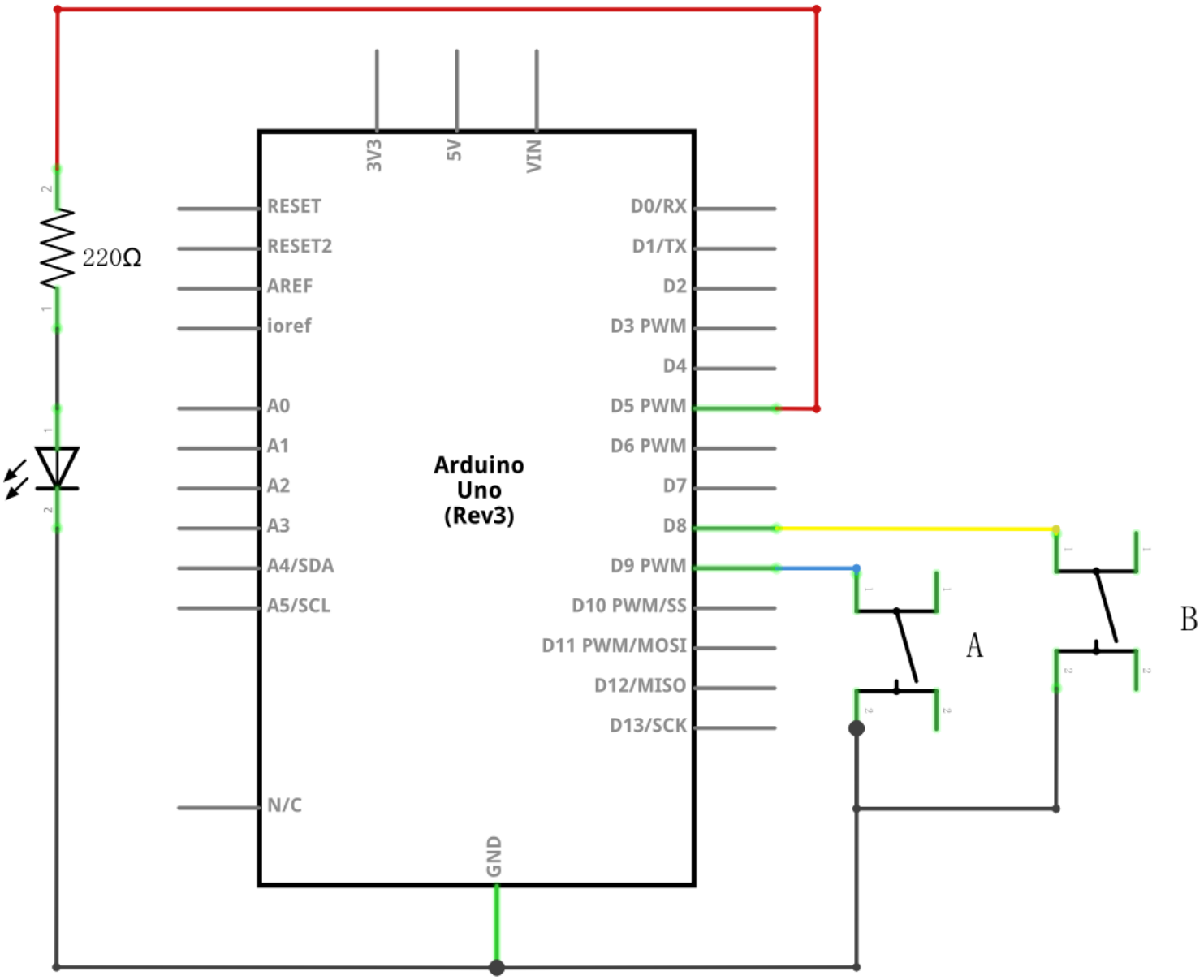




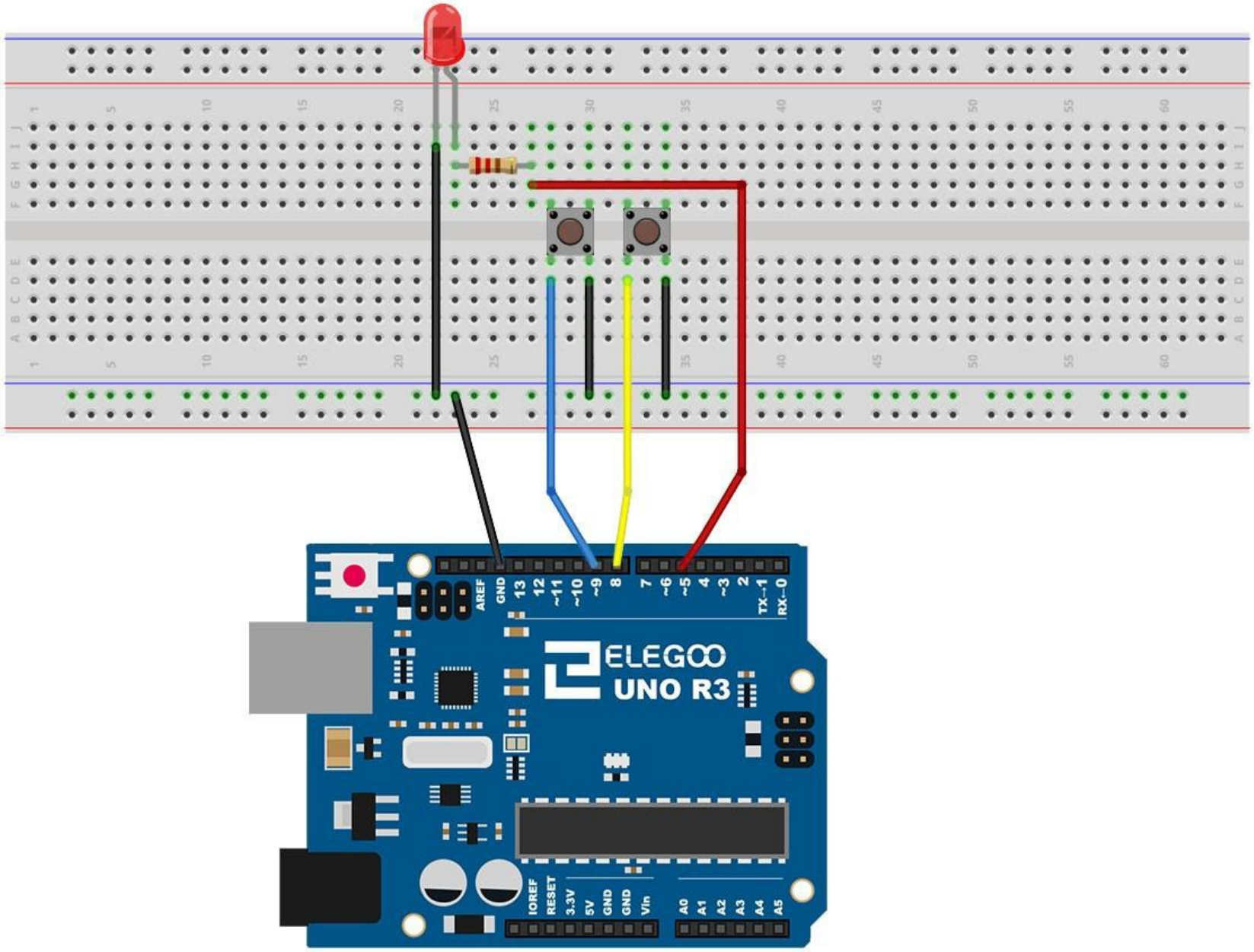
実際には、実際には電気接続が 2 つしかありません。 スイッチパッケージ内では、A と D のように
ピン B と C が一緒に接続されています。

Connection

Schematic



Wiring diagram



スイッチの本体は正方形であるが、ピンはスイッチの両側から突出している。 これは、ピンがブレッドボード上に正しく配置されているときにピンが離れていることを意味します。
LED は、左に短い負のリード線を持っていなければならないことに注意してください。

Code

配線後、コードフォルダのレッスン 5 デジタル入力でプログラムを開き、UPLOAD を押してプログラムをアップロードしてください。 エラーが表示される場合は、プログラムのアップロードに関するチュートリアルの詳細については、レッスン 2 を参照してください。

あなたの UNO ボードにスケッチをロードします。 左ボタンを押すと LED が点灯し、右ボタンを押すと消灯します。

スケッチの最初の部分は、使用される 3 つのピンの 3 つの変数を定義します。'ledPin'は出力ピンで、'buttonApin'はブレッドボードの上部に近いスイッチを指し、 'buttonBpin'は他のスイッチを指します。

'setup'関数は ledPin を OUTPUT として通常のように定義しますが、ここでは 2 つの入力を処理します。 この場合、pinMode を次のように 'INPUT_PULLUP'に設定します:

```
pinMode(buttonApin, INPUT_PULLUP);
pinMode(buttonBpin, INPUT_PULLUP);
```

INPUT_PULLUP のピン・モードは、ピンを入力として使用することを意味しますが、他のものが入力に接続されていない場合は、「プルアップ」して HIGH にする必要があります。 言い換えれば、入力のデフォルト値は、ボタンを押す動作によって LOW に引かれない限り、HIGH です。
このため、スイッチは GND に接続されています。 スイッチを押すと、入力ピンが GND に接続されるため、ハイになりません。
入力が通常 HIGH で、ボタンを押したときに入力が LOW になるので、ロジックは少し上下逆です。
これを 'loop'関数で扱います。

```
void loop()
{
  if (digitalRead(buttonApin) == LOW)
  {
    digitalWrite(ledPin, HIGH);
  }
  if (digitalRead(buttonBpin) == LOW)
```

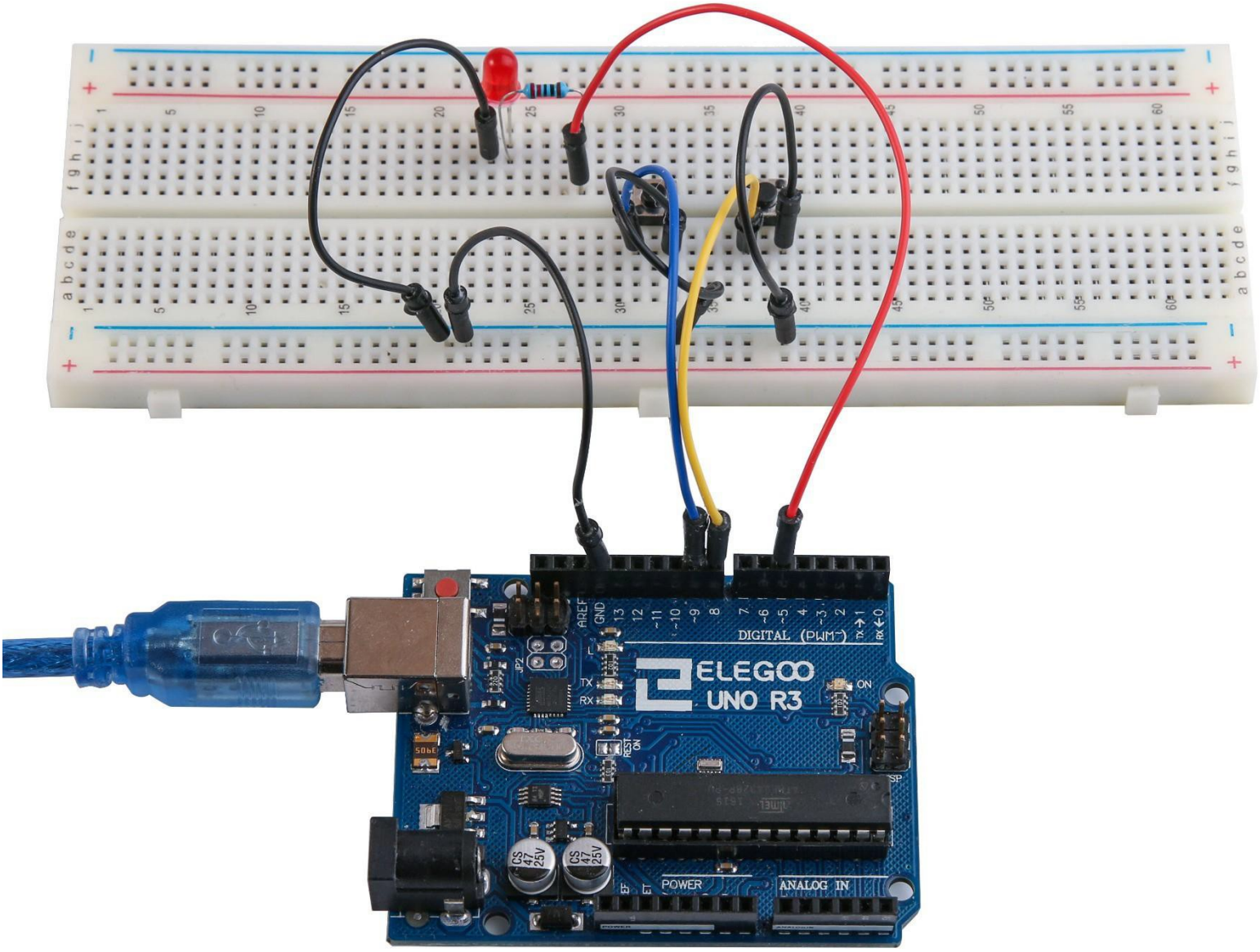
```
  {
    digitalWrite(ledPin, LOW);
  }
}
```

