Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

Lesson 1

An Introduction To Arduino

DFROBOT
DRIVE THE FUTURE

01

What is Arduino ?

Arduino is an open-source hardware and software platform designed for computer programmers, industrial artists, professionals and those interested in developing interactive devices and applications specific to an interactive development environment.

Arduino can receive input signals from various sensors and inputs. By controlling light sources, motors, or other actuators, Arduino can change the surrounding environment. Programs for the microcontroller on the Arduino board are written in Arduino's programming language (based on "Wiring" - an open source framework for microcontrollers) and run in the Arduino development environment (based on "Processing" - an open source programming language and integrated development environment).
Arduino is able to run independently or communicate with software running on a computer (for instance, Flash, Processing and MaxMSP). The open-source Arduino IDE, which is free to download, makes it easy for you to write code, upload it to the board and come up with your own interactive devices.

What can you do with Arduino?:

- ◆ Make a line-following robot
- ◆ Make a fluffy toy that lights up
- ◆ Make your phone ring when you receive an e-mail
- ◆ Make a Metroid-style arm cannon
- ◆ Make a coffee maker that sounds an alarm when your coffee is ready
- ◆ Make device that can record your heart rate when you ride your bike

All of these things can be achieved with Arduino!

The Birth of Arduino

Arduino started in a small picturesque town in northern Italy as as a project for students at the Interaction Design Institute Ivrea. Members of Arduino’s core development team were Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, David Mellis and Nicholas Zambetti.

Massimo Banzi’s students often complained that they couldn’t find a cheap and easy-to-use microcontroller. In the winter of 2005, Banzi mentioned this issue while talking with David Cuartielles, a Spanish chip engineer who was a visiting scholar to Massimo’s university. They decided to design their own circuit board and brought Banzi’s student David Mellis into this project to design the programming language for their board. Mellis finished the source code within two days. It took them another three days to etch the circuit board. They named it Arduino.

Now, anyone is able to make magic with Arduino, even without knowledge of computer programming. With Arduino, you can make a device respond to its0 environment, make flashing light shows

Why “Arduino”?

Ivrea is famous for the story of an oppressed king: In 1002 AD, King Arduino took the crown of Italy; however, in 1004 AD, he was dethroned by King Henry II of Germany.

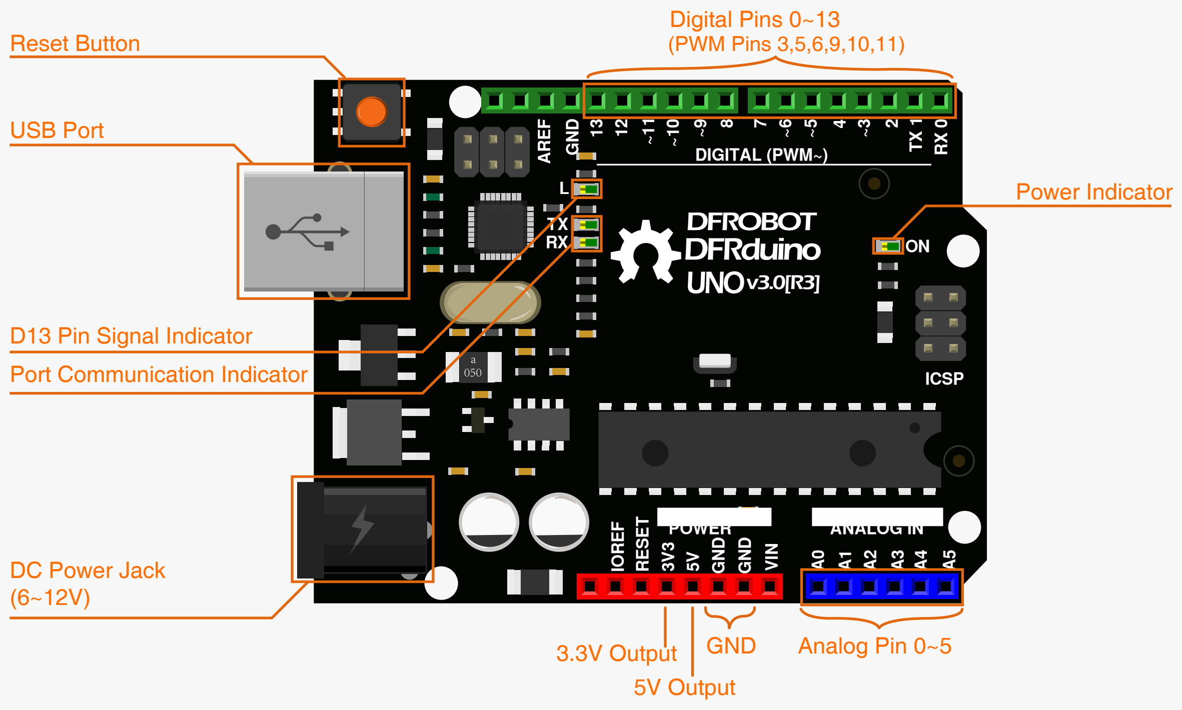
There is now a bar named “de Re Arduino” on Cobblestone Street in Ivrea which was opened in commemoration of King Arduino. Massimo Banzi was a regular customer there, and he fondly named his open-source hardware platform after it.

Introduction to DFRduino

In these lessons we will use DFRobot’s own clone of an Arduino Uno board: “DFRduino Uno”.DFRduino operates in exactly the same way as an Arduino Uno. To save confusion with names, let’s just call it the microcontroller. Let’s examine its features:

Sections with annotations below are parts that will be regularly used. Digital pins and analog pins marked on the diagram are what we call I/O (input/output). Digital pins are numbered from 0 to 13. Analog pins are numbered from 0 to 5.

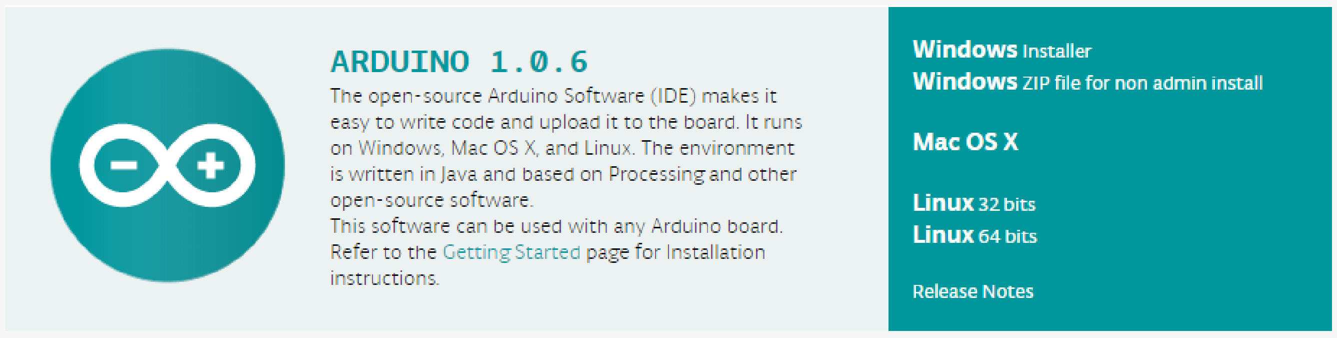
Voltage supply is also shown on the diagram below. DFRduino can be powered either by a USB connection or through a 6~12V DC supply via the barrel jack connector. Also integrated in to the board are four LED lights and a reset button. The LED light marked with “ON” is the power indicator, which will be on once power is connected. The LED light marked with ”L” is an indicator for digital pin 13 which will be discussed in the next section. TX (transmit) and RX (receive) are indicator lights for serial communication. When we upload a program, these two lights will blink rapidly, showing that data is being transmitted and received between the board and your computer.



First Use

1. Download Arduino IDE

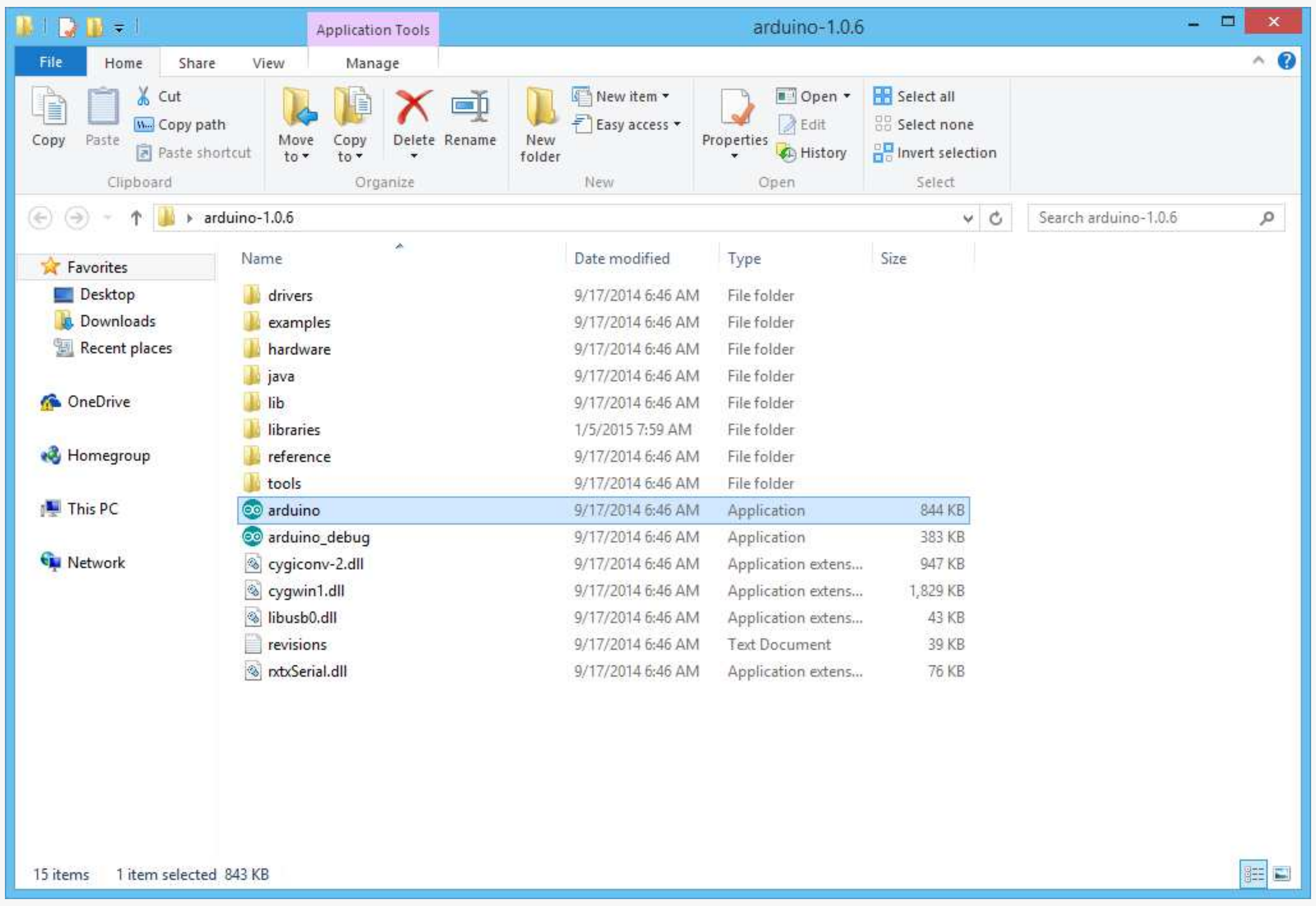
You can download Arduino IDE from this website:
http://www.arduino.org/software


ARDUINO 1.0.6
The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.
This software can be used with any Arduino board.
Refer to the Getting Started page for Installation instructions.

Windows Installer
Windows ZIP file for non admin install
Mac OS X
Linux 32 bits
Linux 64 bits
Release Notes

For Windows users, please click "Windows (ZIP file)". For Mac and Linux users, please select the corresponding link for your operating system.