'loop'関数には 2 つの 'if'ステートメントがあります。 各ボタンに 1 つ。 それぞれが適切な入力で「digitalRead」を行います。

ボタンが押されると、対応する入力は LOW になり、ボタン A が低ければ、ledPin の 'digitalWrite' がオンになることを覚えておいてください。

同様に、ボタン B を押すと、LOW が ledPin に書き込まれます。

Example picture



Lesson 6 アクティブブザー

概要

このレッスンでは、アクティブなブザーでサウンドを生成する方法を学習します。

必要な構成部品:

(1) x Elegoo Uno R3
(1) x Active buzzer
(2) x F-M wires (Female to Male DuPont wires)

部品の紹介

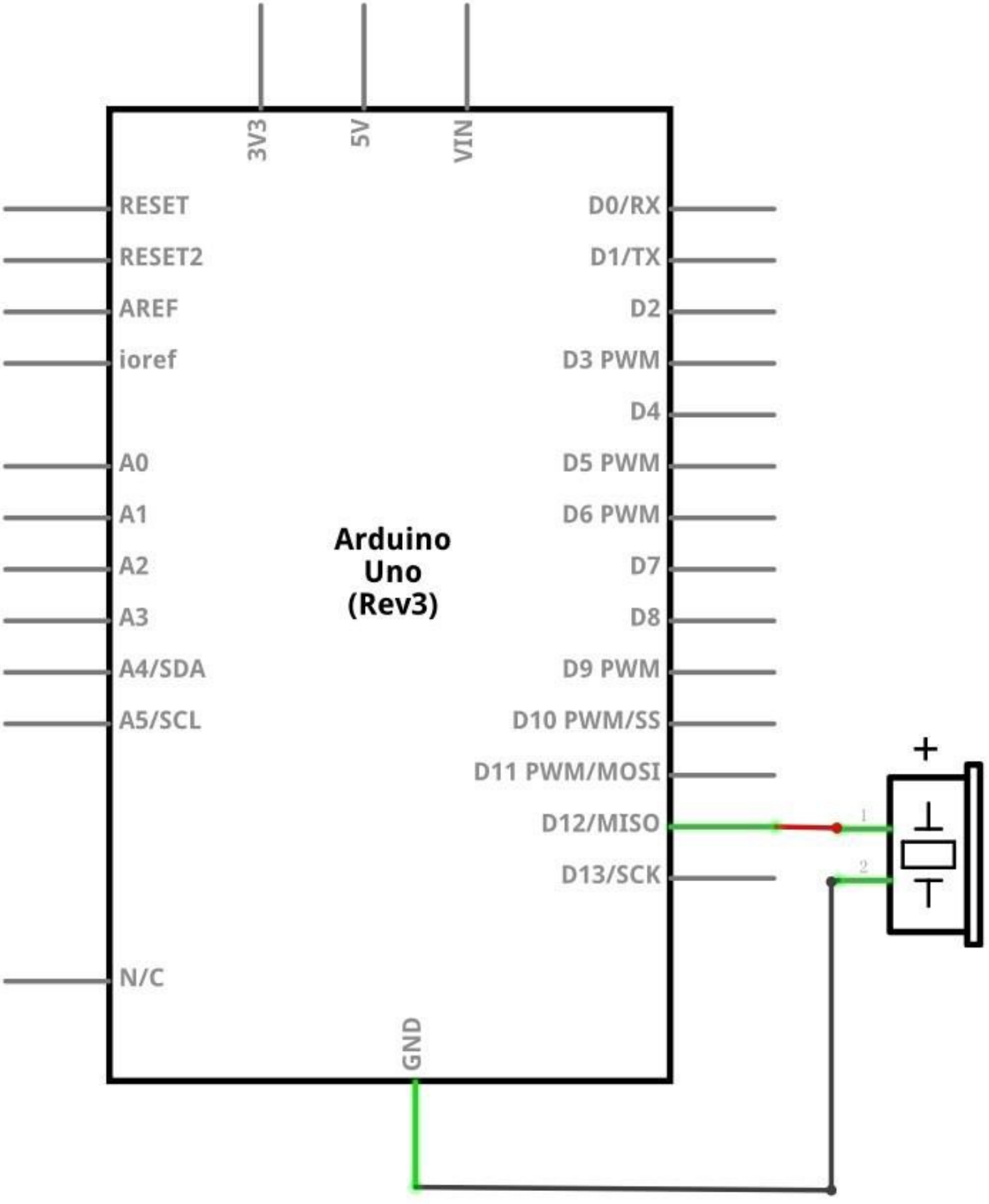
BUZZER:

電子ブザーは DC 電源であり、集積回路を備えています。 それらは、コンピュータ、プリンタ、複写機、アラーム、電子玩具、車載電子機器、電話機、タイマ、および音声デバイス用の他の電子製品に広く使用されている。 ブザーはアクティブなものとパッシブなものに分類できます。 2 つのブザーのピンを上に向けます。 緑色の回路基板を持つものは受動的なブザーですが、黒色のテープで囲まれたものはアクティブなブザーです。

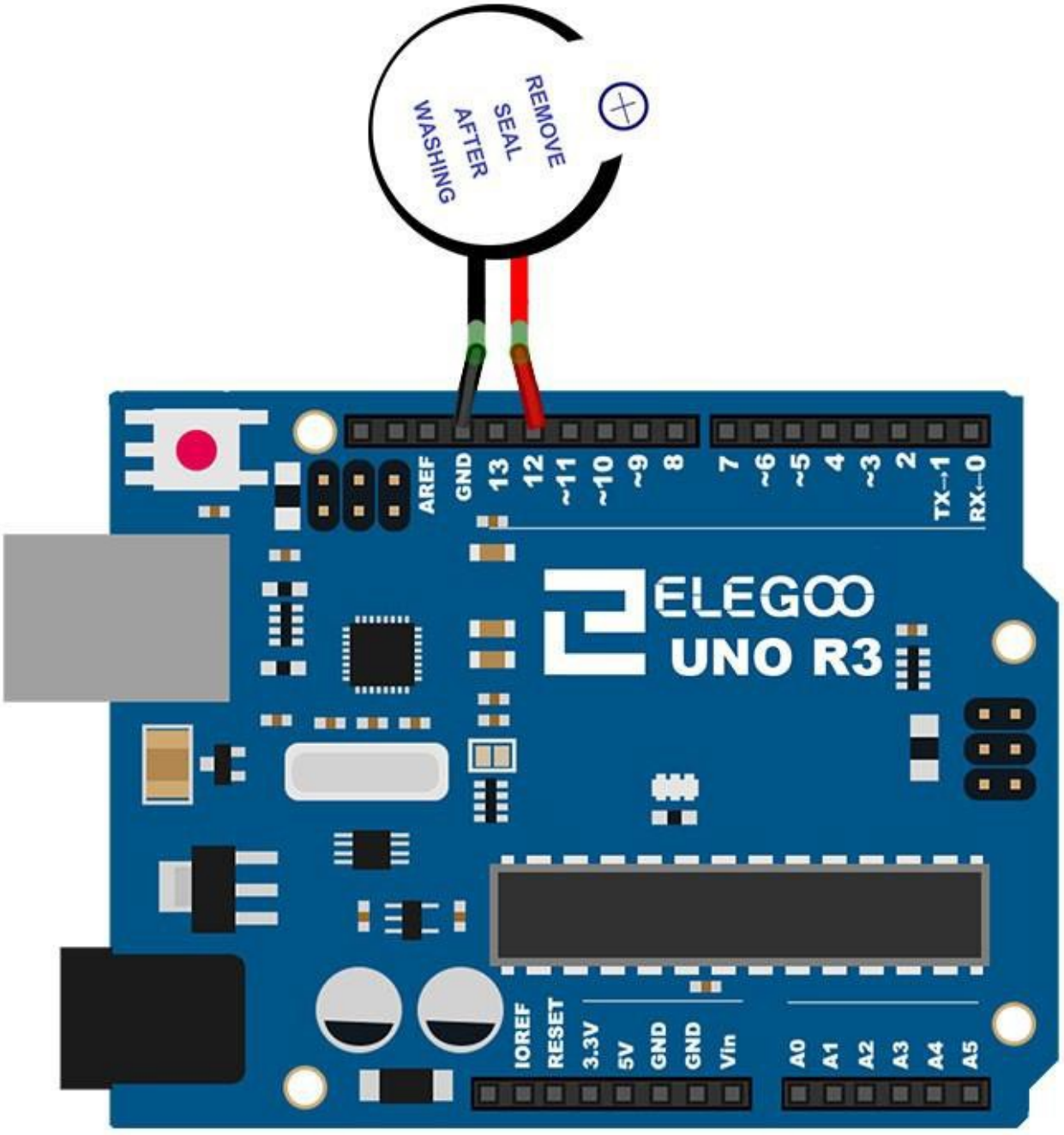
この 2 つの違いは、アクティブなブザーに内蔵振動があることです

音源があるので、電化すると音が鳴ります。 受動的なブザーはそのような音源を持っていないので、DC 信号が使われていればつぶれません。 代わりに、それを駆動するために周波数が 2K～5K の方形波を使用する必要があります。 アクティブなブザーは、複数の内蔵発振器のためにパッシブブザーよりしばしば高価です。

Connection


3V3
5V
VIN
RESET
RESET2
AREF
ioref
A0
A1
A2
A3
A4/SDA
A5/SCL
N/C
GND
D0/RX
D1/TX
D2
D3 PWM
D4
D5 PWM
D6 PWM
D7
D8
D9 PWM
D10 PWM/SS
D11 PWM/MOSI
D12/MISO
D13/SCK
Arduino
Uno
(Rev3)
+
1
2