Once downloaded, extract the files to a directory of your choice. Once extracted, open the directory. It should look like the image below:

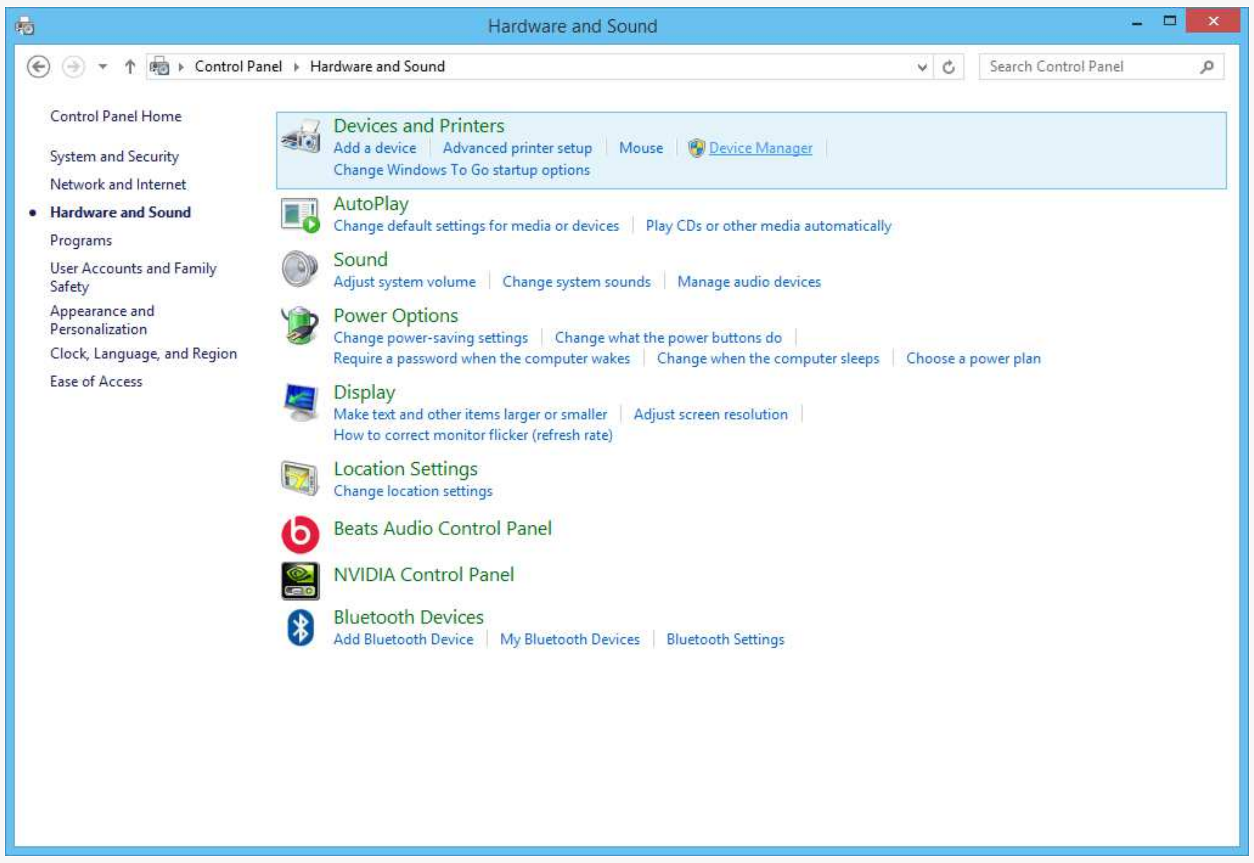


Name	Date modified	Type	Size
drivers	9/17/2014 6:46 AM	File folder	
examples	9/17/2014 6:46 AM	File folder	
hardware	9/17/2014 6:46 AM	File folder	
java	9/17/2014 6:46 AM	File folder	
lib	9/17/2014 6:46 AM	File folder	
libraries	1/5/2015 7:59 AM	File folder	
reference	9/17/2014 6:46 AM	File folder	
tools	9/17/2014 6:46 AM	File folder	
arduino	9/17/2014 6:46 AM	Application	844 KB
arduino_debug	9/17/2014 6:46 AM	Application	383 KB
cygiconv-2.dll	9/17/2014 6:46 AM	Application extens...	947 KB
cygwin1.dll	9/17/2014 6:46 AM	Application extens...	1,829 KB
libusb0.dll	9/17/2014 6:46 AM	Application extens...	43 KB
revisions	9/17/2014 6:46 AM	Text Document	39 KB
rxtxSerial.dll	9/17/2014 6:46 AM	Application extens...	76 KB

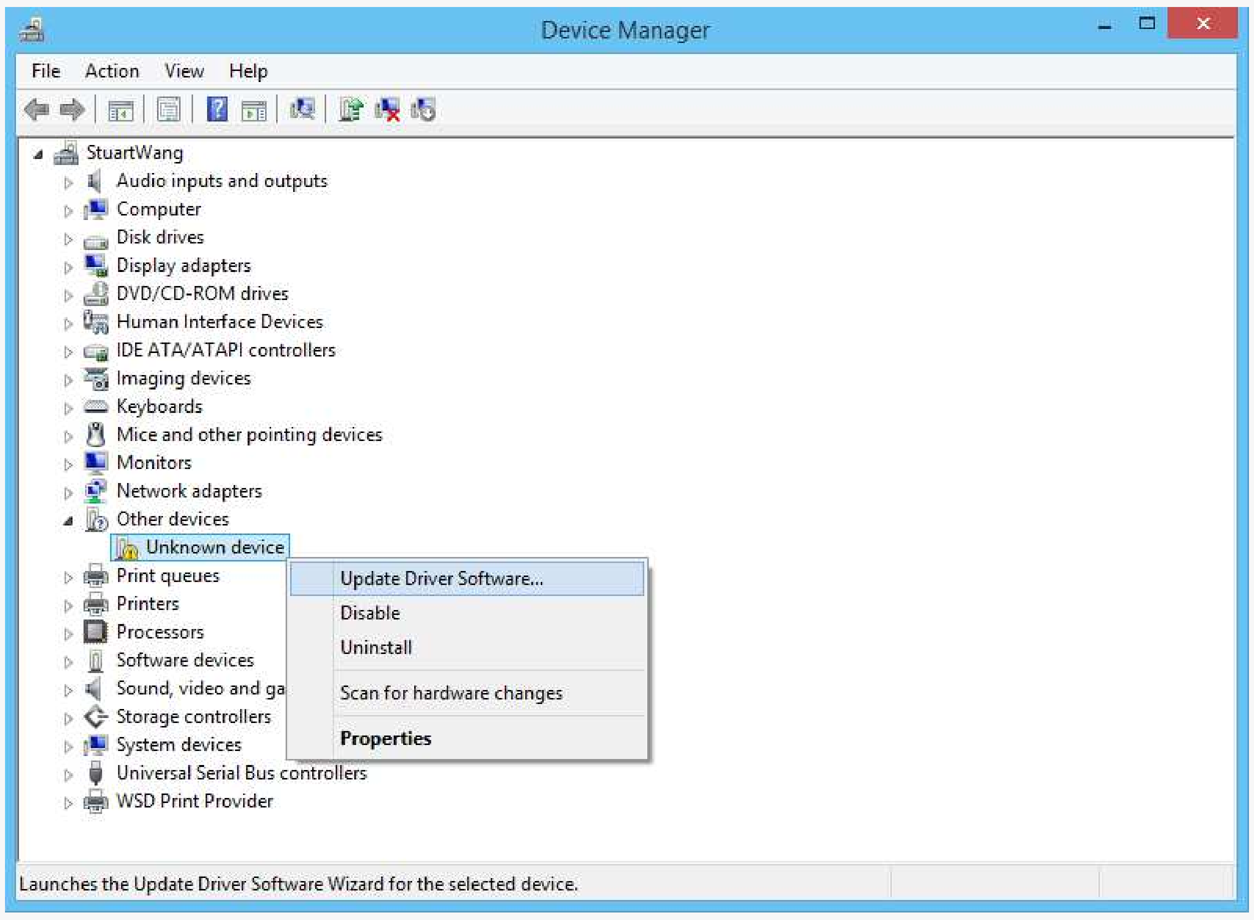
15 items 1 item selected 843 KB

2. Install Drivers

Connect the microcontroller to your computer using a USB cable. Once connected, the board's power indicator light will turn on.

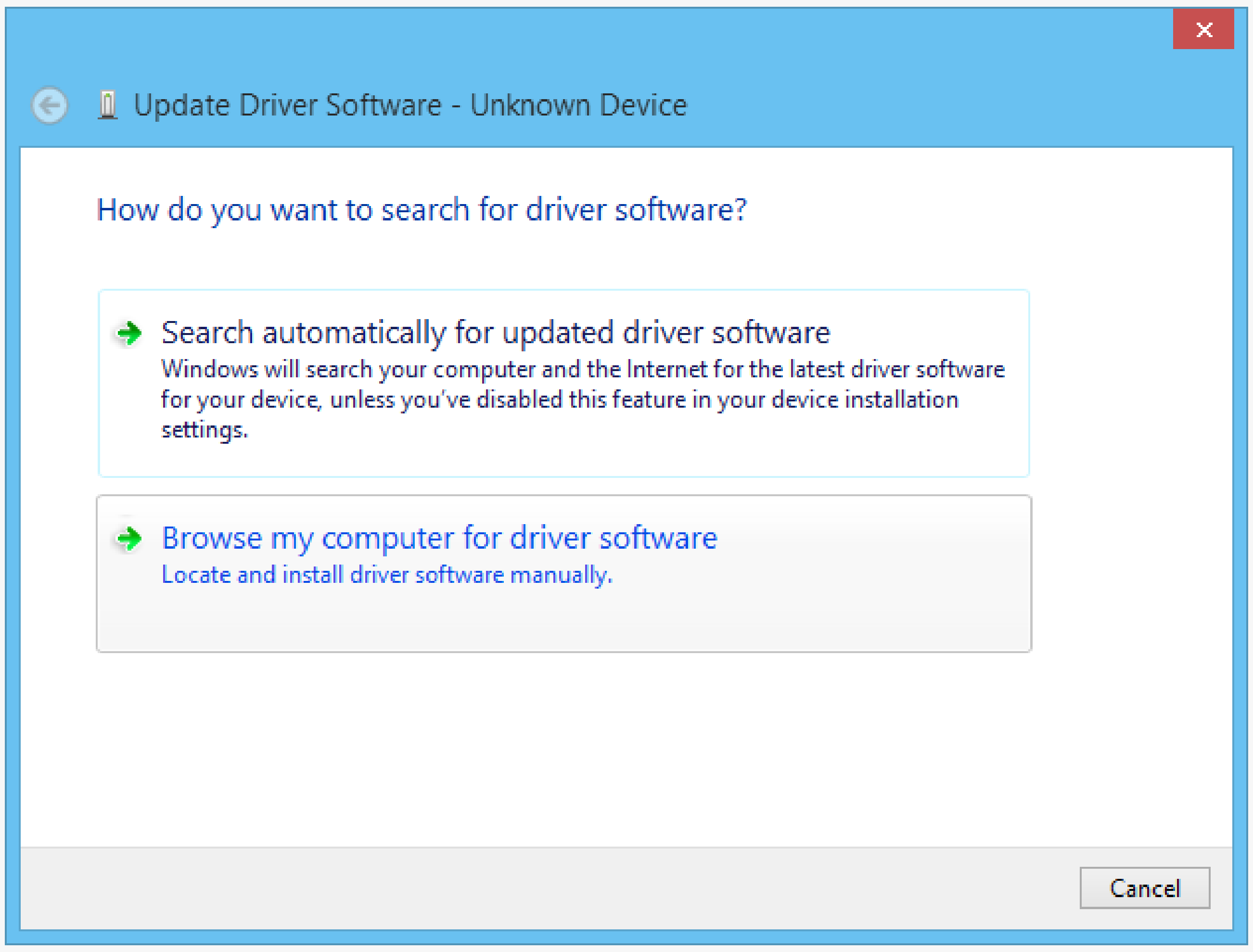

Hardware and Sound
Control Panel ▸ Hardware and Sound
Search Control Panel
Control Panel Home
System and Security
Network and Internet
Hardware and Sound
Programs
User Accounts and Family Safety
Appearance and Personalization
Clock, Language, and Region
Ease of Access
Devices and Printers
Add a device | Advanced printer setup | Mouse | Device Manager
Change Windows To Go startup options
AutoPlay
Change default settings for media or devices | Play CDs or other media automatically
Sound
Adjust system volume | Change system sounds | Manage audio devices
Power Options
Change power-saving settings | Change what the power buttons do
Require a password when the computer wakes | Change when the computer sleeps | Choose a power plan
Display
Make text and other items larger or smaller | Adjust screen resolution
How to correct monitor flicker (refresh rate)
Location Settings
Change location settings
Beats Audio Control Panel
NVIDIA Control Panel
Bluetooth Devices
Add Bluetooth Device | My Bluetooth Devices | Bluetooth Settings

Open "Control Panel" and navigate to "Device Manager" to set up the drivers.



Launches the Update Driver Software Wizard for the selected device.

You will then see a dialog box pop up. Select the second item “Manually Search for Drivers”. A browse window will appear that requires you to point it to the correct directory.

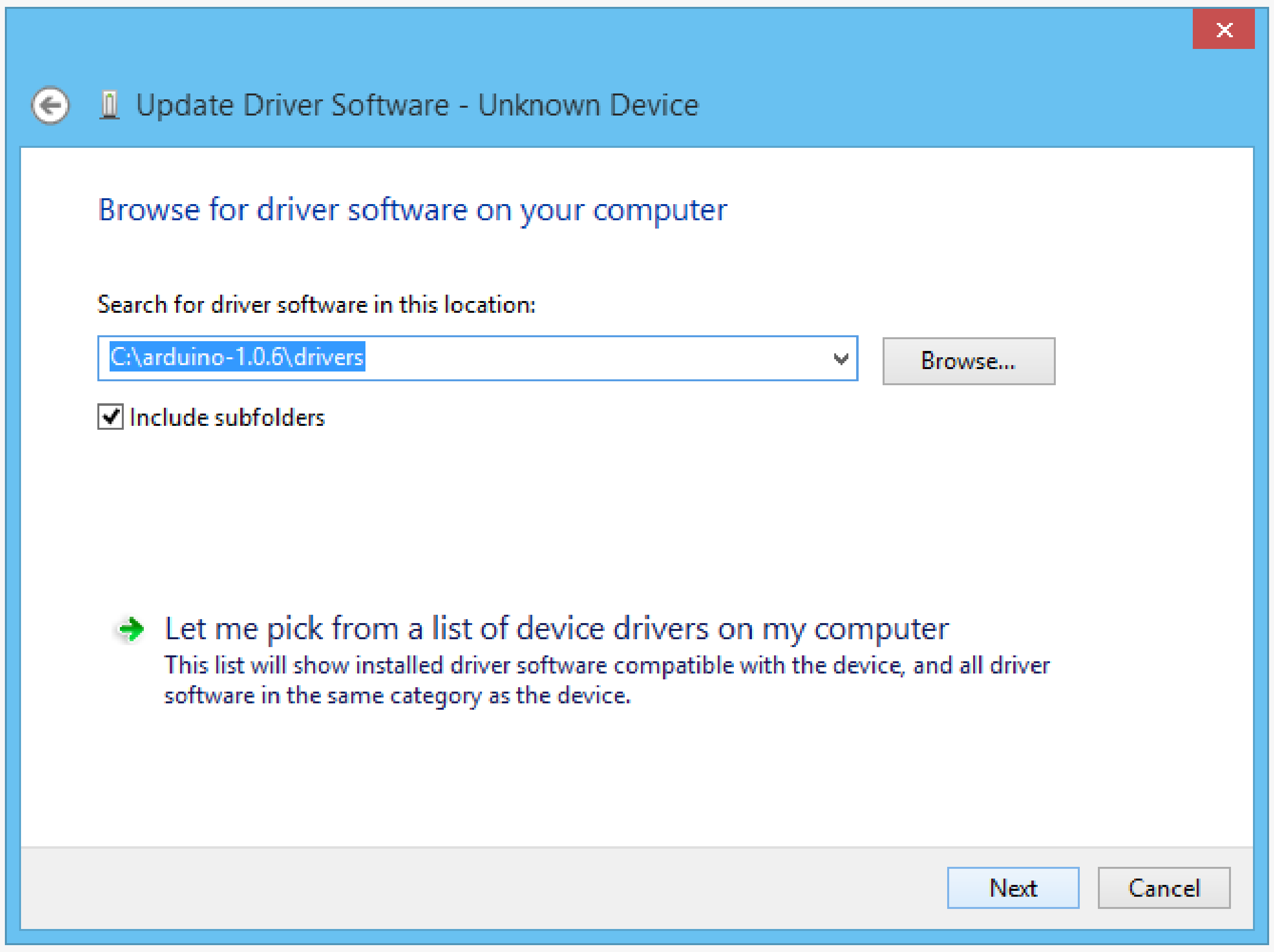

Update Driver Software - Unknown Device

How do you want to search for driver software?

Search automatically for updated driver software
Windows will search your computer and the Internet for the latest driver software for your device, unless you've disabled this feature in your device installation settings.

Browse my computer for driver software
Locate and install driver software manually.

Navigate the browse dialog to your Arduino directory. Inside this directory, there is a subfolder called “drivers” where the Arduino drivers are stored. Select this directory and then click “Next”.


Update Driver Software - Unknown Device

Browse for driver software on your computer

Search for driver software in this location:
C:\arduino-1.0.6\drivers

☑ Include subfolders

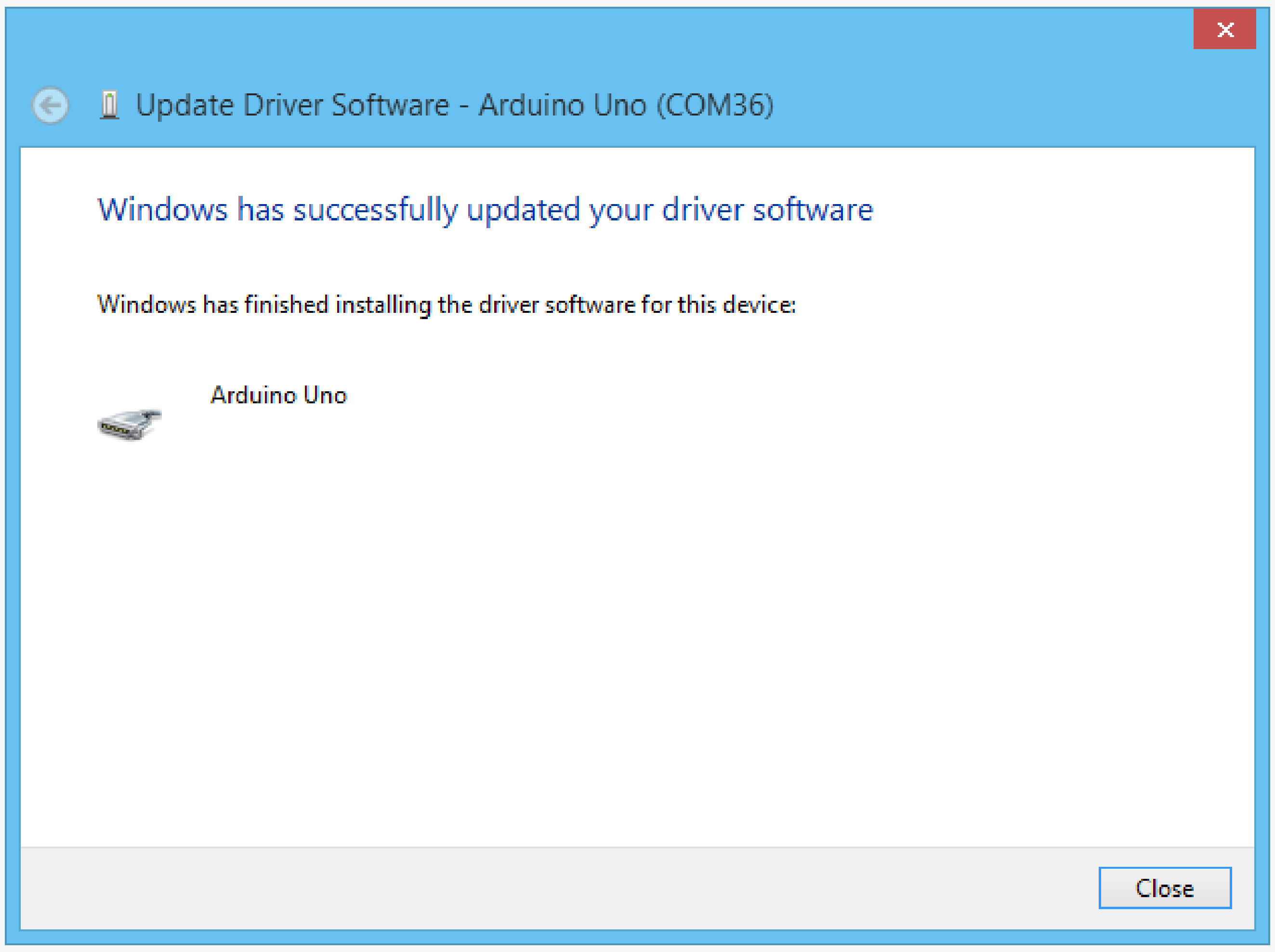
Let me pick from a list of device drivers on my computer
This list will show installed driver software compatible with the device, and all driver software in the same category as the device.

Hopefully this window will appear.

This shows that drivers have been successfully installed!

If you encounter issues, check this website for help:

https://www.arduino.cc/

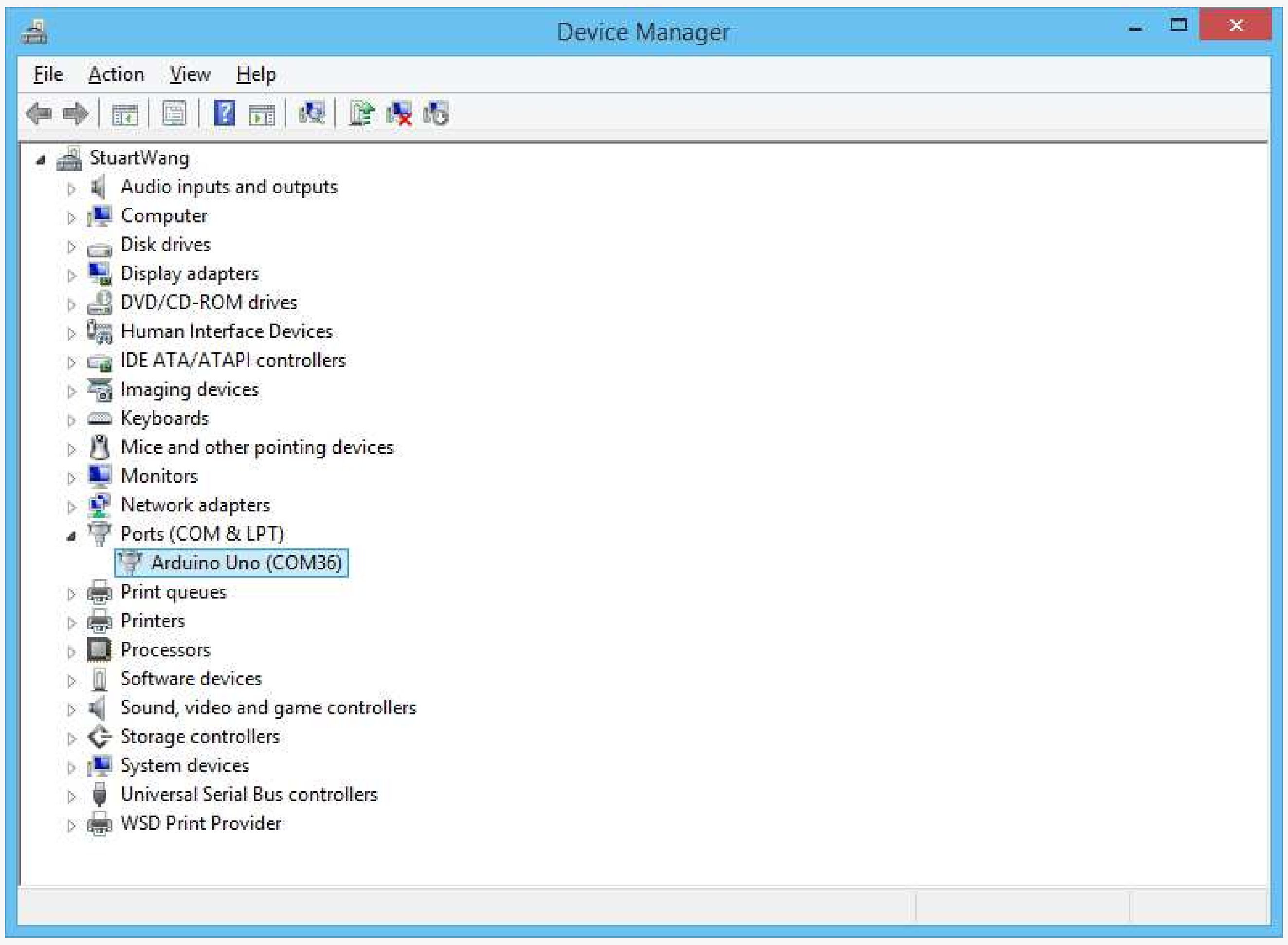


Windows has successfully updated your driver software

Windows has finished installing the driver software for this device:

Arduino Uno

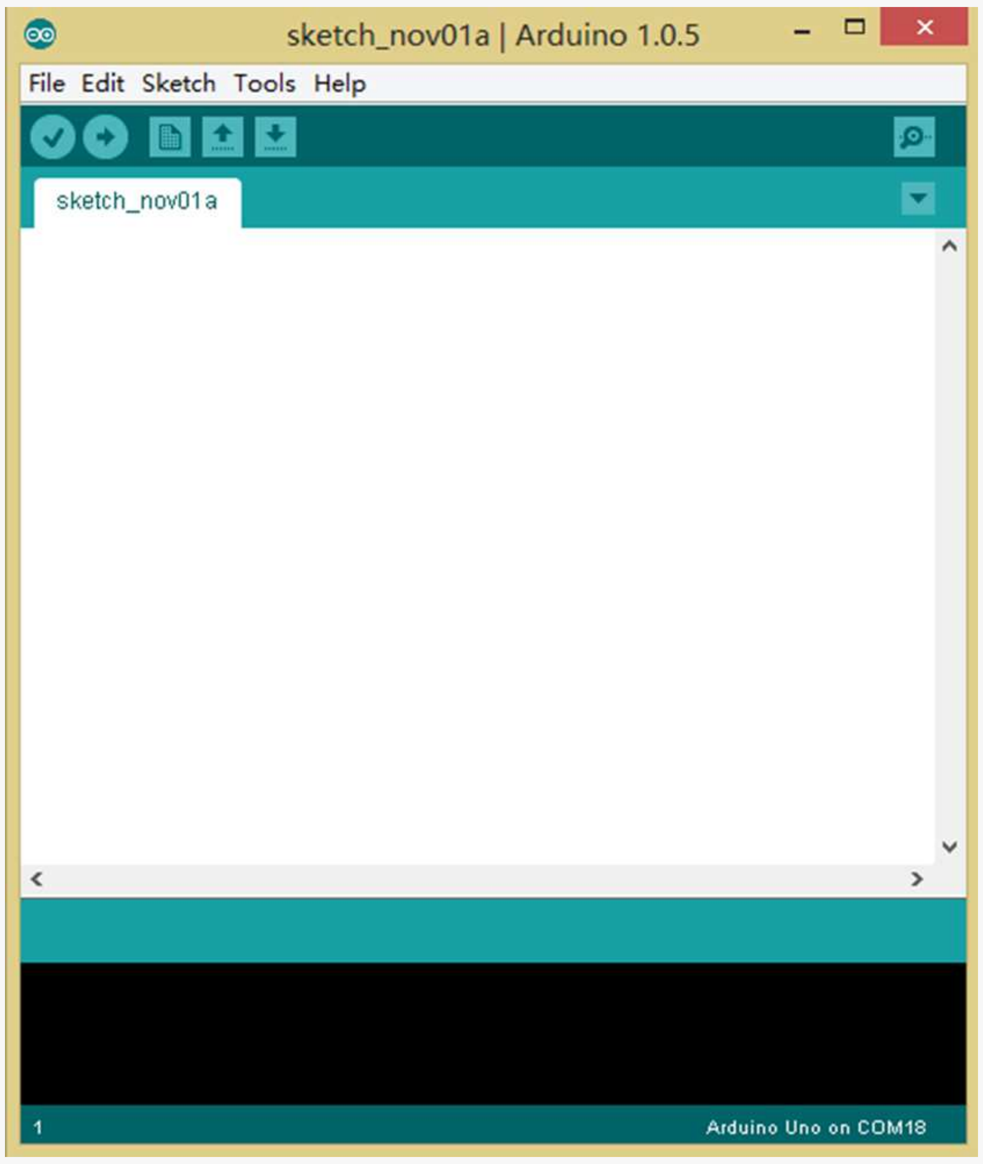
Return to “Device Manager”. The computer will have assigned a serial port to the microcontroller (your computer will show it as Arduino Uno). The serial port will vary depending on your computer, but should appear as “COM”, followed by a number.