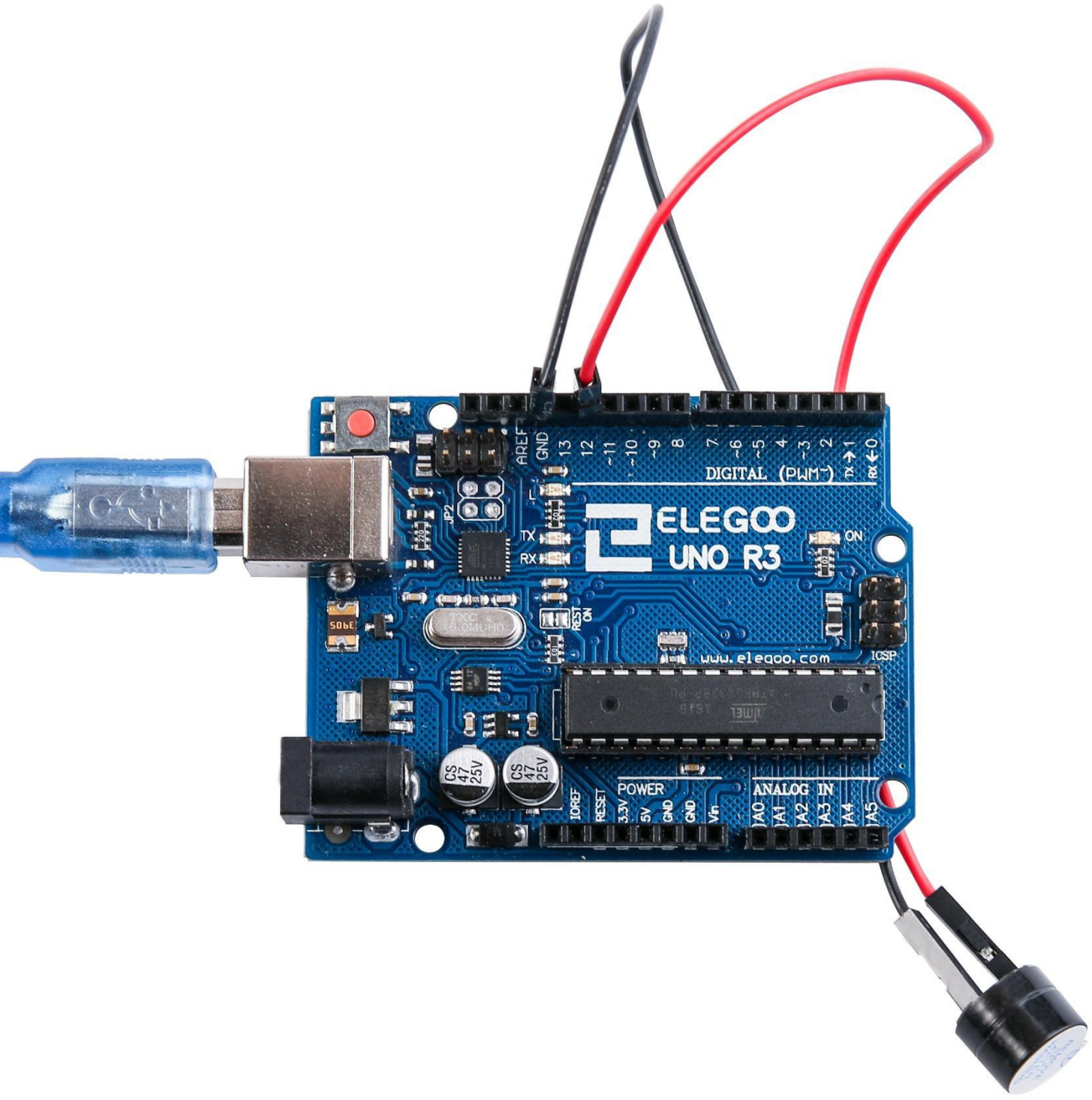
Wiring diagram



Code

配線後、コードフォルダのレッスン 6 のサウンドを作成し、UPLOAD をクリックしてプログラムをアップロードしてください。 エラーがある場合のプログラムアップロードの詳細については、レッスン 2 を参照してください。

Example picture



Lesson 7 傾斜ボールスイッチ

概要

このレッスンでは、小さな傾きを検出するためにチルトボールスイッチを使用する方法を学習します。

必要な構成部品:

(1) x Elegoo Uno R3
(1) x Tilt Ball switch
(2) x F-M wires (Female to Male DuPont wires)

部品の紹介

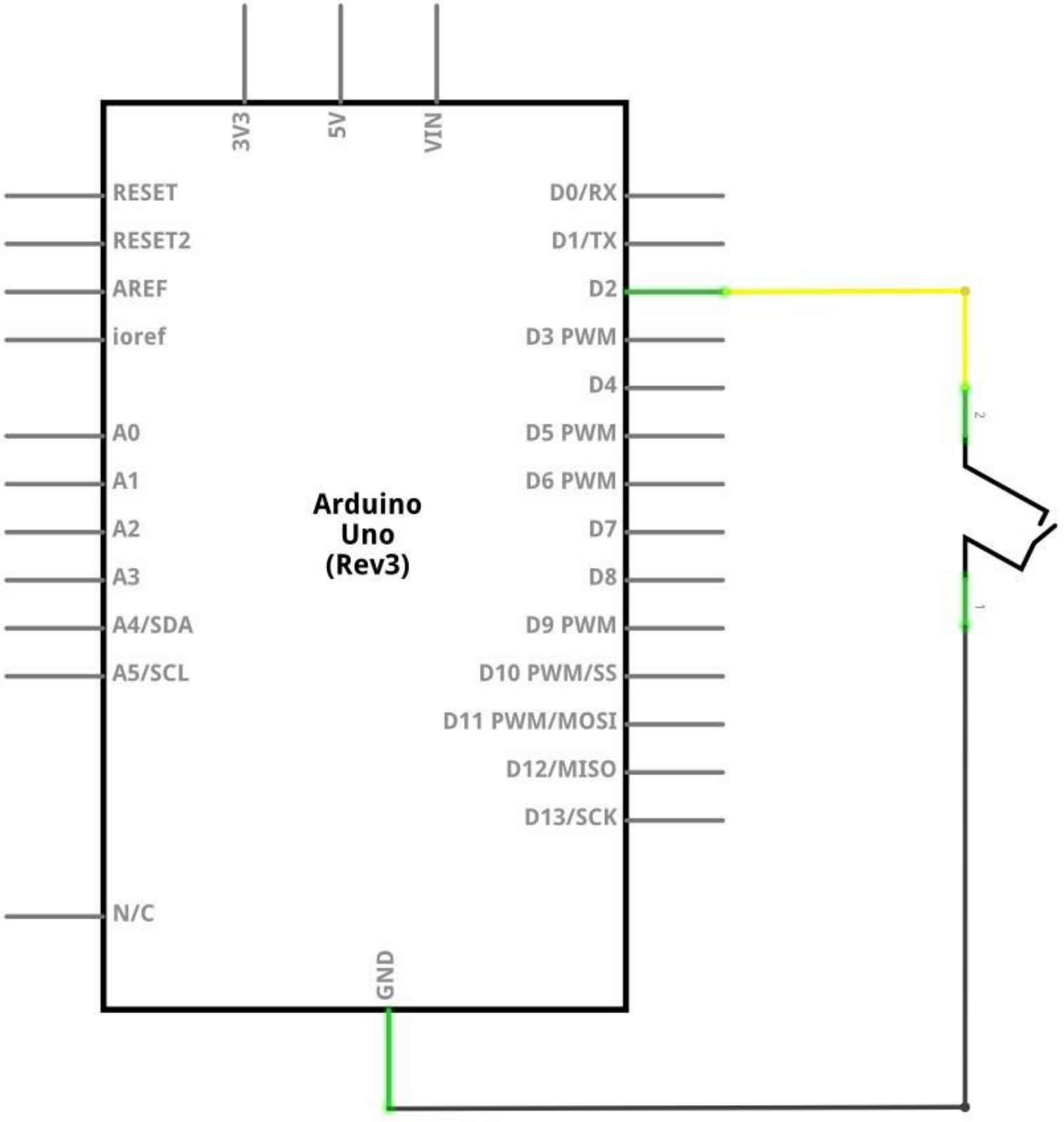
Tilt sensor:

傾斜センサ（傾斜ボールスイッチ）を使用すると、向きや傾きを検出できます。それらは小さく、安価で、低電力で使いやすいものです。正しく使用すると、磨耗しません。彼らのシンプルさは、おもちゃ、ガジェット、家電製品で人気があります。明らかな理由から、「水銀スイッチ」、「傾斜スイッチ」または「ローリングボールセンサー」と呼ばれることもあります。

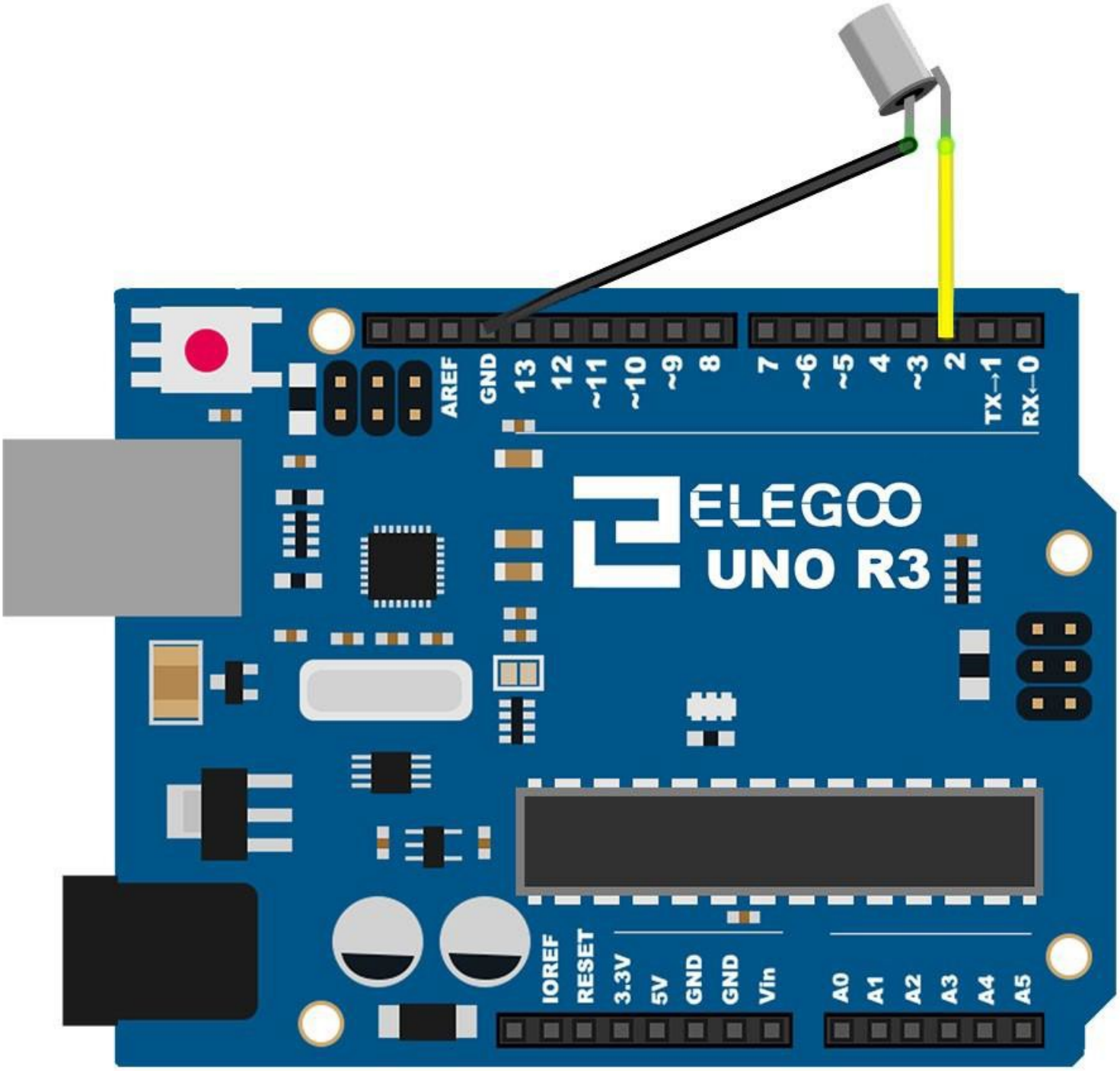
それらは通常、何らかの種類の空洞（円筒形が一般的ではありますが、必ずしも一般的ではありません）の中に、水銀や転がり球のような導電性の自由質量を内包しています。キャビティの一端は 2 つの導電性要素（極）を有する。センサがその端部が下方になるように配向されると、質量は極に転がり、それらを短絡させ、スイッチスローとして作用する。
完全な加速度計と同じくらい正確で柔軟性はありませんが、傾斜スイッチは動きや方向を検出できます。もう 1 つの利点は、大きなものが自分の力を切り替えることができるということです。一方、加速度計は、デジタルまたはアナログ電圧を出力し、余分な回路を使用して解析する必要があります。