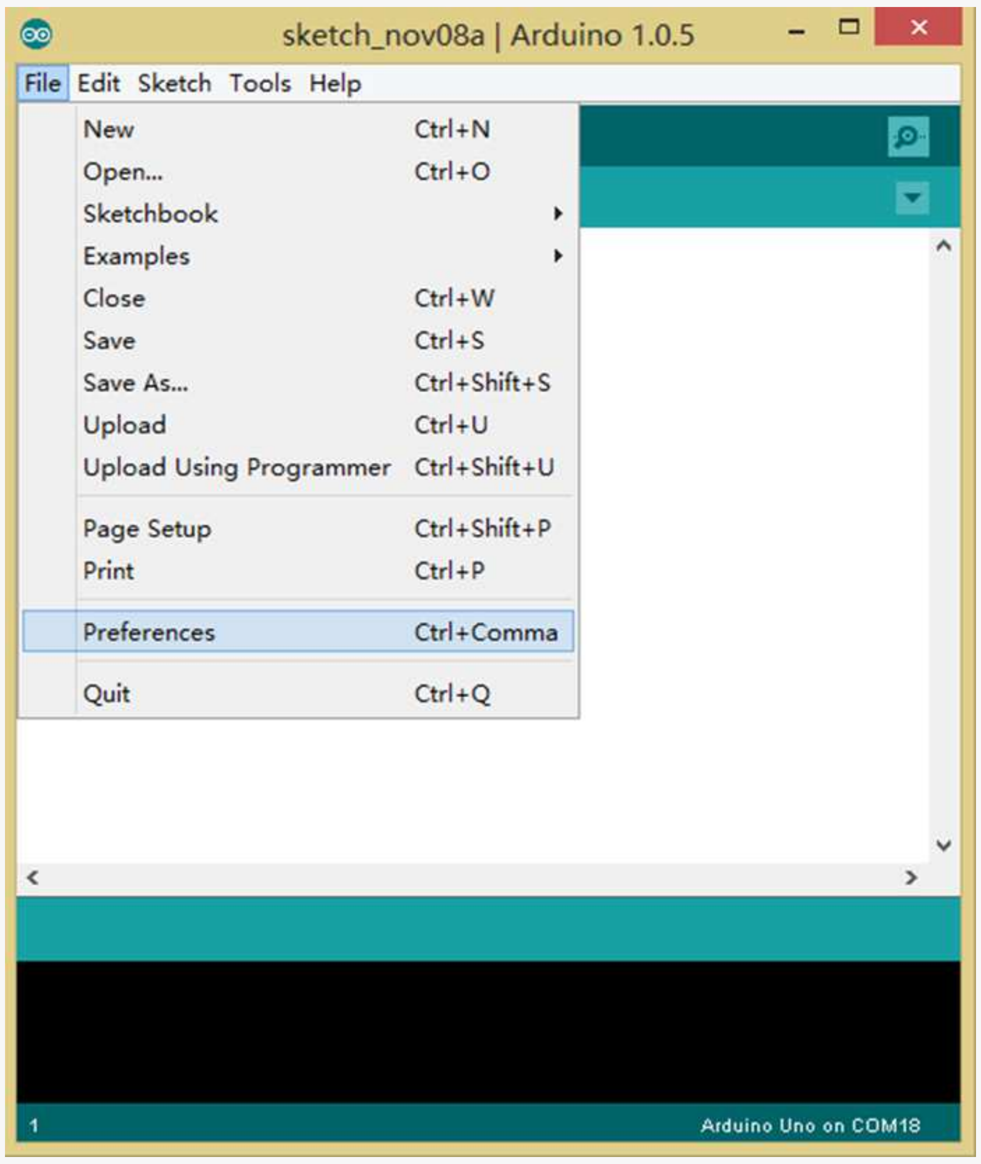
- StuartWang
  - Audio inputs and outputs
  - Computer
  - Disk drives
  - Display adapters
  - DVD/CD-ROM drives
  - Human Interface Devices
  - IDE ATA/ATAPI controllers
  - Imaging devices
  - Keyboards
  - Mice and other pointing devices
  - Monitors
  - Network adapters
  - Ports (COM & LPT)
    - Arduino Uno (COM36)
  - Print queues
  - Printers
  - Processors
  - Software devices
  - Sound, video and game controllers
  - Storage controllers
  - System devices
  - Universal Serial Bus controllers
  - WSD Print Provider

3.Introduction to Arduino IDE

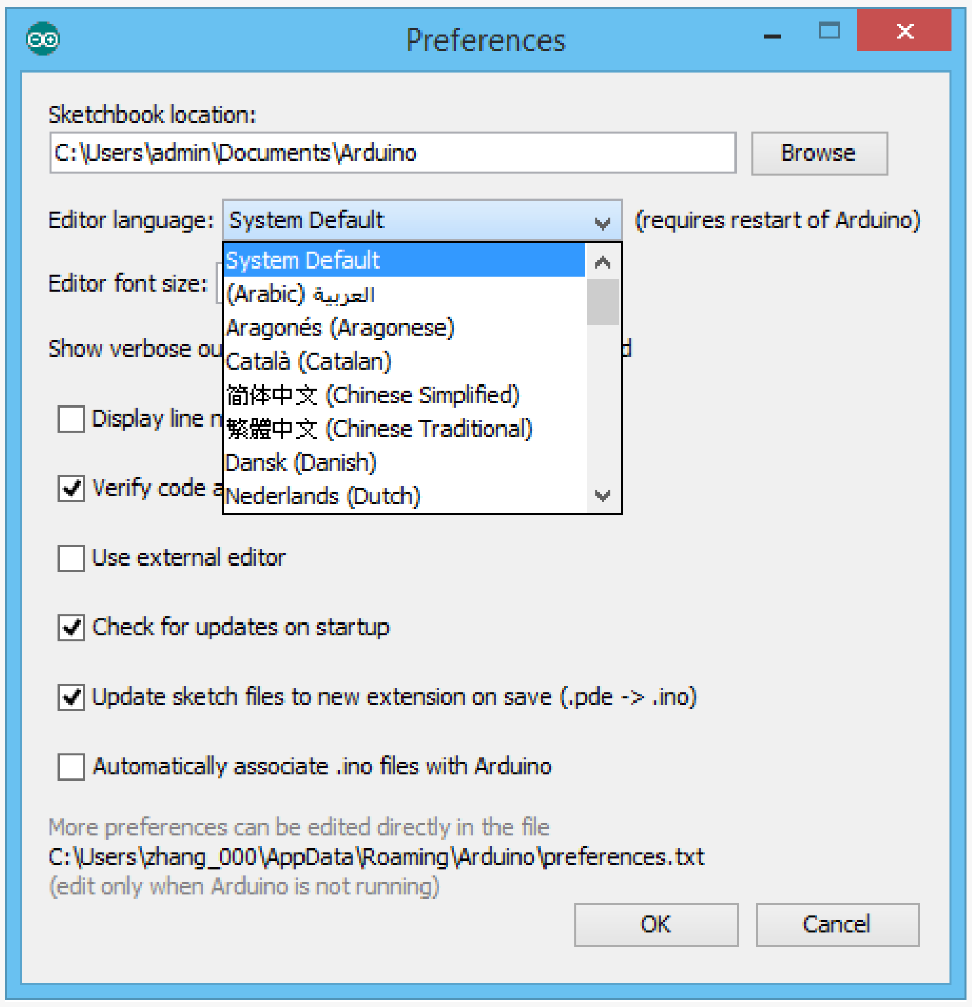
Inside your Arduino directory, open “Arduino.exe”. The application will open and the code editing interface will appear.



If you wish to change the language of the interface, select “File” and then “Preferences” to open preferences.



The dialog box shown below will pop up. Select “Editor Language”, choose your language and then click OK.



Sketchbook location:
C:\Users\admin\Documents\Arduino

Editor language: System Default (requires restart of Arduino)

System Default
(Arabic) العربية
Aragonés (Aragonese)
Català (Catalan)
简体中文 (Chinese Simplified)
繁體中文 (Chinese Traditional)
Dansk (Danish)
Nederlands (Dutch)

Editor font size:

Show verbose ou

☐ Display line n

☑ Verify code a

☐ Use external editor

☑ Check for updates on startup

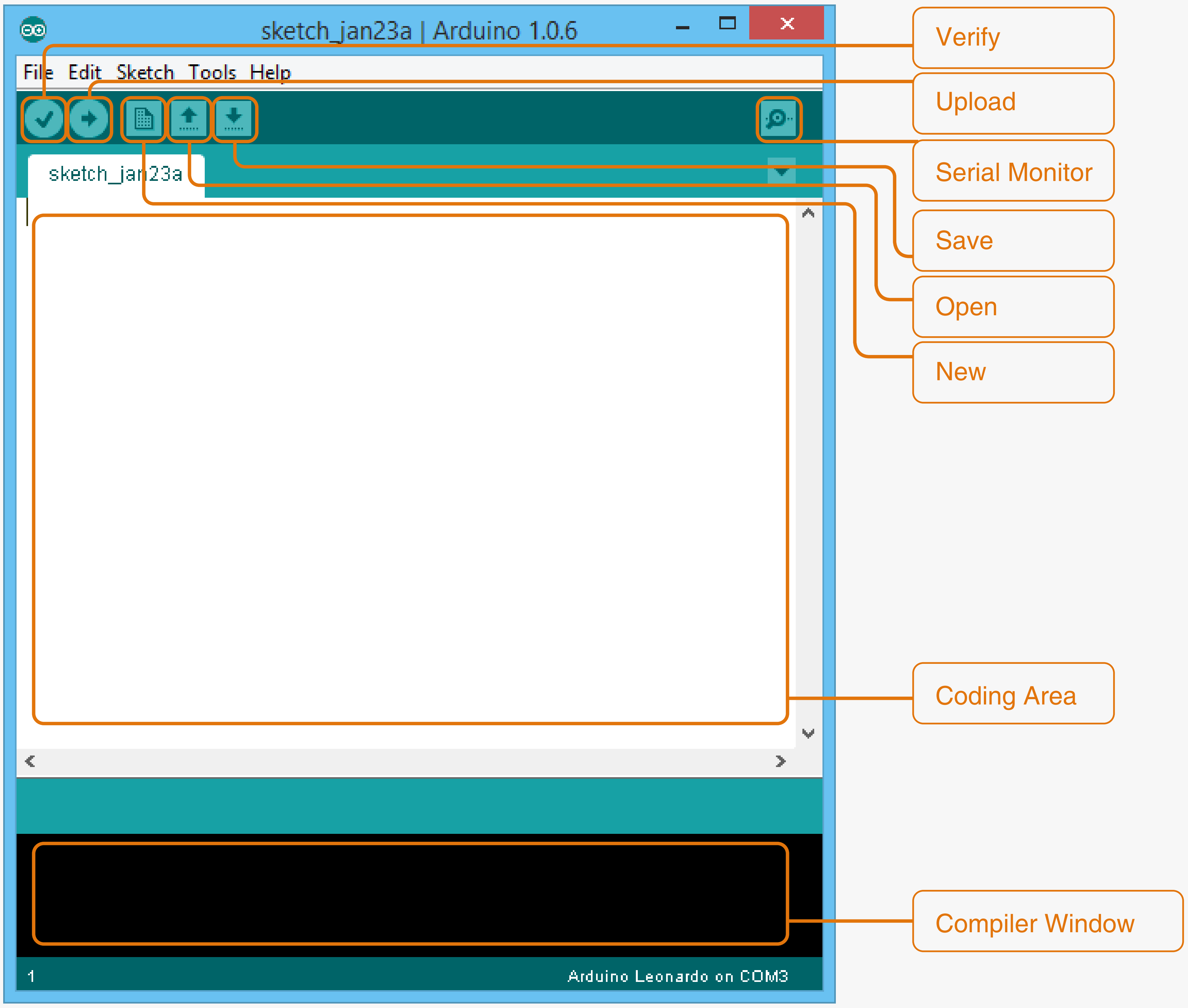
☑ Update sketch files to new extension on save (.pde -> .ino)

☐ Automatically associate .ino files with Arduino

More preferences can be edited directly in the file
C:\Users\zhang_000\AppData\Roaming\Arduino\preferences.txt
(edit only when Arduino is not running)

Close Arduino IDE and then reopen it for the changes to take effect.

We will use Arduino IDE a lot in these tutorials, so lets learn about its features:

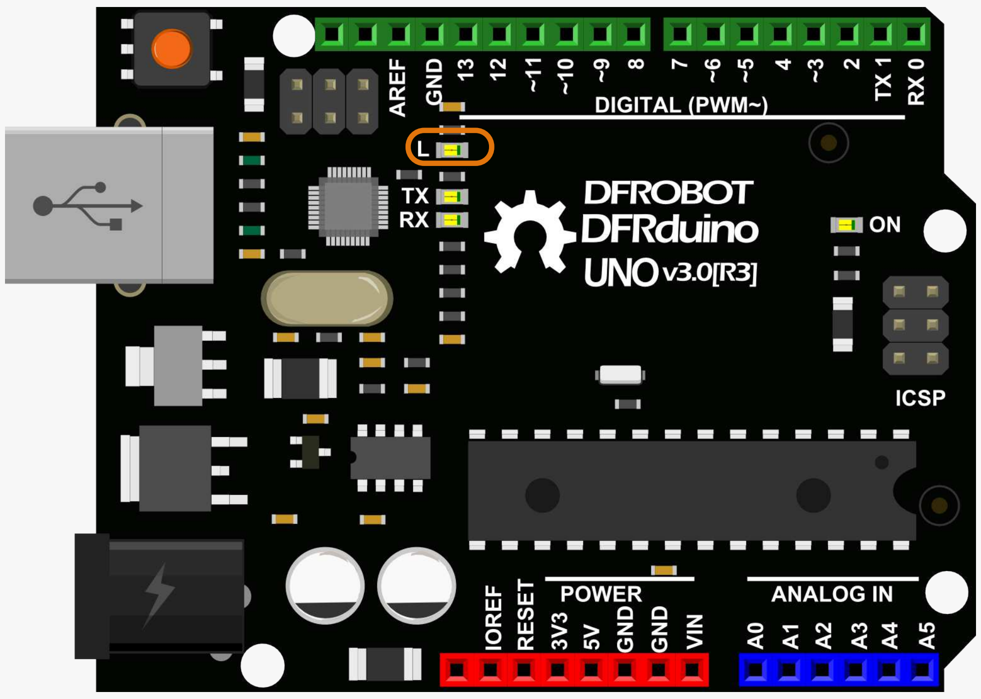


Arduino IDE allows you to edit and upload programs to your microcontroller. Arduino IDE calls programs “sketches”, but in these tutorials, we will simply say “program” or “code”, rather than “sketch”, but they really mean the same thing.

The interface shown in the picture above is where you’ll write and upload code within Arduino IDE. The main white area is where you input code. We use the C programming language to program the microcontroller, which we will discuss in more detail in later chapters. When you press verifyor upload, the code instructions you have written will be translated in to machine language by a piece of software called a compilerso that the microcontroller can understand it. This process is called “compiling” or “verifying”. Messages shown in the black area will show information about the compiling and upload process.

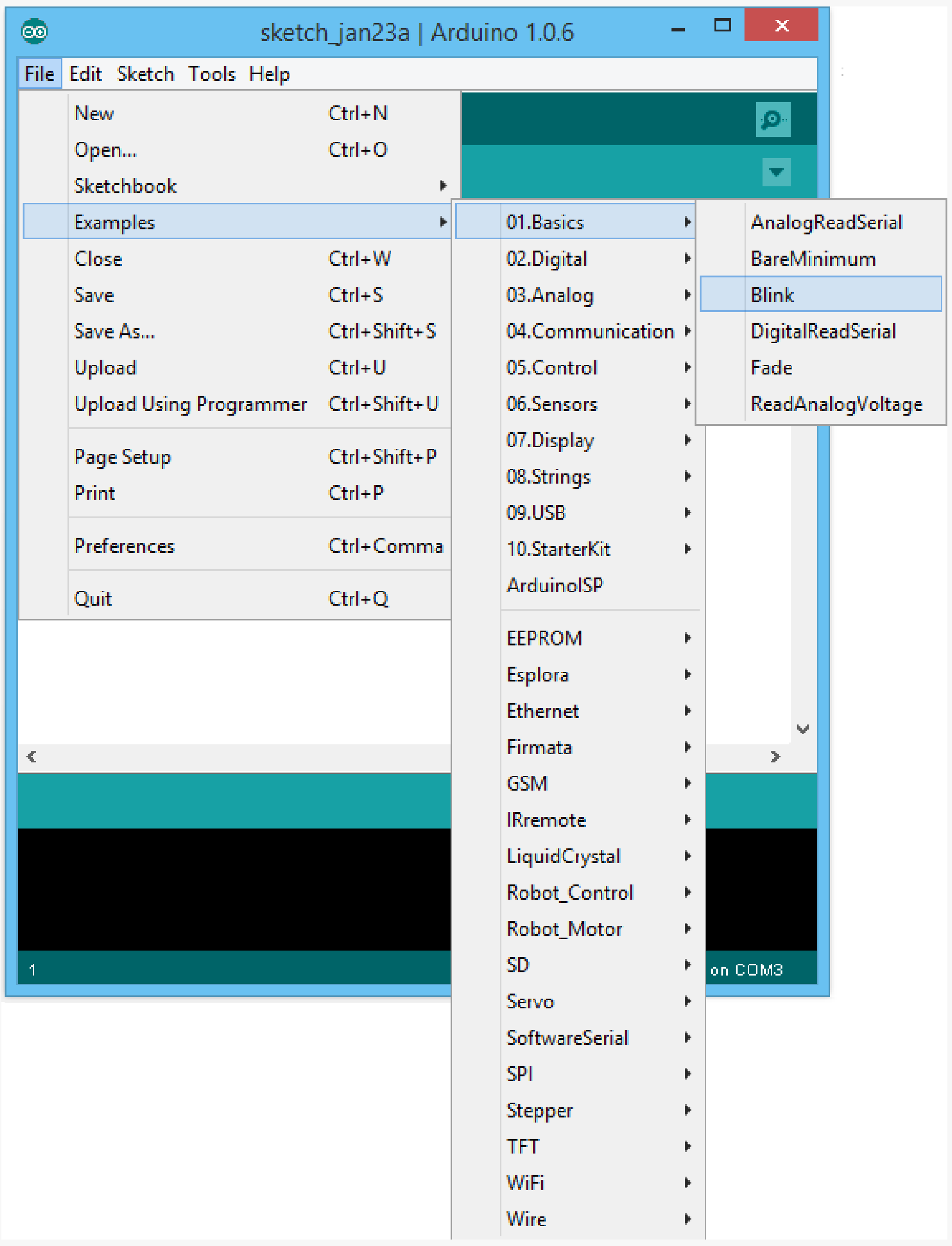
4. Upload a Blink Program

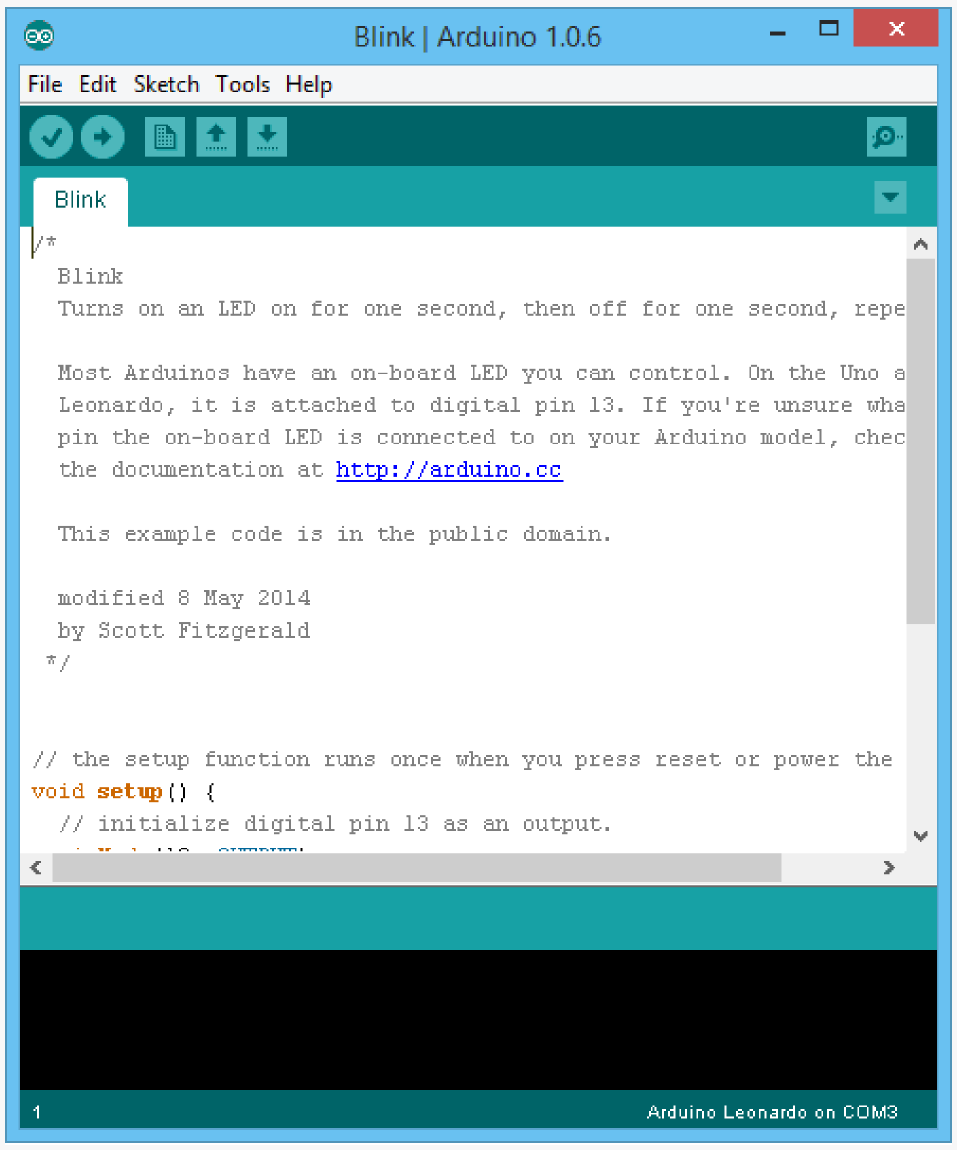
Let's upload a simple program to become familiar with the uploading process and to test the microcontroller. This code will turn the "L" LED light on the microcontroller on and off. The location of the "L" LED is circled in the diagram below.



Connect your computer to the microcontroller via the USB.

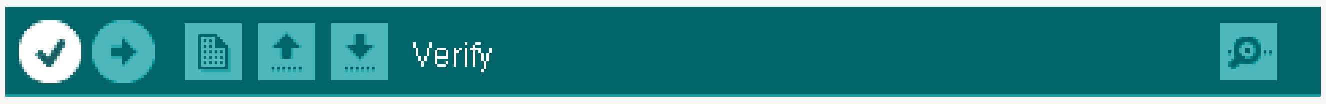
In the Arduino IDE, go to `File > Examples > 01. Basics > Blink`.