Connection

Schematic



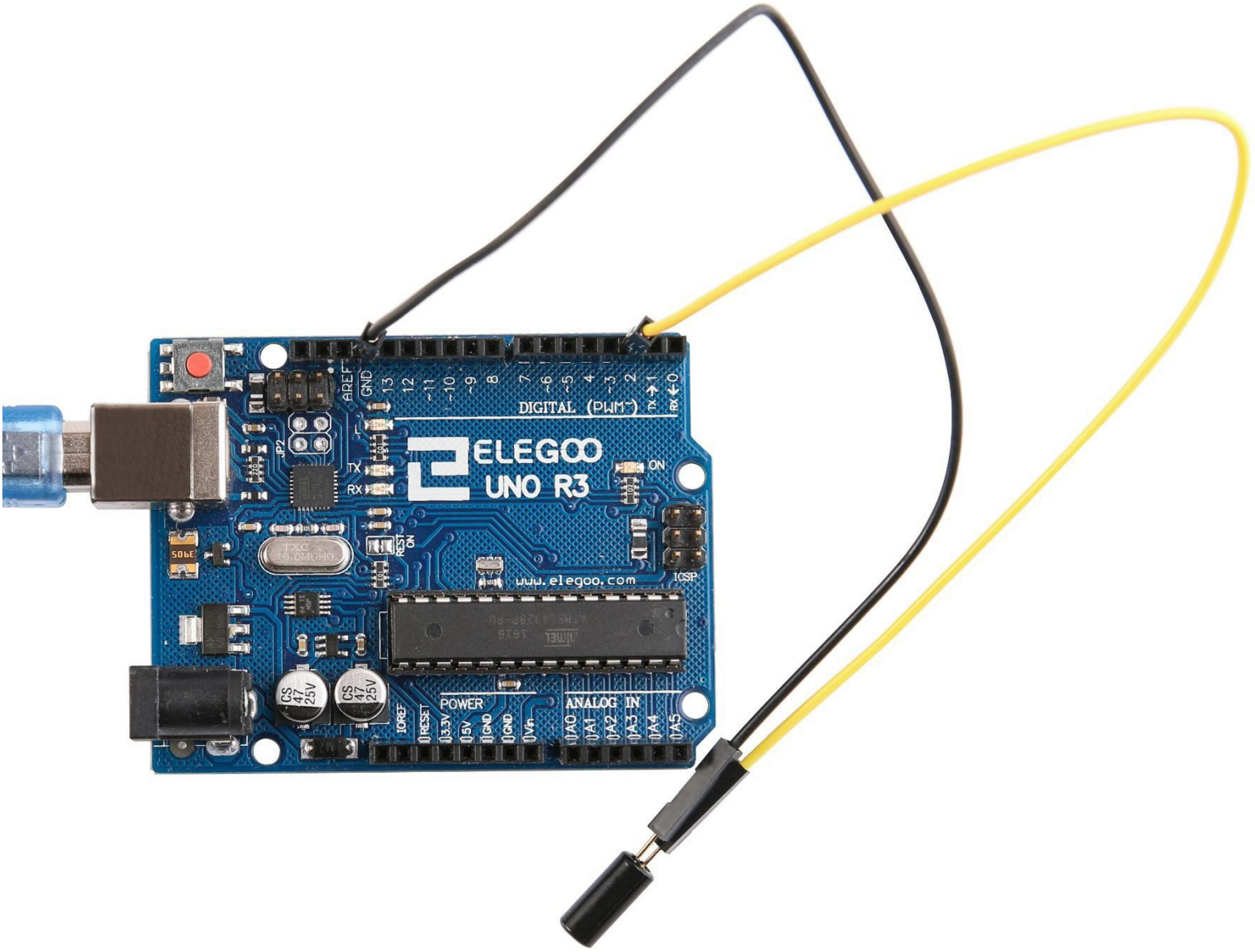
Wiring diagram



Code

配線後、コードフォルダのレッスン 8 ボールスイッチでプログラムを開き、UPLOAD をクリックしてプログラムをアップロードしてください。 エラーがある場合のプログラムアップロードの詳細については、レッスン 2 を参照してください。

Example picture



Lesson 8 Eight LED with 74HC595

概要

このレッスンでは、8 つの出力ピンをあきらめることなく、UNO で 8 つの大きな赤い LED を使用する方法を学習します。

UNO ピンに 8 個の LED を接続して抵抗を接続することもできますが、すぐに UNO のピンが使い尽くされます。 UNO にたくさんのものが繋がっていない場合。 そうすることは OK ですが、ボタンやセンサー、サーボなどが必要な時がありますが、ピンが残っていないことを知る前によく覚えておいてください。 したがって、これを行う代わりに、74HC595 シリアル・パラレル・コンバータと呼ばれるチップを使用することになります。 このチップには 8 つの出力（完璧）と 3 つの入力があり、一度に少しずつデータを入力します。

このチップは、LED を駆動するのが少し遅くなります（1 秒間に 8,000,000 回ではなく 1 秒間に約 500,000 回 LED を変えることができます）が、人間が検出できるよりも速く、速いです。

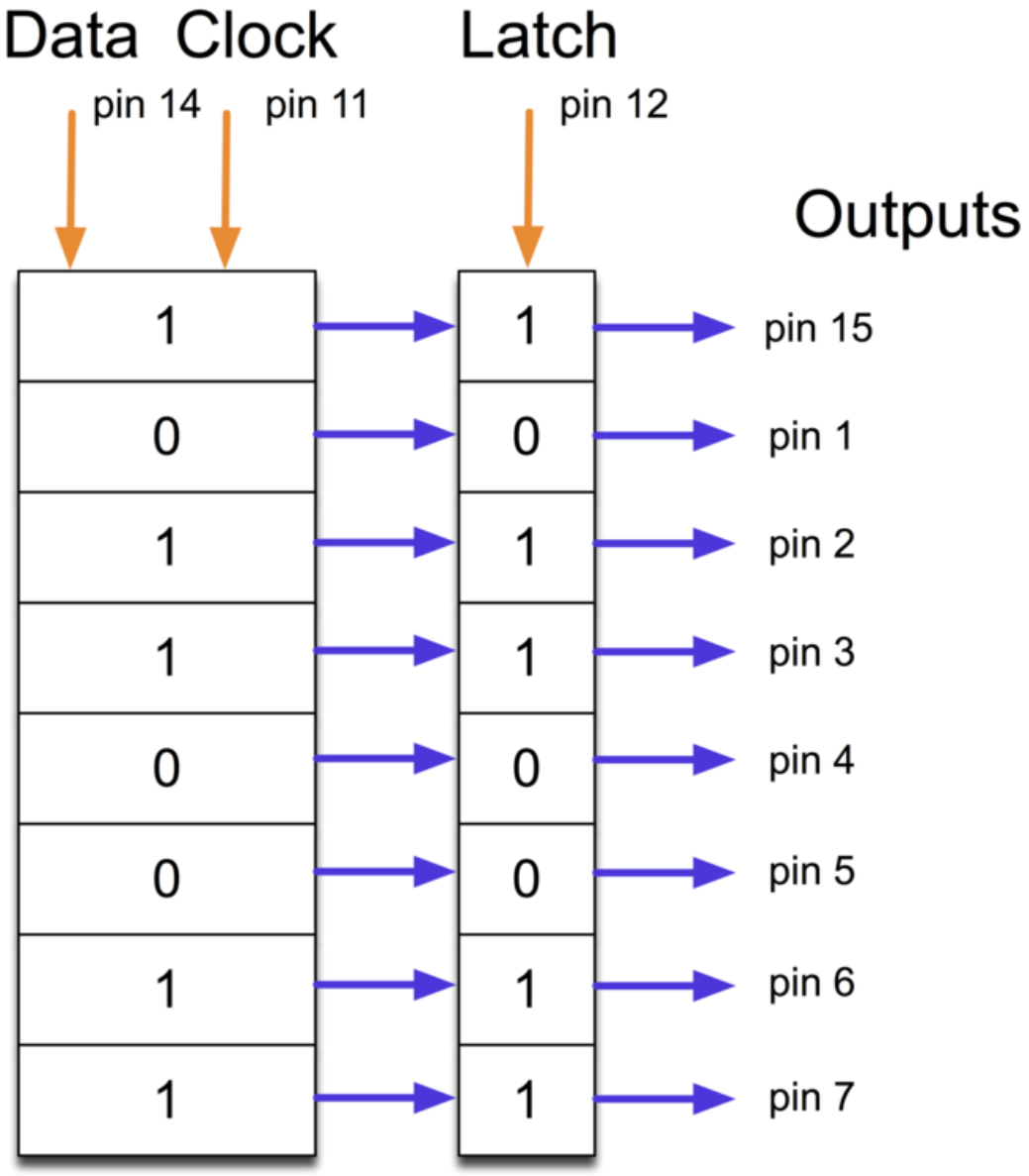
必要な構成部品:

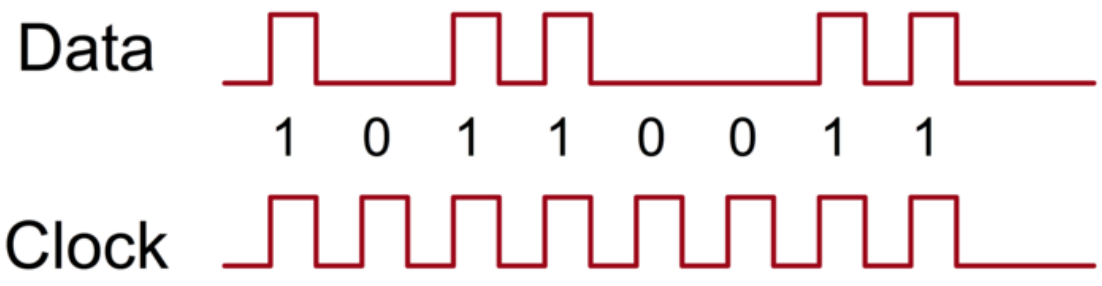
(1) x Elegoo Uno R3
(1) x 830 tie-points breadboard
(8) x leds
(8) x 220 ohm resistors
(1) x 74hc595 IC
(14) x M-M wires (Male to Male jumper wires)

部品の紹介

74HC595 Shift Register:

シフトレジスタは、8 つのメモリ位置として考えられるものを保持する一種のチップであり、それぞれが 1 または 0 のいずれかになります。これらの値をそれぞれオンまたはオフに設定するには、 チップの「データ」ピンと「クロック」ピンです。



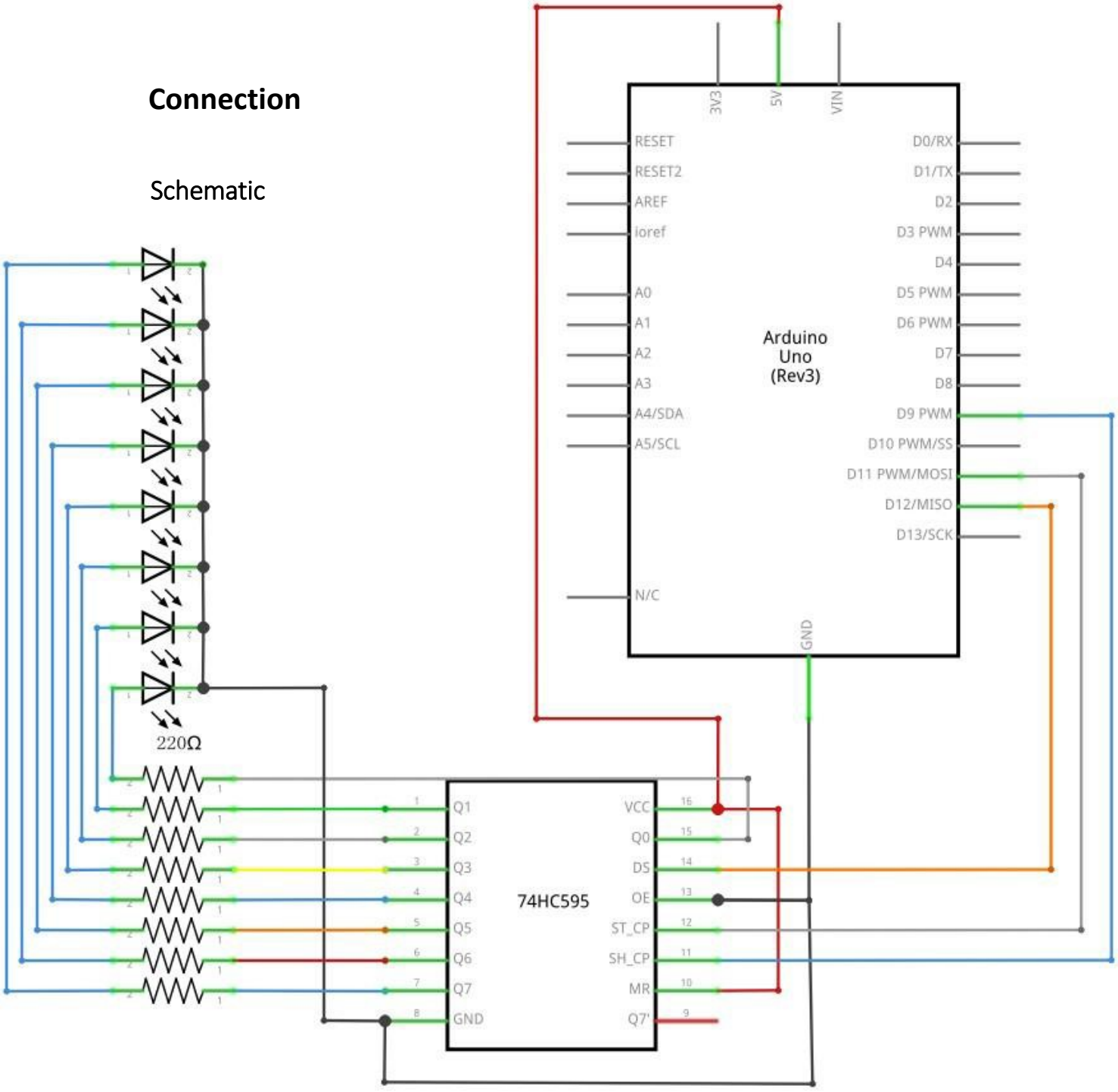


クロックピンは 8 個のパルスを受信する必要があります。 各パルスで、データピンがハイの場合、1 がシフトレジスタにプッシュされます。 8 個のパルスがすべて受信されると、 'Latch'ピンを有効にすると 8 個の値がラッチ・レジスタにコピーされます。 これは必要です。 そうしないと、データがシフトレジスタにロードされている間、間違った LED が点滅します。

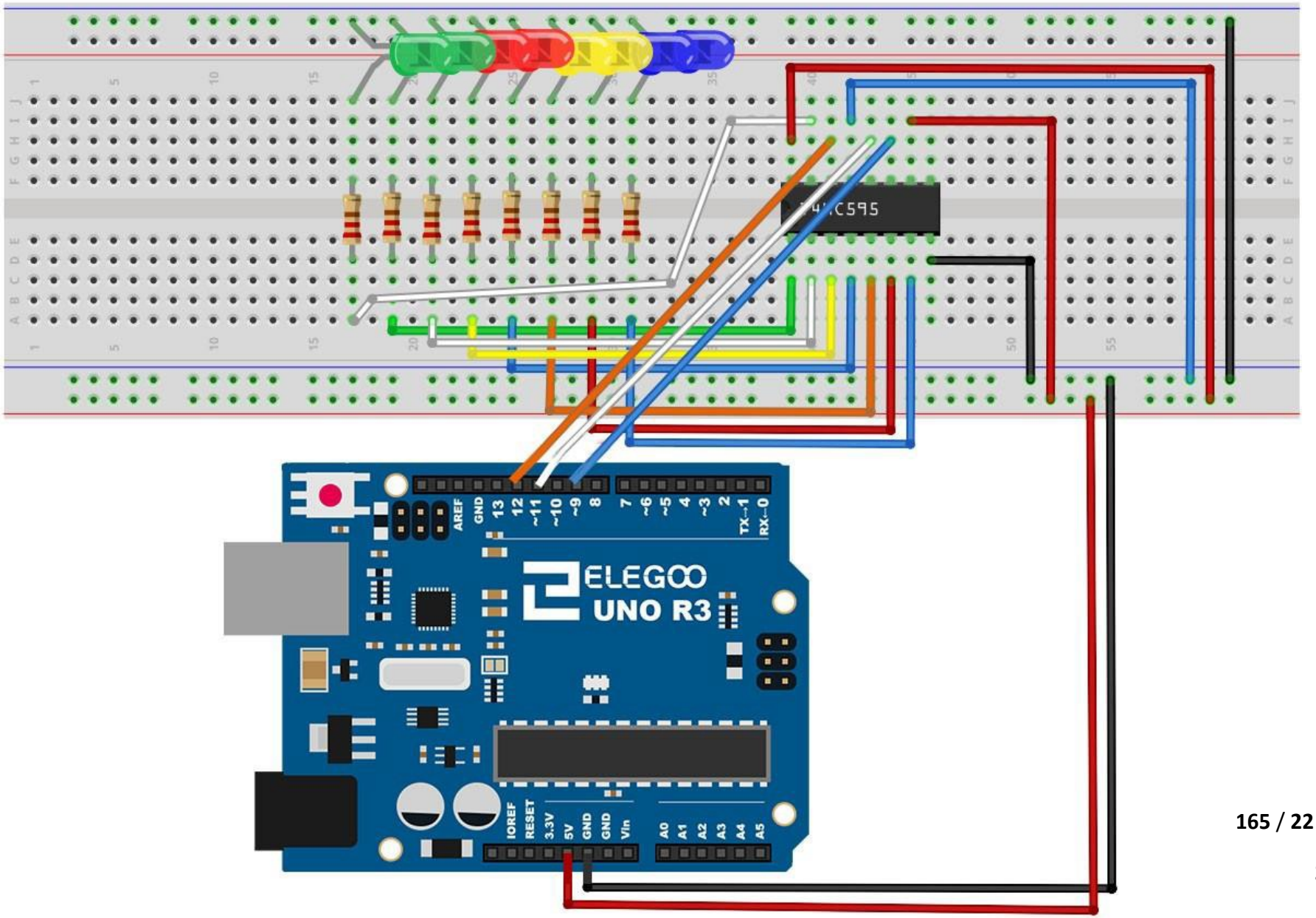
また、チップには出力イネーブル（OE）ピンがあり、これを使用して出力をすべて一度にイネーブルまたはディセーブルします。 これを PWM 対応の UNO ピンに接続し、 'analogWrite'を使用して LED の輝度を制御することができます。このピンはアクティブローであるため、GND に接続します。

Connection

Schematic



Wiring diagram



8 つの LED と 8 つの抵抗を接続するので、実際にはかなりの数の接続が必要です。

他のすべてのものがそれに接続するので、おそらく 74HC595 チップを最初に置くのが最も簡単です。 小さな U 字型ノッチがブレッドボードの上部に向くようにしてください。 チップのピン 1 はこのノッチの左側にあります。

UNO からのデジタル 12 がシフトレジスタの 14 番ピンに行きます UNO からのデジタル 11 がシフトレジスタの 12 番ピンに行きますデジタル UNO からデジタル 9 番がシフトレジスタのピン＃11 に行きます

IC の出力の 1 つを除いて、すべてがチップの左側にあります。 したがって、接続を容易にするために、LED も同様です。

チップの後、抵抗器を所定の位置に配置します。 抵抗器のリードのどれも互いに接触していないことに注意する必要があります。 電源を UNO に接続する前に、これを再度確認する必要があります。 リードが接触していない状態で抵抗器を配置することが困難な場合は、リード線を短くしてブレッドボードの表面に近づけるようにしてください。

次に、ブレッドボードに LED を配置します。 より長い正の LED 導線は、チップ上にあるブレッドボードのどちらの側にもなければなりません。

上記のようにジャンパーリードを取り付けます。 IC のピン 8 からブレッドボードの GND カラムに行くものを忘れないでください。

少し後でリストされたスケッチをロードし、それを試してみてください。 すべての LED が点灯するまで各 LED が順番に点灯し、すべて消灯してサイクルが繰り返されます。

Code

配線後、コードフォルダにあるプログラムを開きます - レッスン 24 8 つの LED と 74HC595 を開き、アップロードをクリックしてプログラムをアップロードします。エラーがある場合のプログラムアップロードの詳細については、レッスン 2 を参照してください。

最初に行うのは、使用する 3 つのピンを定義することです。 これらは UNO

74HC595 のラッチ、クロック、およびデータ・ピンに接続されるデジタル出力を備えています。

```
int latchPin = 11; int clockPin = 9; int dataPin
= 12;
```

次に、 'leds'と呼ばれる変数が定義されます。 これは、LED が現在オンまたはオフになっているパターンを保持するために使用されます。 タイプ「バイト」のデータは、8 ビットを使用する数字を表す。 各ビットはオンまたはオフのどちらでもかまいませんので、これは私たちの 8 つの LED のうちのどれがオンまたはオフになっていますか?

```
 byte leds = 0;
```

「セットアップ」機能は、使用している 3 つのピンをデジタル出力に設定するだけです。

```
 void setup()
{
   pinMode(latchPin,      OUTPUT);      pinMode(dataPin,
   OUTPUT); pinMode(clockPin, OUTPUT);
}
```

'loop'関数は最初、変数 'leds'に値 0 を渡して、すべての LED をオフにします。次に、 'leds'パターンをシフトレジスタに送り、すべての LED がオフになる 'updateShiftRegister'を呼び出します。 後'updateShiftRegister'がどのように動作するかを扱います。

ループ関数は 0.5 秒間停止し、 'for'ループと変数 'i'を使用して 0 から 7 までカウントし始めます。 毎回、Arduino 関数 'bitSet'を