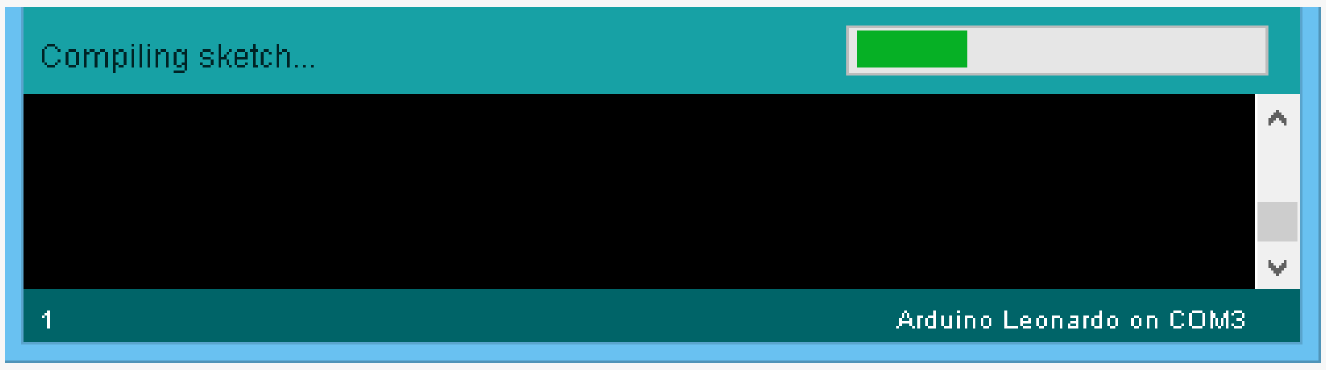


```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repe

  Most Arduinos have an on-board LED you can control. On the Uno a
  Leonardo, it is attached to digital pin 13. If you're unsure wha
  pin the on-board LED is connected to on your Arduino model, chec
  the documentation at http://arduino.cc

  This example code is in the public domain.

  modified 8 May 2014
  by Scott Fitzgerald
 */


// the setup function runs once when you press reset or power the
void setup() {
  // initialize digital pin 13 as an output.
```

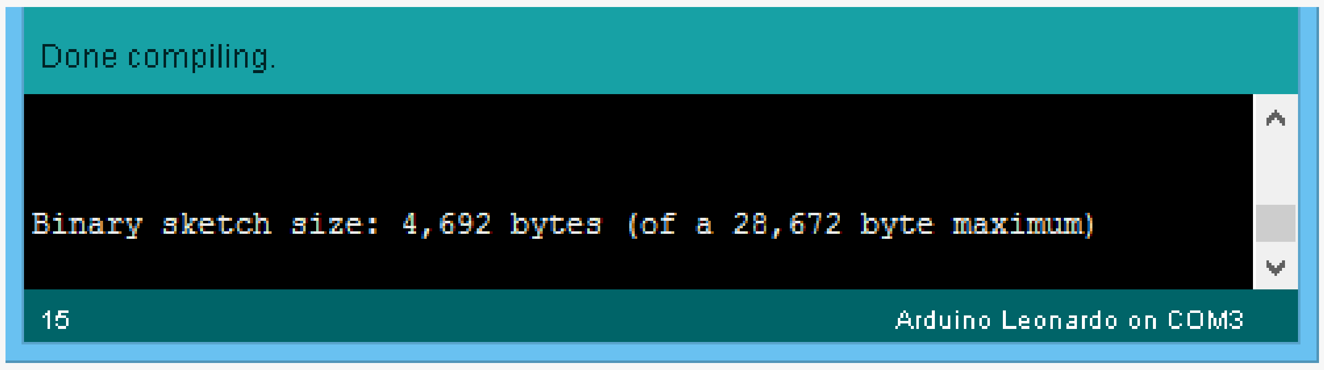
As this is an example program, there will be no syntax errors in the code. When we write code ourselves we can click “Verify” to check if there are any syntax errors. Try it now.



The code is run through the compiler. This green bar shows the compiling process.



Hopefully you will see this message:

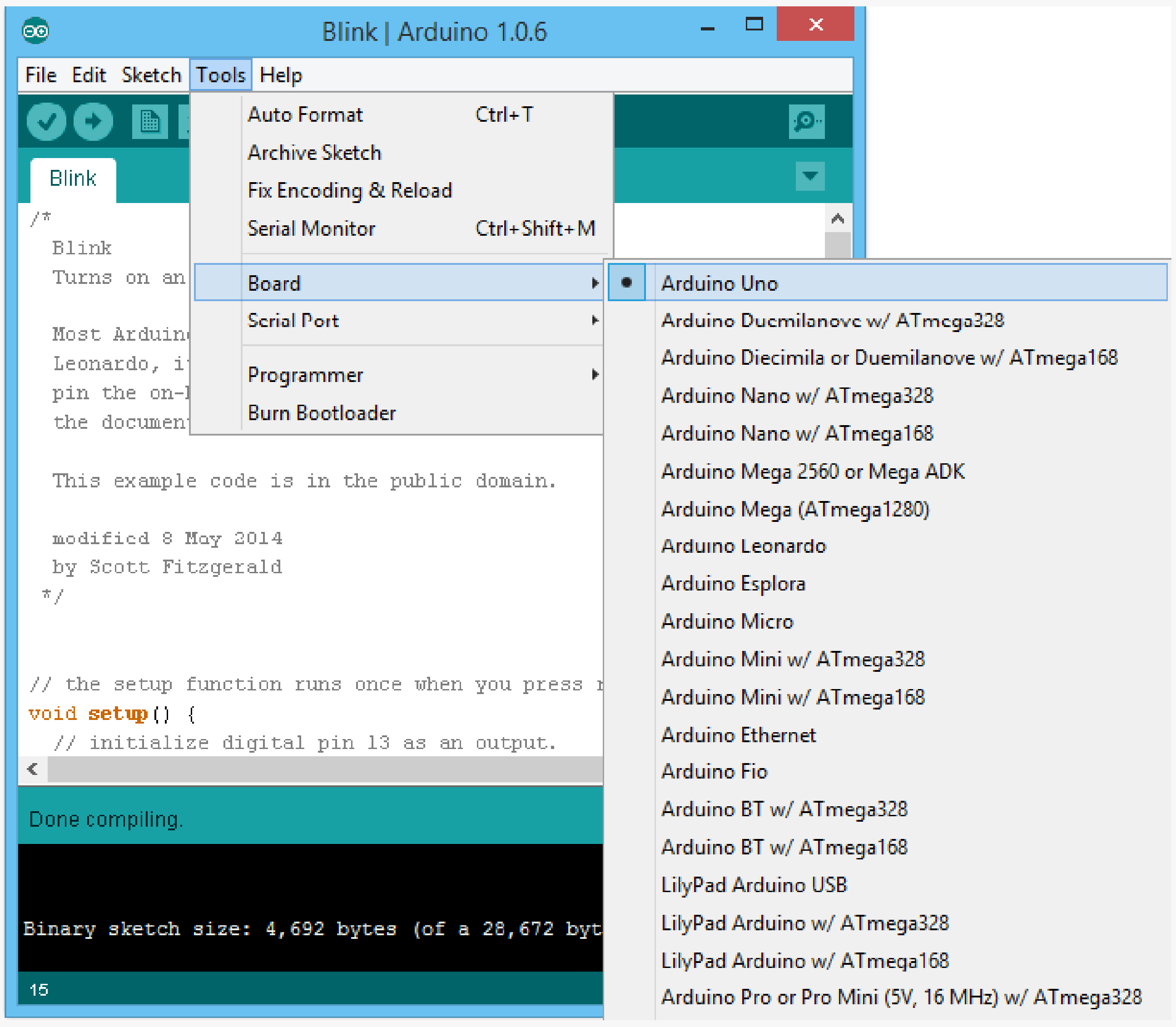


```
Binary sketch size: 4,692 bytes (of a 28,672 byte maximum)
```

Before code is uploaded to the microcontroller, the IDE will automatically verify it.
If there are any errors in your code, a big ugly orange message box will appear above the compiler window indicating a problem, and your code can not upload until it is fixed!

Before we can upload a program, we need to tell the computer where to send it. In Arduino IDE, we need to select which type of Arduino board we are using as well as the serial port that the board is connected to.

Navigate to `Tools > Board` and then select Arduino UNO from the list. That's it!

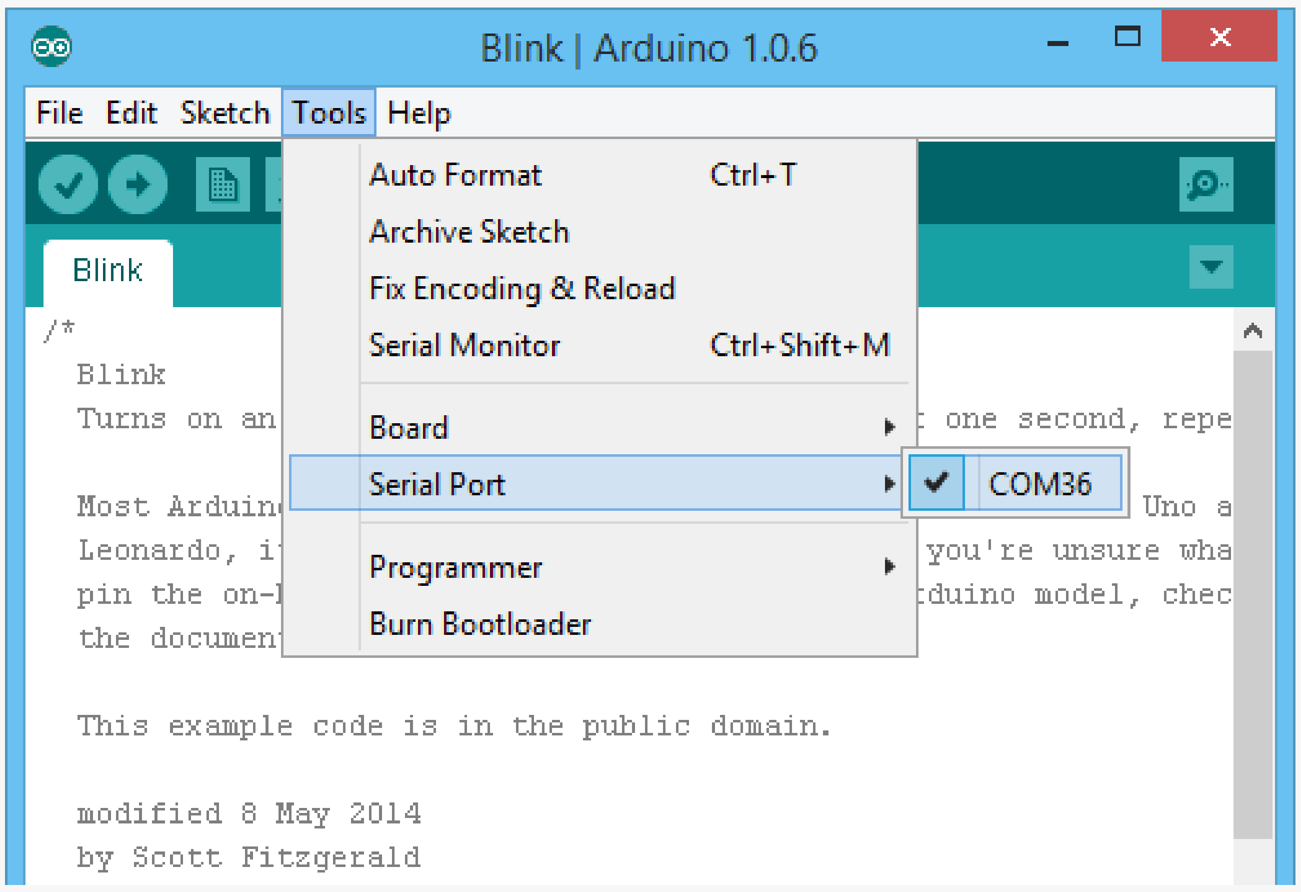

Blink | Arduino 1.0.6
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Board
Serial Port
Programmer
Burn Bootloader
Blink
● Arduino Uno
Arduino Duemilanove w/ ATmega328
Arduino Diecimila or Duemilanove w/ ATmega168
Arduino Nano w/ ATmega328
Arduino Nano w/ ATmega168
Arduino Mega 2560 or Mega ADK
Arduino Mega (ATmega1280)
Arduino Leonardo
Arduino Esplora
Arduino Micro
Arduino Mini w/ ATmega328
Arduino Mini w/ ATmega168
Arduino Ethernet
Arduino Fio
Arduino BT w/ ATmega328
Arduino BT w/ ATmega168
LilyPad Arduino USB
LilyPad Arduino w/ ATmega328
LilyPad Arduino w/ ATmega168
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328

```
/*
  Blink
  Turns on an

  Most Arduin
  Leonardo, i
  pin the on-b
  the documen

  This example code is in the public domain.

  modified 8 May 2014
  by Scott Fitzgerald
 */


// the setup function runs once when you press r
void setup() {
  // initialize digital pin 13 as an output.
```

```
Done compiling.

Binary sketch size: 4,692 bytes (of a 28,672 byt
15
```

(There are many different types of Arduino board of all shapes and sizes. Some have more RAM and more I/O pins and can hold larger programs. We are using an Arduino Uno variant, so we must select Arduino Uno from the boards list.)

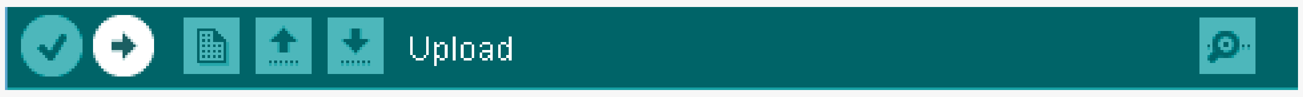
Now select the serial (COM) port. Navigate to `Tools > Port` and your available serial ports will be listed. Usually this will be "COM" followed by a number. Select the available port.
In our example the board is connected to the computer on COM36.



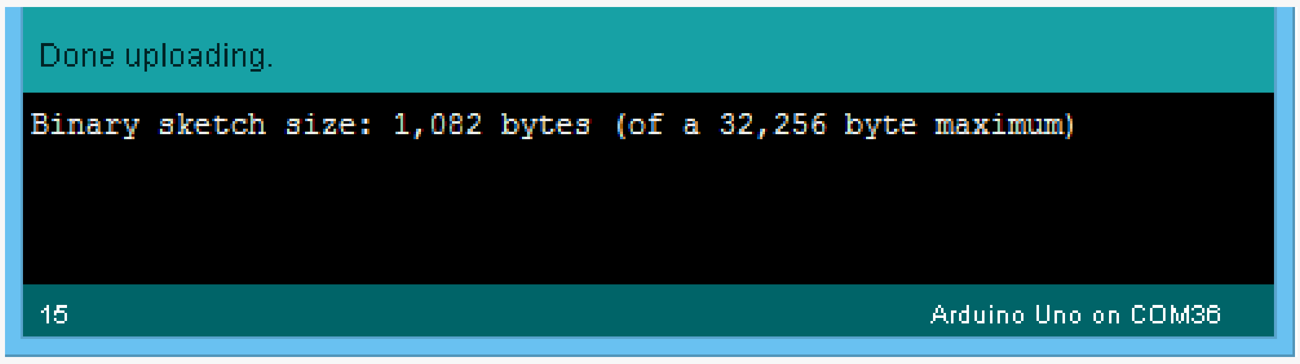
```
/*
  Blink
  Turns on an                              t one second, repe

  Most Arduin                              Uno a
  Leonardo, i                              you're unsure wha
  pin the on-l                             rduino model, chec
  the documen

  This example code is in the public domain.

  modified 8 May 2014
  by Scott Fitzgerald
```

We should now be able to upload code to the microcontroller.

Click upload in Arduino IDE. If all is well, the code will verify and then upload to the microcontroller.



When the upload has completed successfully, the microcontroller will automatically reset and the code will execute. Is the light blinking?



```
Done uploading.
Binary sketch size: 1,082 bytes (of a 32,256 byte maximum)
15                                         Arduino Uno on COM36
```

This concludes the blink tutorial.

To review, there are three fundamental steps to upload code to your microcontroller:

1. Verify code
2. Select board and COM port
3. Upload!

If you have any questions or comments about this tutorial, or any other tutorials in this guide, feel free to contact us on the DFRobot forum:
http://www.dfrobot.com/forum/

The DFRobot forum is a community where hackers and makers can share their exciting ideas. Come and visit to get ideas about what you can make with your DFRduino microcontroller, or just share what you've done following these tutorials. We love your feedback!

Lesson 2

How Does a Device “Think”?

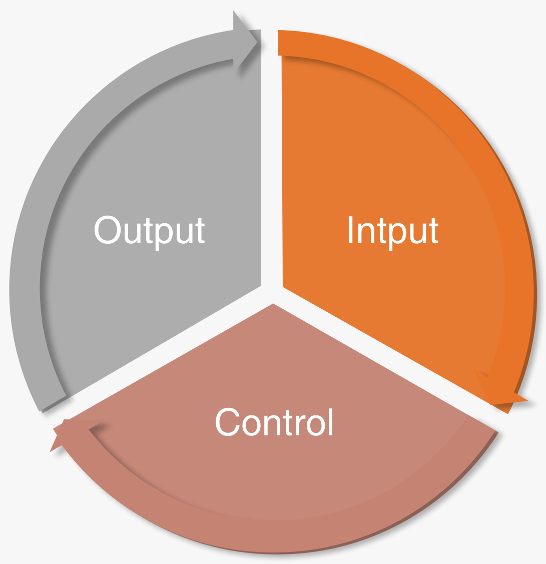
02

DFROBOT
DRIVE THE FUTURE

Simple Automatic Control Devices

Any device that includes a micro-controller can be considered a simple automatic control device. There are three essential constituent parts for a simple automatic control device:

- An **input** unit for collecting signals
- An **output** unit for sending out signals
- A **control** unit for processing the signals received (such as a microcontroller).



Let’s draw an analogy between an automatic control device and a person. In a person, signals are sensed through sight, hearing, taste, touch and smell, and then processed by the brain which then outputs responses to each. In this case, the output signal would be the actions of that person.

A microcontroller can output responses as well. Output responses include sound, light and movement (DC Motors, LEDs, servos etc.)

Here is another way to explain it: imagine someone says “hello” to you, and you immediately reply “hello” in return. In this case, your ear is the input unit; your brain is the control unit and your mouth is the output unit.

How can we carry out this whole process (receiving signals, processing signals and then sending out signals) using our micro-controller?

As well as our microcontroller, we simply need an acoustic sensor and a buzzer. The acoustic sensor “hears” a sound, the microcontroller receives a signal and then outputs a signal to the buzzer for it to buzz. In this case the acoustic sensor is the input unit; the microcontroller is the control unit and the buzzer is the output unit.

Think!

Can you tell which components in the kit can be used as input units and which can be used as output units?

Input Units - Sensors

A sensor (also known as a transducer) is a physical unit whose purpose is to sense or detect the characteristics of its environment (such as light, temperature or moisture levels) and then transmit this data to another device.

Sensor Pins

The three categories of sensor pins are as follows:

- Digital Pin
- Analog Pin
- Protocol Pin (Digital)

A protocol pin is also a kind of digital pin. I2C, Serial and SPI are frequently used digital pins.

Control Unit - DFRduino Uno

The microcontroller is the control unit. Think of it as the brain of your device.

Output Units - Actuators

There are many different types of actuators. An actuator is a type of device that is responsible for moving or controlling a system or mechanism. It is also the mechanism by which a control system acts upon an environment. It may convert electrical energy into motion, sound or light. A buzzer or speaker are actuators that output sound.

The Relationship Between Programs and Hardware

The input unit, control unit and output unit mentioned above are all hardware. In the context of our person analogy, hardware is the body of our device. However, the brain is much more important as it produces ideas and then controls actions every person takes. Code here functions as the mind of a person. Both body and mind are indispensable to a person.

The Difference between Digital & Analog Signals

Digital Signals:

A digital signal has two states: `HIGH` or `LOW`.

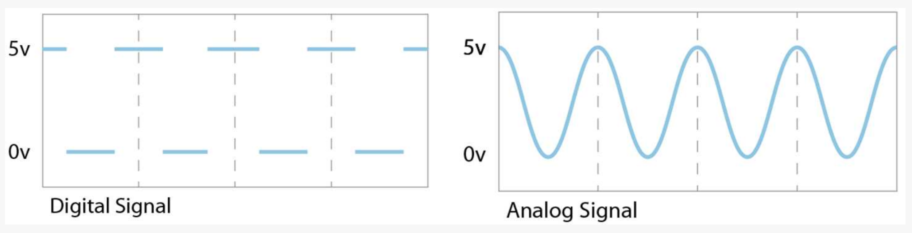
`HIGH` is a 5V signal and represents “1” (or on).

`LOW` is a 0V signal and represents 0 (or off).

Analog Signals:

An analog signal has a range of values.

The analog pins of your microcontroller can have between 0V and 5V is mapped to a range between 0 and 1023. For instance, 0 is mapped as 0V; 1023 is mapped as 5V and 512 is mapped as 2.5V.



Digital Signals & Analog Signals in the World of Electronics

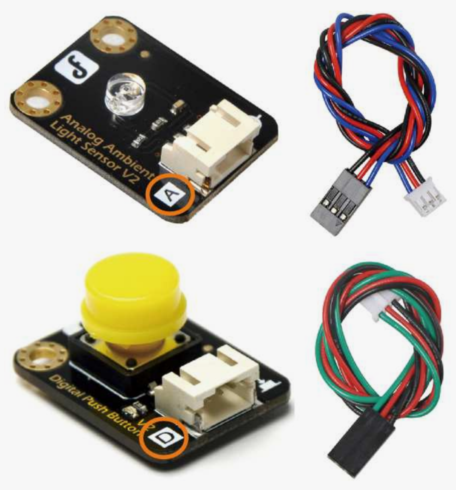
The input unit, the microcontroller and the output unit communicate by signals, which in turn are processed by code. How do input units and controllers communicate with each other? How do controllers communicate with the output units? To answer the above questions, we first need to understand two concepts: digital signals and analog signals.

“Digital” & “Analog” in a DFRobot Kit

Here are two ways to tell whether sensors in your DFRobot Kit are digital or analog:

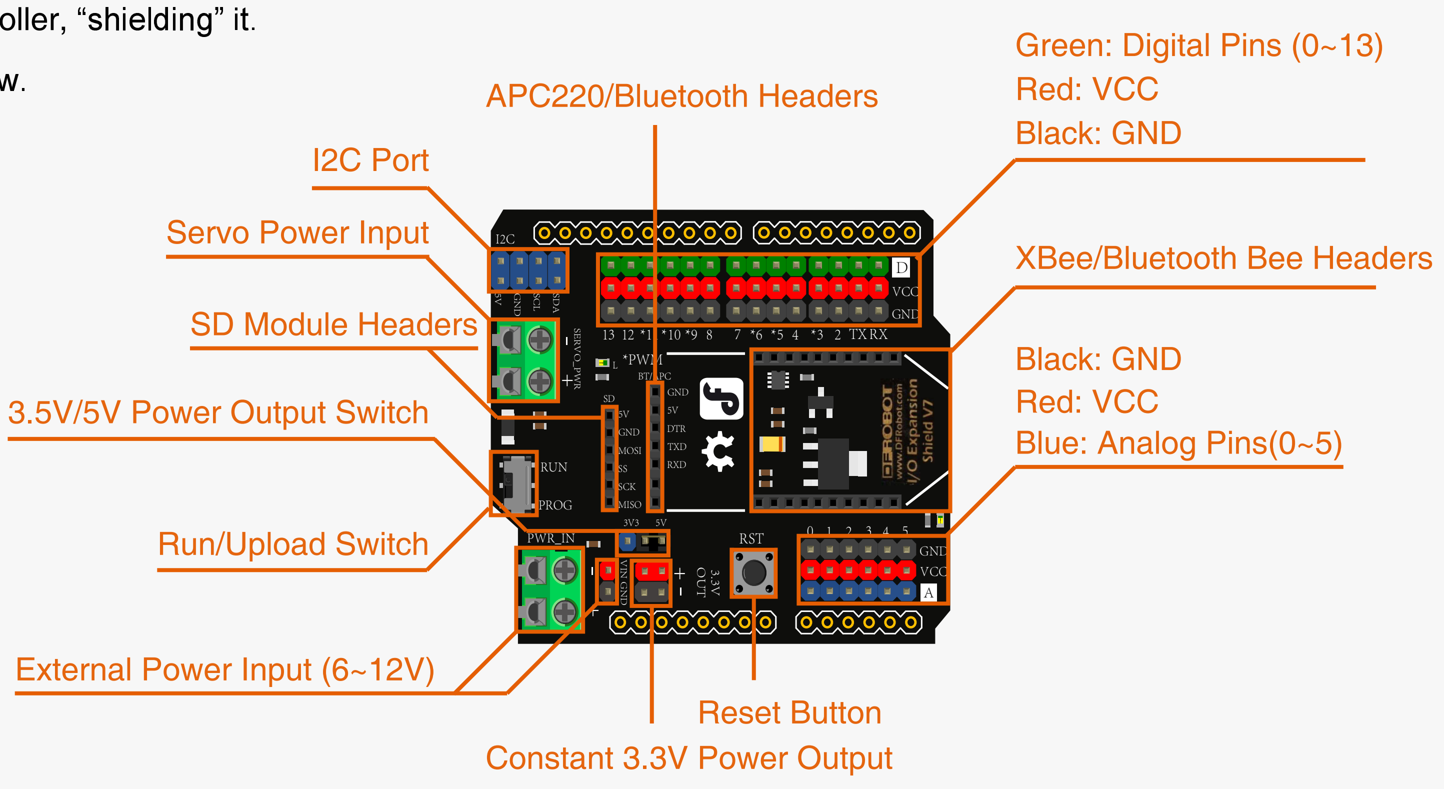
(1) If a sensor has a green wire it uses digital signal; if a sensor has a blue wire it uses analog signal.

(2) “A” or “D” may be marked on the sensor’s board. “D” represents “digital” and “A” represents “analog”.

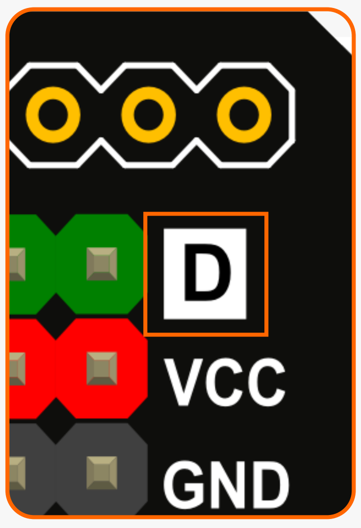


I/O Expansion Shield V7.1

Let’s take a look at the DFRobot I/O Expansion Shield. This shield gives us lots of extra pins to connect sensors to. It stacks neatly on top of your microcontroller, “shielding” it. The diagram below gives you an overview.



Analog and digital pins are marked on the expansion shield. The area with “A” is for analog sensors, and the area with “D” is for digital ones.



The advantage of the I/O expansion shield is that there are more power and ground pins than those on the bare microcontroller board. This gives you enough power pins to connect various sensors at the same time.

On the expansion shield, there are a row of power pins in red and a row of GND pins in black below the digital pins.

Different colors shown in our DF Kit have different meanings:

Green = Digital signal

Blue = Analog signal

Red = Power

Black = Ground

If you would like to know more about the sensor expansion shield, please refer to the DFRobot wiki for information:

http://www.dfrobot.com/wiki/index.php/IO_Expansion_Shield_for_Arduino_V7_SKU:DFR0265

This session gives you a general idea of what makes our devices work.
Are you excited to get your devices going? You’ll get a chance to in the next lesson!

Lesson 3

Analog and Digital Signals

03

DFROBOT
DRIVE THE FUTURE

Let's begin!

Let’s draw an analogy between microcontrollers and people

The microcontroller is the “brain”.

Code is the microcontroller’s “mind”.

Hardware (sensors and actuators) are the microcontroller’s “body”.