使用して、変数 'leds'の LED を制御するビットを設定します。 次に、 'updateShiftRegister'を呼び出して、leds が変数'leds 'にあるものを反映するように更新します。 次に、「i」がインクリメントされ、次の LED が点灯するまでに半秒の遅延があります。

```
 void loop()
{
   leds = 0; updateShiftRegister(); delay(500);
   for (int i = 0; i < 8; i++)
   {
      bitSet(leds, i); updateShiftRegister(); delay(500);
   }
}
```

関数 'updateShiftRegister'は、最初に latchPin をローに設定し、次に 'latchPin'を再びハイにする前に UNO 関数 'shiftOut'を呼び出します。 これは 4 つのパラメータをとり、最初の 2 つはデータとクロックにそれぞれ使用するピンです。

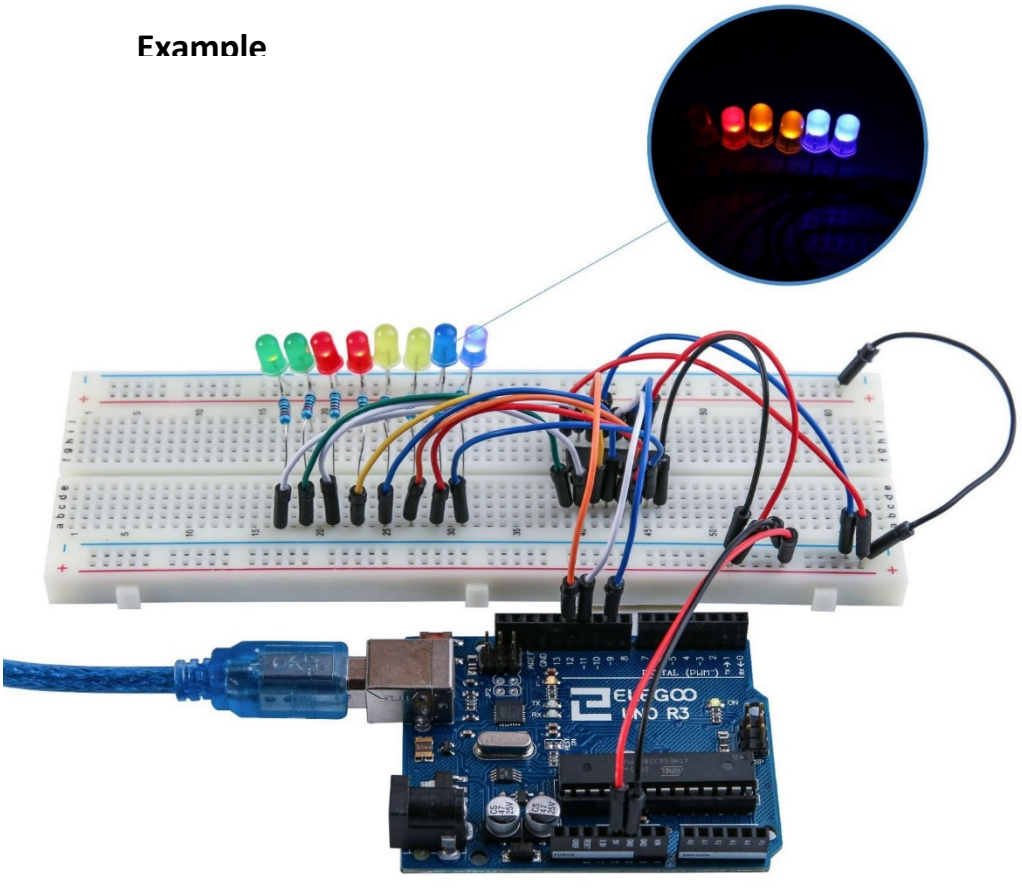
3 番目のパラメータは、データのどの部分を開始するかを指定します。 私たちは、最も重要なビットから始めるつもりです。これは、「最下位ビット」です。Bit' (LSB).

最後のパラメータは、シフトレジスタにシフトされる実際のデータです（この場合は 'leds'です）。

```
void updateShiftRegister()
{
digitalWrite(latchPin, LOW); shiftOut(dataPin, clockPin, LSBFIRST, leds); digitalWrite(latchPin, HIGH);
}
```

LED の 1 つをオンにするのではなくオフにしたい場合は、同様の Arduino 関数 (bitClear) を 'leds'変数でコールします。 これにより、'leds'のビットが 0 に設定され、実際の LED を更新するために 'updateShiftRegister'を呼び出すだけでそれに続く必要があります。

Example



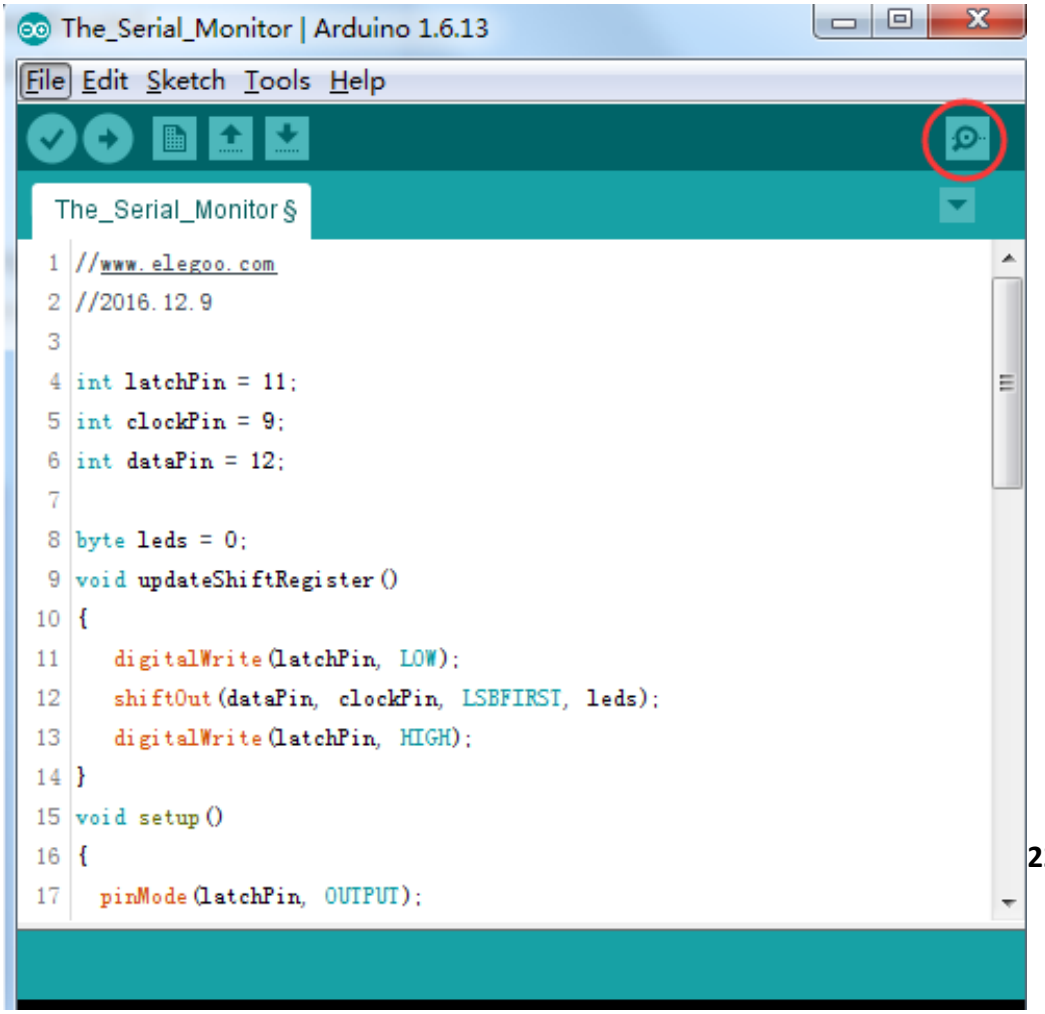
Lesson 9 シリアルモニタ

概要

このレッスンでは、レッスン 24 をベースに、Arduino シリアルモニターを使用してコンピュータから LED を制御する機能を追加します。 シリアルモニタは、コンピュータと UNO との間の「つなぎ」です。 それは、デバッグに便利で、キーボードから UNO を制御するために、テキストメッセージを送受信することができます! たとえば、コンピュータからコマンドを送信して LED を点灯させることができます。 このレッスンでは、レッスン 24 とまったく同じ部品と同様のブレッドボードレイアウトを使用します。まだレッスン 8 に従っていない場合は、レッスン 8 に従ってください。

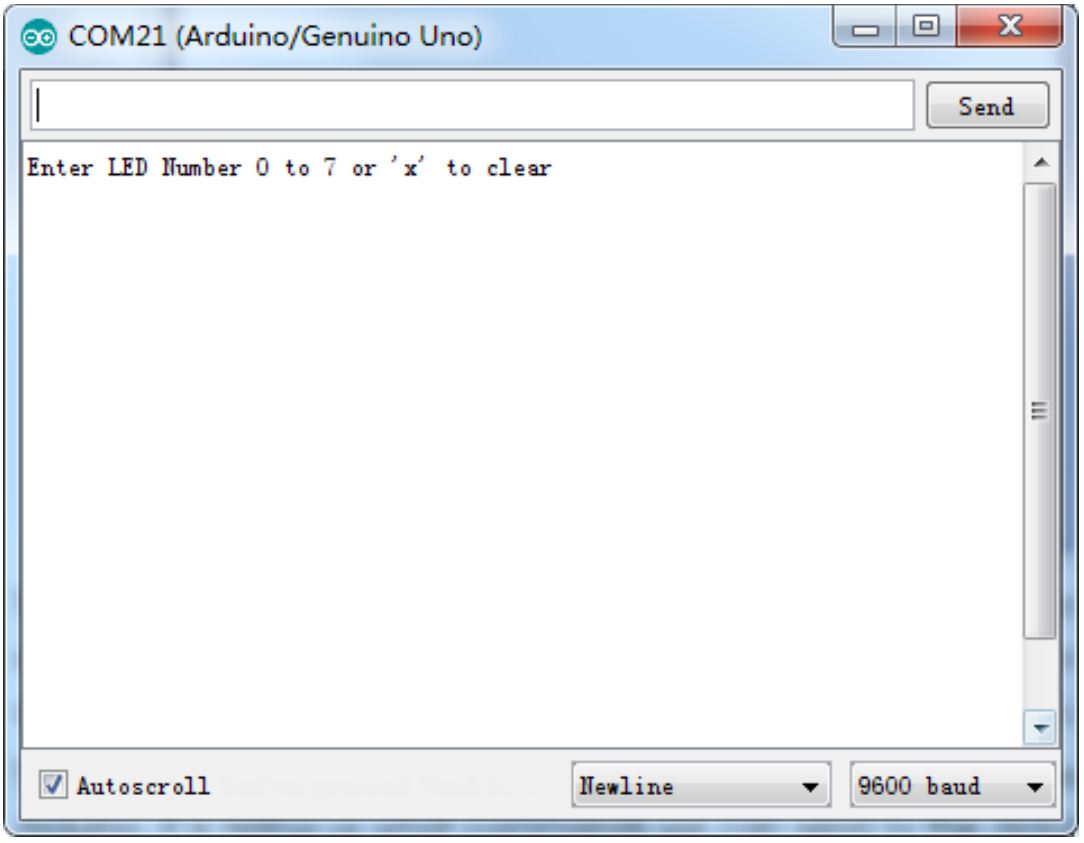
Steps taken

このスケッチをあなたの UNO にアップロードしたら、Arduino IDE のツールバーの一番右のボタンをクリックしてください。 ボタンは下に丸で囲まれています。



次のウィンドウが開きます.

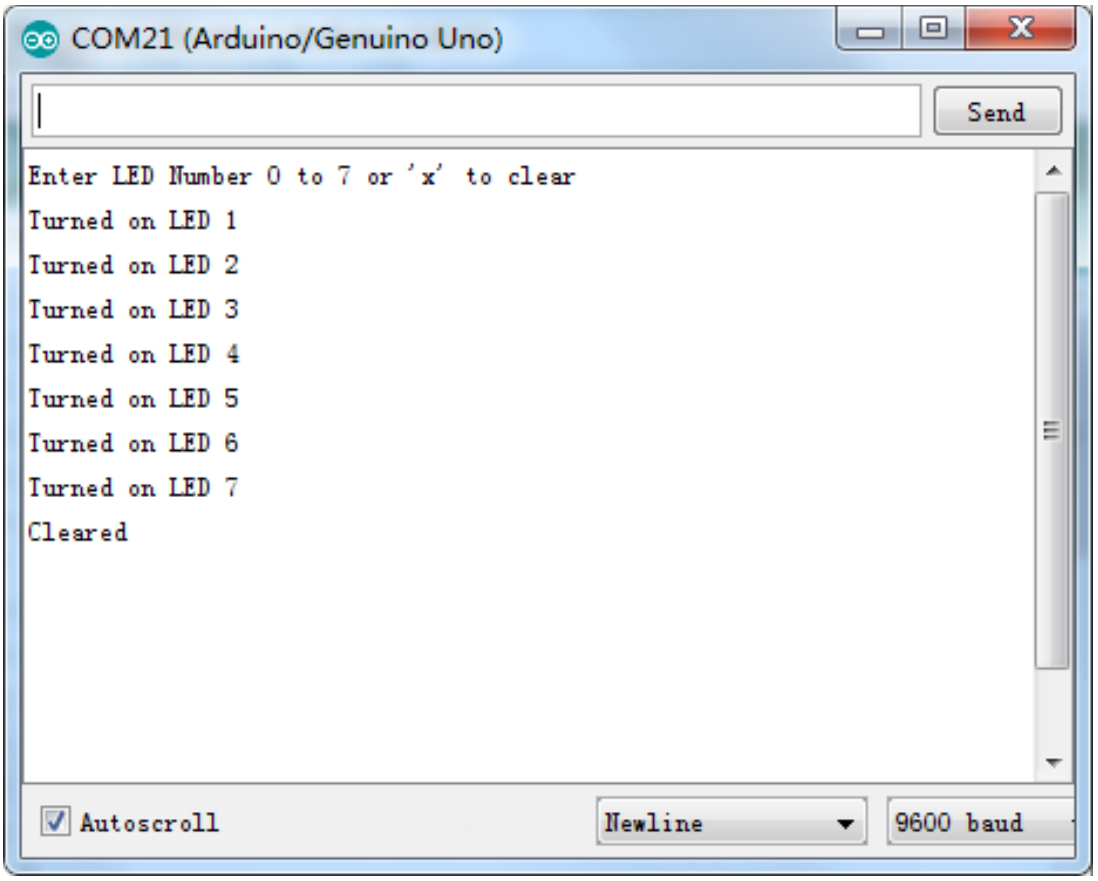
[シリアルモニタ]ボタンをクリックしてシリアルモニタをオンにします。 シリアルモニタの基本については、レッスン 1 で詳しく説明しています。



このウィンドウはシリアルモニタと呼ばれ、Arduino IDE ソフトウェアの一部です。 その仕事は、コンピュータから UNO ボード（USB 経由）へのメッセージの送信と、UNO からのメッセージの受信の両方を可能にすることです。

Arduino によって「LED 番号 0~7 を入力するか、x 'をクリアする」というメッセージが送信されました。 Arduino に送ることができるコマンドは、「x」（すべての LED をオフにする）またはオンにする LED の番号（0 は下の LED、1 は次の LED 1 つ、上の LED で 7 まで）。

シリアルモニタの上部に「送信」ボタンを押して、次のコマンドを入力してみてください。 これらの文字をそれぞれ入力した後、「送信」を押します。x 0 3 5
LED がすでにすべて消灯している場合は、x を入力しても効果はありませんが、各番号を入力すると対応する LED が点灯し、UNO ボードから確認メッセージが表示されます。 シリアルモニタが以下のように表示されます。



もう一度 x と入力して[送信]を押すと、すべての LED が消灯します。

Code

配線後、コードフォルダにあるプログラムを開きます - レッスン 25 シリアルモニタを開き、アップロードをクリックしてプログラムをアップロードしてください。エラーがある場合のプログラムアップロードの詳細については、レッスン 2 を参照してください。

ご想像のとおり、スケッチはレッスン 24 で使用されているスケッチに基づいています。そこでここでは新しいビットについて説明します。 Arduino IDE の完全なスケッチを参照すると便利です。

'setup'機能には、最後に 3 つの新しい行があります:

```
void setup()
{
  pinMode(latchPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
  updateShiftRegister();
  Serial.begin(9600);
```

```
  while (! Serial); // Wait until Serial is ready - Leonardo
  Serial.println("Enter LED Number 0 to 7 or 'x' toclear");
}
```

まず、 'Serial.begin（9600）'というコマンドがあります。これによりシリアル通信が開始され、UNO は USB 接続を介してコマンドを送信できます。値 9600 は接続の「ボーレート」と呼ばれます。これはデータの送信速度です。これをより高い値に変更できますが、Arduino シリアルモニターも同じ値に変更する必要があります。これについては後で説明します。今のところ、9600 のままにしておいてください。

'while'で始まる行は、Arduino がメッセージの送信を開始する前に、USB 接続のもう一方の端に何かがあることを保証します。そうしないと、メッセージは送信されても表示されないことがあります。このラインは、Arduino UNO が Arduino ボードを自動的にリセットするので、Arduino Leonardo を使用している場合にのみ必要です。これは、シリアルモニターを開いたときに自動的にリセットされますが、これは Leonardo では起こりません。

'setup'の最後の新しい行は、シリアルモニタの上部に表示されているメッセージを送信します。

「ループ」機能は、すべてのアクションが発生する場所です:

```
void loop()
{
  if (Serial.available())
  {
    char ch = Serial.read();
    if (ch >= '0' && ch <= '7')
    {
      int led = ch - '0';
      bitSet(leds, led);
      updateShiftRegister();
      Serial.print("Turned on LED ");
      Serial.println(led);
    }
    if (ch == 'x')
    {
      leds = 0;
      updateShiftRegister();
```

```
            Serial.println("Cleared");
        }
    }
}
```

ループ内で発生するすべてのものは、 'if'ステートメント内に含まれます。 だから、組み込みの Arduino 関数 'Serial.available（）'の呼び出しが 'true'でなければ、それ以外のことは起こりません。 Serial.available（）は、データが UNO に送信され、処理準備が整っていれば 'true'を返します。 受信メッセージはバッファと呼ばれるものに保持され、そのバッファが空でない場合は Serial.available（）が true を返します。

メッセージが受信された場合は、次のコード行に表示されます:

```
char ch = Serial.read();
```

これにより、バッファから次の文字が読み込まれ、バッファから削除されます。 また、変数 'ch'に割り当てます。 変数 'ch'は 'char'を表す 'char'型であり、名前が示唆するように、単一の文字を保持します。

シリアルモニタの上部にあるプロンプトの指示に従っている場合、この文字は 0 から 7 の間の 1 桁の数字か「x」の文字のいずれかになります。

次の行の 'if'ステートメントは、 'ch'が文字 '0'以上で、文字 '7'以下であることを確認して、1 桁であるかどうかをチェックします。 この方法で文字を比較するのが少し奇妙に見えますが、完全に受け入れられます。

各文字は、その ASCII 値と呼ばれる固有の番号で表されます。 これは、<=と> =を使用して文字を比較すると、実際には比較されていた ASCII 値であることを意味します。

テストに合格すると、次の行に進みます:

```
int led = ch – '0';
```

今は文字の算術演算を行っています！ 入力された数字のいずれかから数字「0」を減算しています。 したがって、 '0'を入力すると '0' - '0'は 0 になります。 '7'を入力すると '7' - '0'は実際に使用されている ASCII 値なので数字 7 と等しくなります 減算で

LED をオンにしたいということを知っているので、そのビットを変数 'leds'にセットし、シフトレジスタを更新するだけです。

```
bitSet(leds, led);
updateShiftRegister();
```

次の 2 行は、確認メッセージをシリアルモニタに書き戻します。

```
Serial.print("Turned on LED ");
Serial.println(led);
```

最初の行は Serial.println ではなく Serial.print を使用しています。 2 つの違いは、Serial.print は、そのパラメータに何も印刷した後に新しい行を開始しないということです。最初の行でこれを使用します。これは、メッセージを 2 つの部分に出力するためです。 最初に、一般的なビット： 'Turned on LED'と LED の番号。

LED の数は、テキスト文字列ではなく、「int」変数に保持されます。 Serial.print は、二重引用符で囲まれたテキスト文字列、または 'int'かそれともかなりの種類の変数をとることができます。

ケースを処理する 'if'ステートメントの後に、1 つの数字が処理された場合、 'ch'が文字 'x'であるかどうかを確認する 2 番目の 'if'ステートメントがあります。

```
    if (ch == 'x')
    {
      leds = 0;
      updateShiftRegister();
      Serial.println("Cleared");
    }
```

そうであれば、すべての LED をクリアし、確認メッセージを送信します。

Lesson 10 光電池

概要

このレッスンでは、アナログ入力を使用して光強度を測定する方法を学習します。 レッスン 8 で構築し、ライトのレベルを使用して点灯させる LED の数を制御します。
光電管は、鍋が上にあったブレッドボードの底にあります。

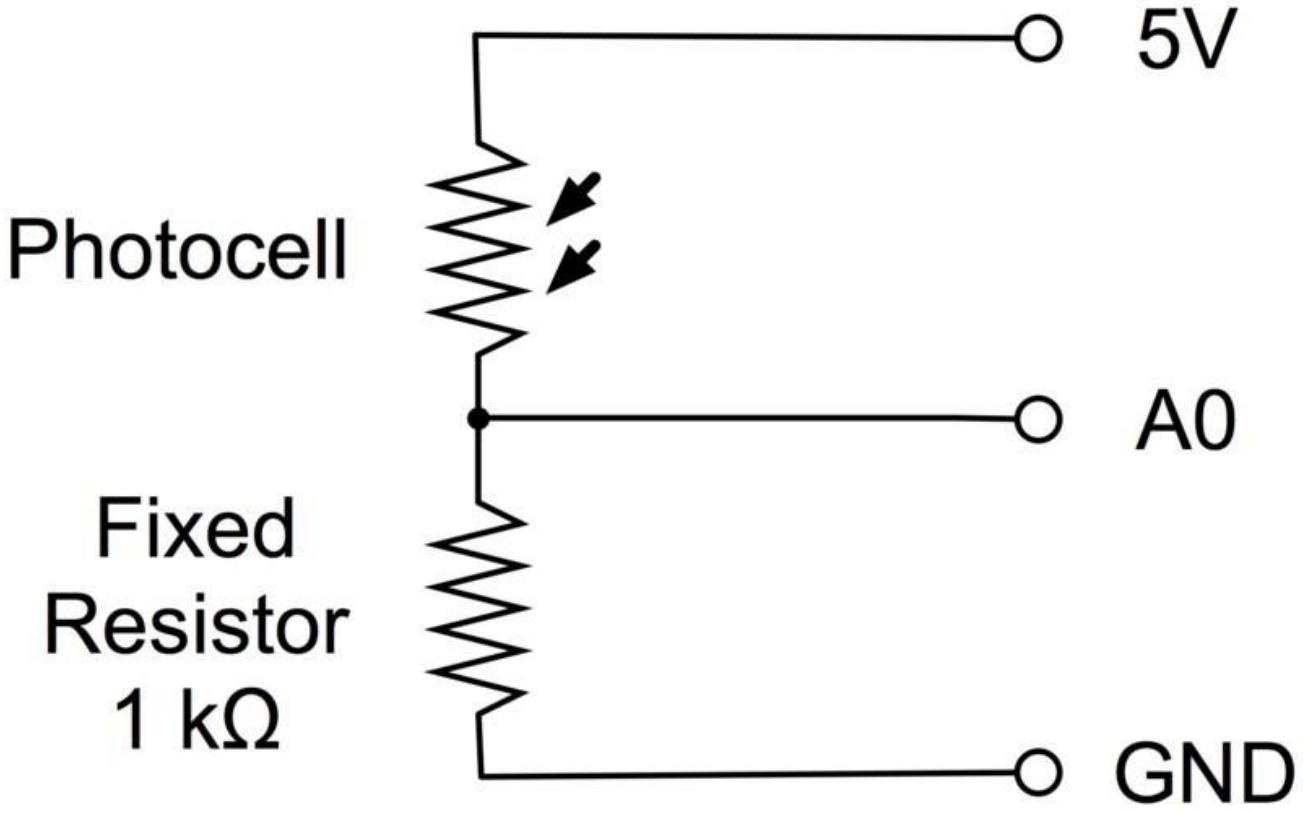
必要な構成部品:

(1) x Elegoo Uno R3
(1) x 830 tie-points breadboard
(8) x leds
(8) x 220 ohm resistors
(1) x 1k ohm resistor
(1) x 74hc595 IC
(1) x Photoresistor (Photocell)
(16) x M-M wires (Male to Male jumper wires)

部品の紹介

PHOTOCELL:

使用される光電池は、光依存性抵抗器と呼ばれるタイプのもので、LDR と呼ばれることもあります。 名前が示すように、これらのコンポーネントは抵抗器のように機能しますが、どれだけの光がそれらに当たるかに応じて抵抗が変化します。
これは、暗闇では約 50kΩ、明るい光では 500Ωの抵抗を持っています。 この抵抗値の変化を UNO R3 ボードのアナログ入力で測定できるものに変換するには、電圧に変換する必要があります。
これを行う最も簡単な方法は、固定抵抗と組み合わせることです。



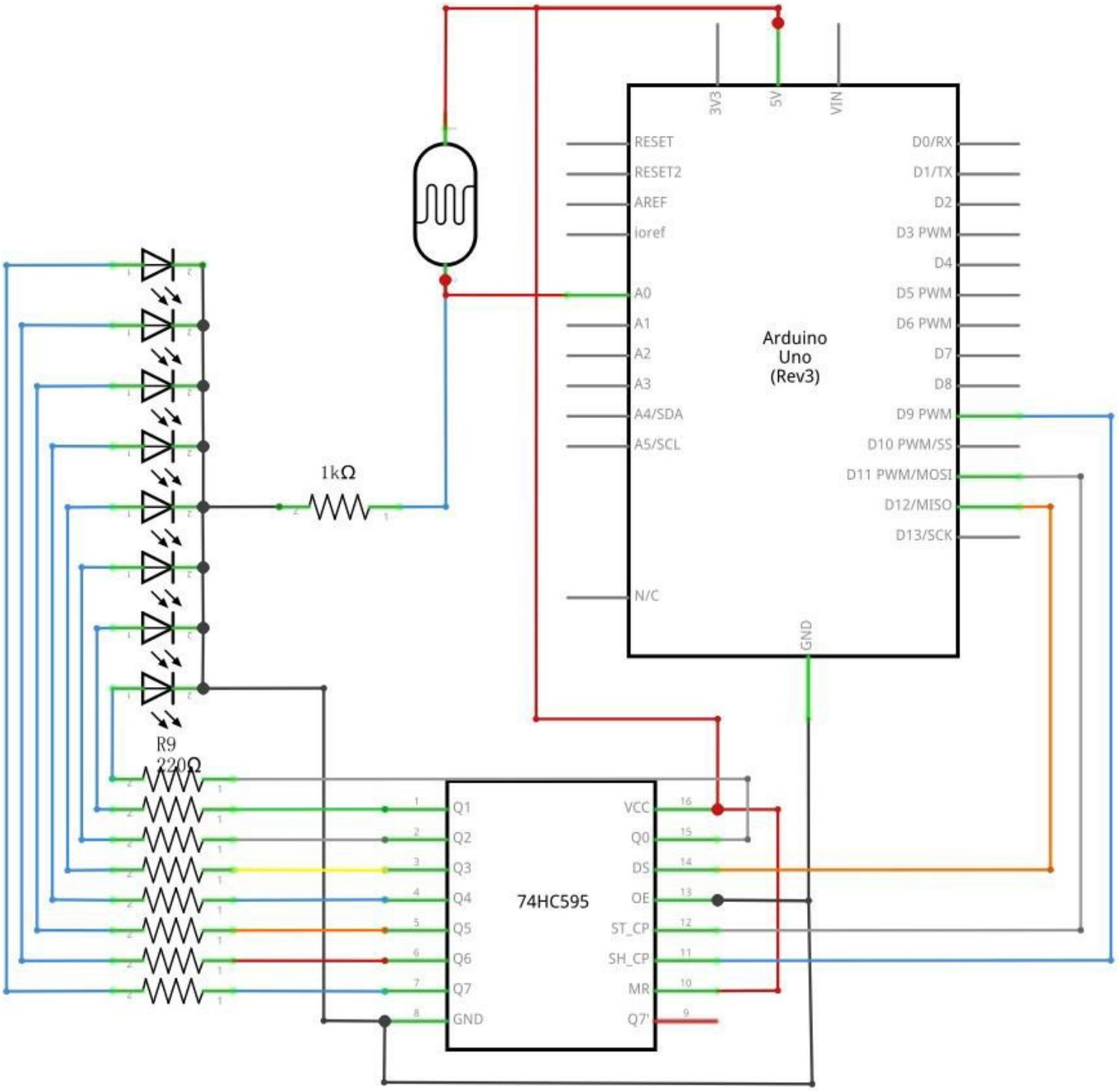
抵抗と光電池は一緒にポットのように振る舞います。 光が非常に明るい場合、光電池の抵抗は固定値の抵抗と比較して非常に低く、ポットが最大になったかのようです。

フォトセルが鈍い光の中にあるとき、抵抗は固定された 1kΩの抵抗よりも大きくなり、ポットが GND に向かって回転しているかのようになります。

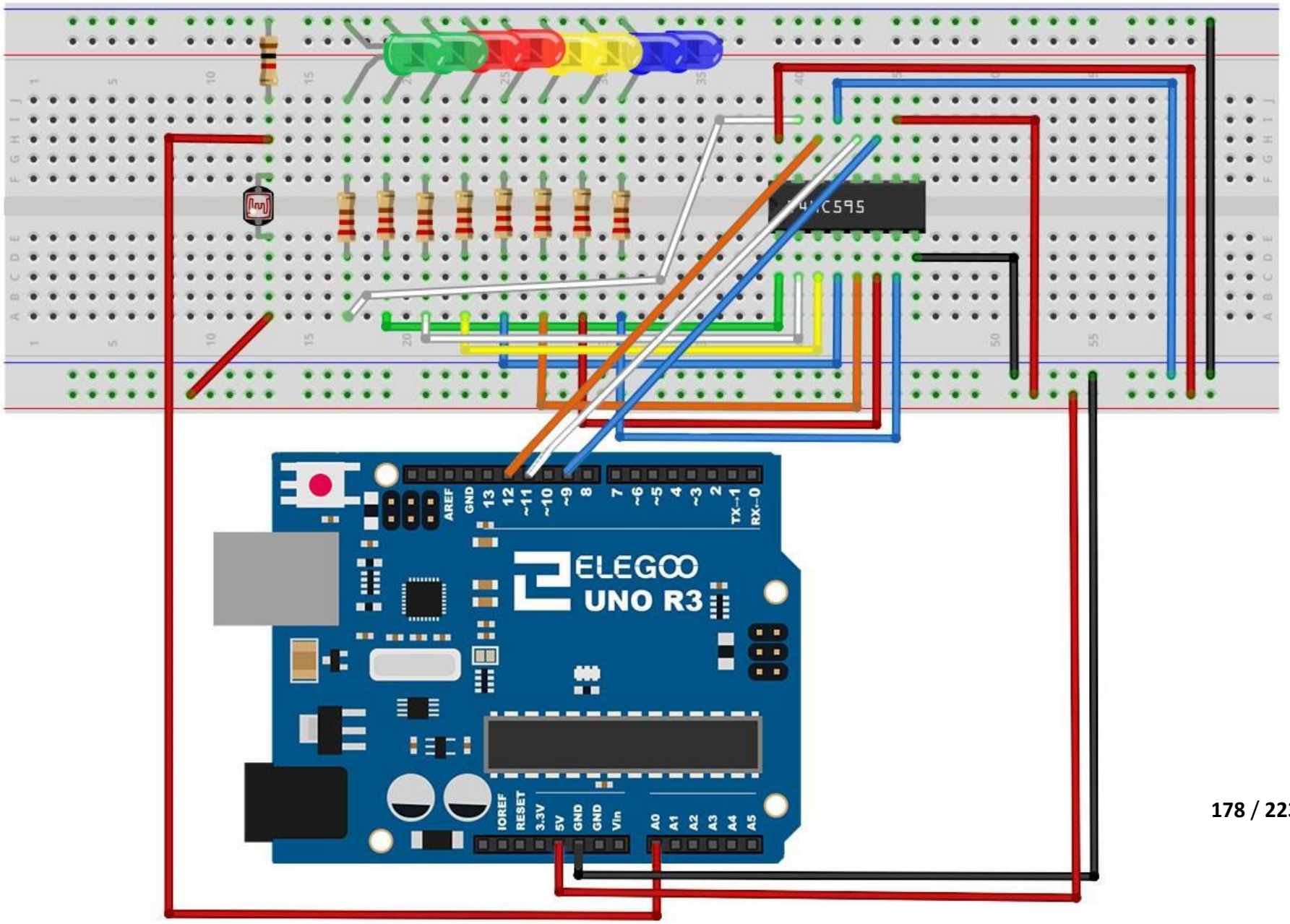
次のセクションにあるスケッチをロードして、光電管を指で覆い、光源の近くに保持してみてください。

Connection

Schematic



Wiring diagram



Code

配線後、プログラムをコードフォルダ - レッスン 26 フォトセルで開き、アップロードをクリックしてプログラムをアップロードしてください。 エラーがある場合のプログラムアップロードの詳細については、レッスン 2 を参照してください。

最初に留意すべき点は、アナログピンの名前を potPin ではなく「lightPin」に変更したことです。ポットが接続されていないためです。

スケッチに対する唯一の他の実質的な変更は、点灯する LED の数を計算する線です:

```
int numLEDSLit = reading / 57;        // all LEDs lit at 1k
```

今回は生の読みを 114 ではなく 57 で割っています。換言すると、それをポットで半分に分けて LED を 9 つのゾーンに分割します。 この余分な要因は固定 1kΩの抵抗を考慮することです。 これは、光電池の抵抗が 1kΩ（固定抵抗と同じ）である場合、生の読み取り値は 1023/2 = 511 になります。これは点灯しているすべての LED と同じになり、ビット（numLEDSLit）が 8。