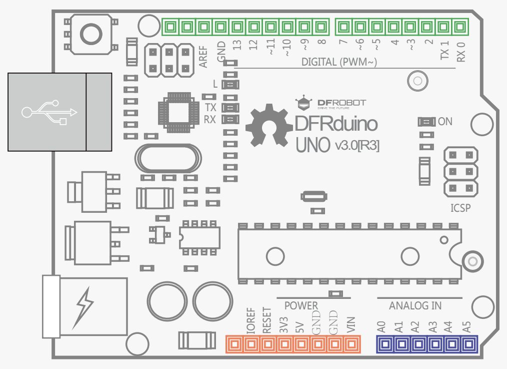
Signals, such as analog and digital signals, are the microcontroller’s “nerve impulses”.

In this session, we will explore the differences between analog and digital signals in more detail.

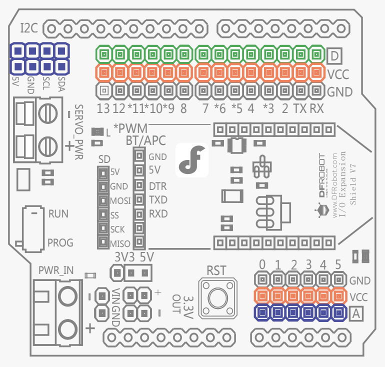
Digital Signals

We will use a digital button module to demonstrate a digital signal. This is included in your kit.

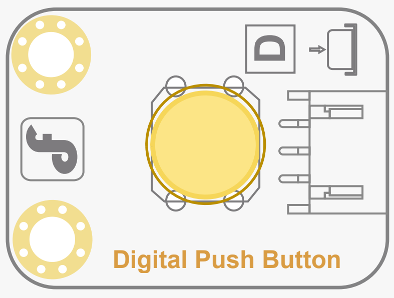
Parts Needed:



DFRduino Uno (and USB cable) x1



I/O Sensor Expansion Shield V7.1 x1



Digital Push Button Module x1

Connections

Take the Sensor Expansion Shield and stack it on top of your microcontroller – make sure that the pins are correctly aligned to avoid damage to the shield.

Take the Digital Push Button module and connect it to Digital Pin 2 (be sure that the power, ground and signal connections are correct or you risk damaging your components!)

See the diagram below for further details:

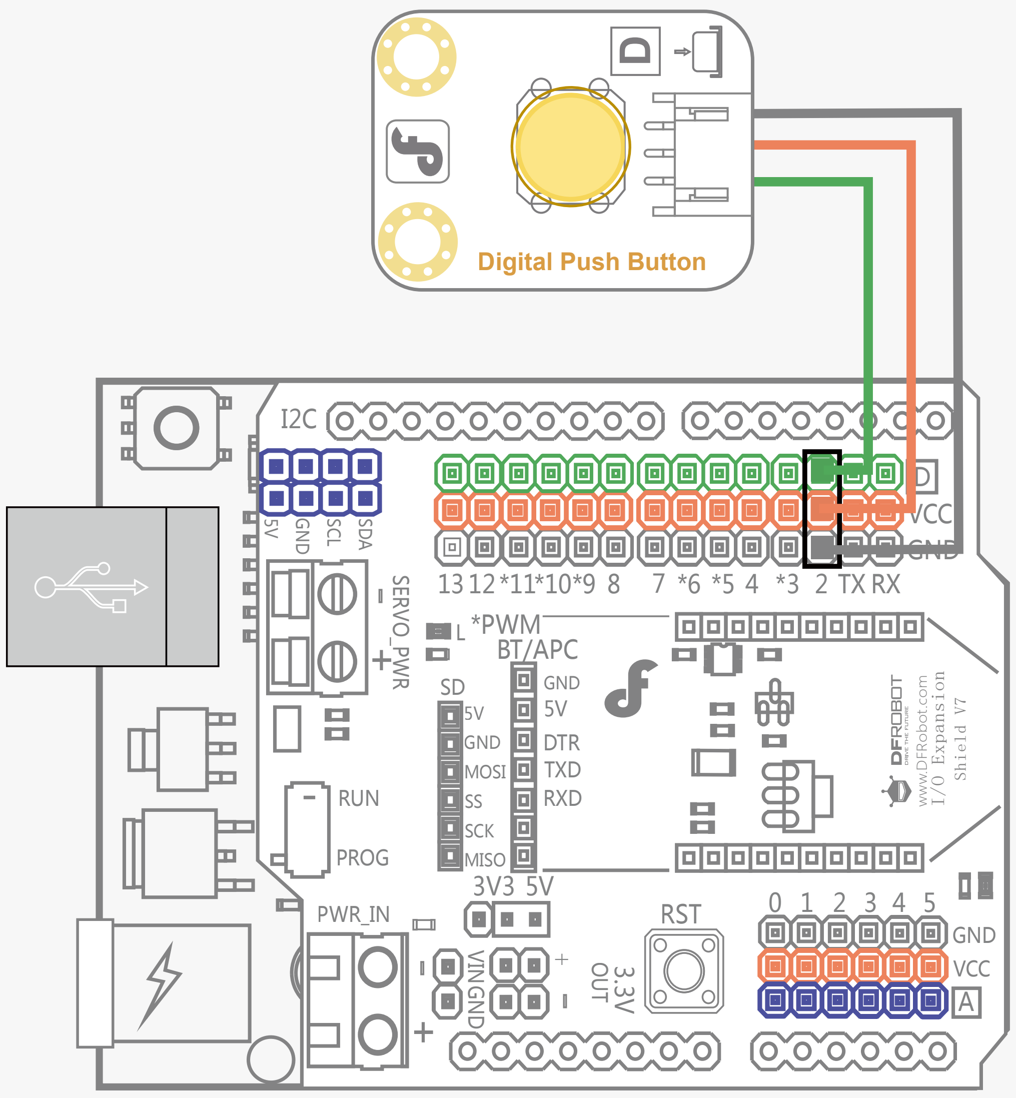


Fig. 2-1 Digital Pin Connection

When the connections are made, connect the USB cable. This will power on the microcontroller. We can now prepare our program to upload.

Serial Port Monitoring

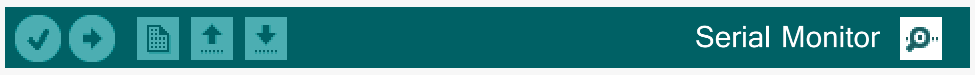
Open Arduino IDE and select: `File > Examples > 01.Basics > DigitalReadSerial`

The following code should appear:

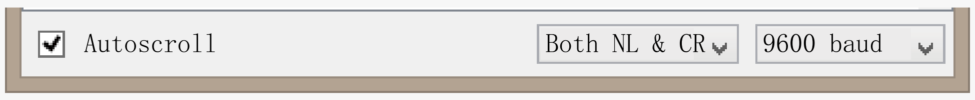
```
int pushButton = 2;              //connect to digital pin 2
void setup() {                    // initial function
  Serial.begin(9600);             // set up baud rate of the serial port
  pinMode(pushButton, INPUT);     // set the button to be in the output mode
}
void loop() {                                //main function
  int buttonState = digitalRead(pushButton);      // record statistics about the status of digital pin 2
  Serial.println(buttonState);    // Serial port print statistics about the status of digital pin 2
  delay(1);                              // delay 1ms
}
```

Click “Upload”. Arduino IDE will verify the code and then upload it to your microcontroller.

Once the upload is complete, open the serial monitor. To do this, navigate to `Tools > Serial Monitor`, or click the magnifying glass icon on the top right hand corner of the toolbar.

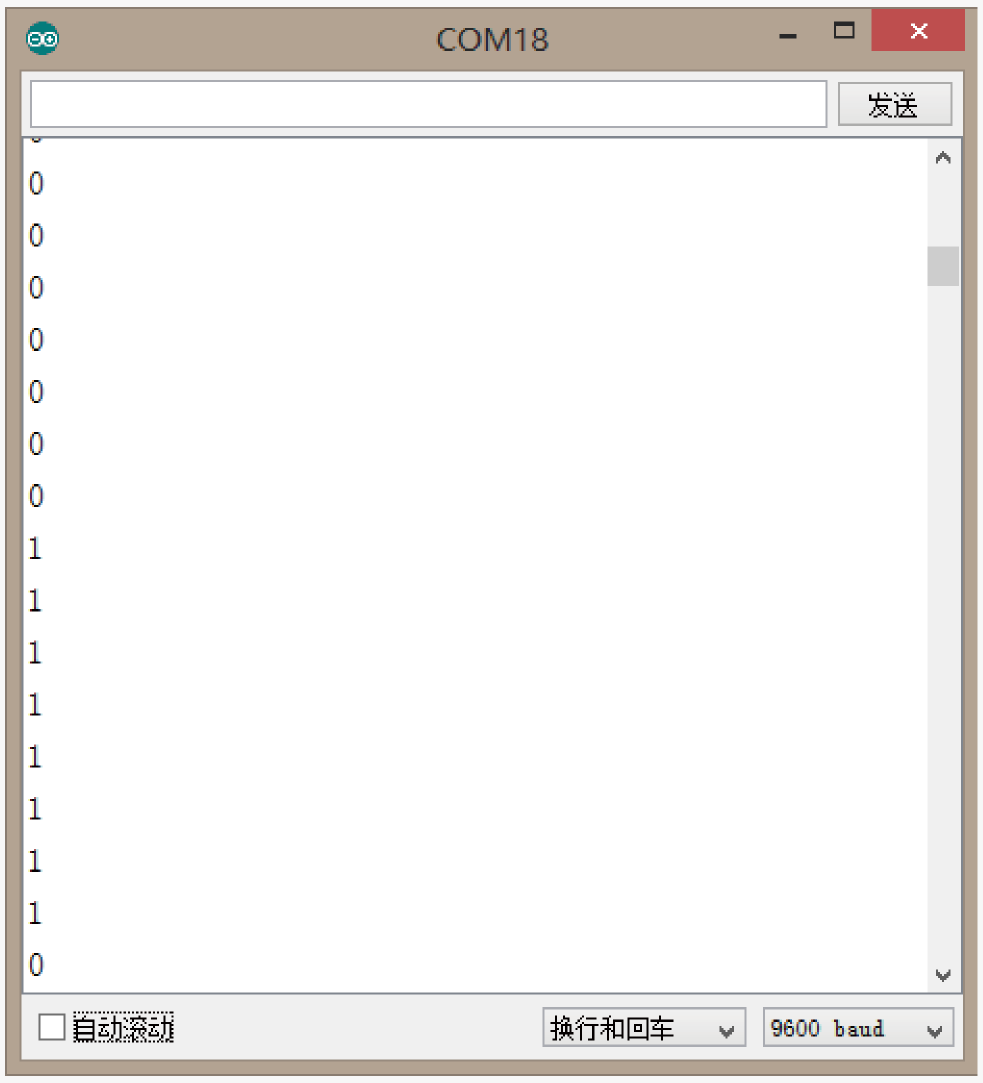


The baud rate is the data speed that the serial port communicates. It needs to be set to `9600 baud`.



The serial monitor shows information from the microcontroller. We are going to use it to view the status of the button. You should see “0” appearing continuously. This indicates that the button is off (not being pressed).

If you press the button, you will see “1” appear for as long as the button is pressed. This indicates that the button is on (being pressed).



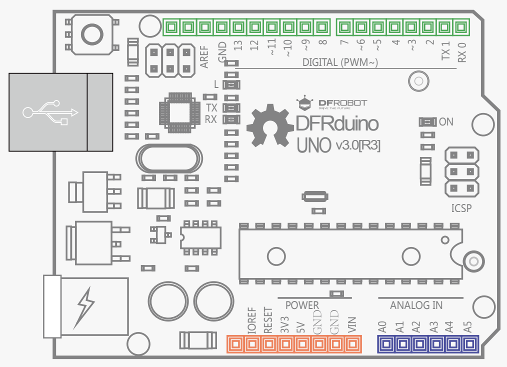
```
0
0
0
0
0
0
0
1
1
1
1
1
1
1
1
0
```

“0” and “1” or off and on are the two values of digital signal. We can use this in various ways with our devices.

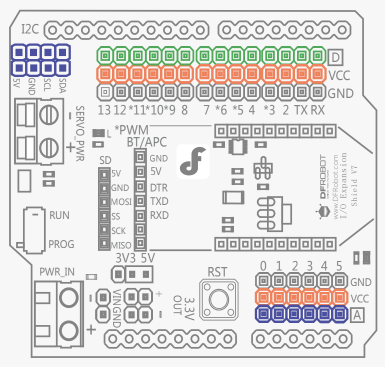
Analog Signal

In this section we are going to explore analog signal in more detail. We are going to use a sensor which uses analog values: a rotation sensor.

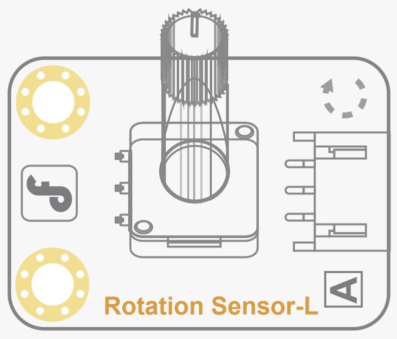
Parts Needed:



DFRduino Uno (and USB cable) x1



I/O Sensor Expansion Shield V7.1 x1



Analog Rotation Sensor x1