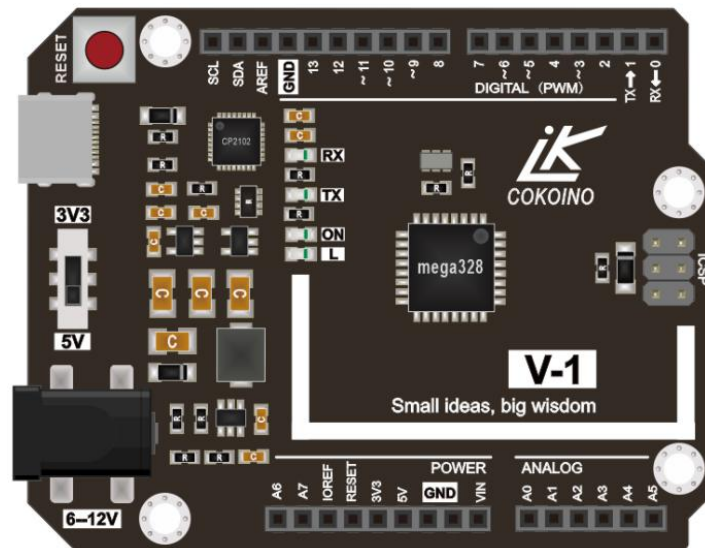


---

# 1. LK COKOINO V-1 Board for Arduino



## Table of Contents

1. Overview.....	2
2. Introduction of LK COKOINO V-1 board.....	3
3. Install the Arduino Software (IDE) on Windows PCs.....	8
4. Introduction to the Arduino IDE.....	16
5. Upload your first sketch.....	21
6. Arduino library file.....	24

---

## 1.Overview

This is an upgrade controller board based on arduino UNO R3, which is fully compatible with arduino UNO R3 and more powerful. Its features are as follows:

- 1) The most popular type C USB interface: easy to plug and unplug, can be connected to DC5V \ 2A power supply
- 2) CP2102 USB to serial chip: compatible with various PC systems
- 3) DC to DC 5V power supply voltage stabilization system: makes the control board reduce heat and save more power, the maximum output of the control board is 2A current
- 4) More powerful DC3.3V voltage regulator chip: maximum load 3.3V \ 500mA
- 5) Add two analog input ports A6 and A7: connect more modules that output analog signals
- 6) System voltage selection switch of 3.3V and 5V system: perfect communication with 3.3V or 5V external system

### **Introduction to Arduino – What is it?**

Arduino is an open-source electronics platform based on easy-to-use hardware and software.

Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

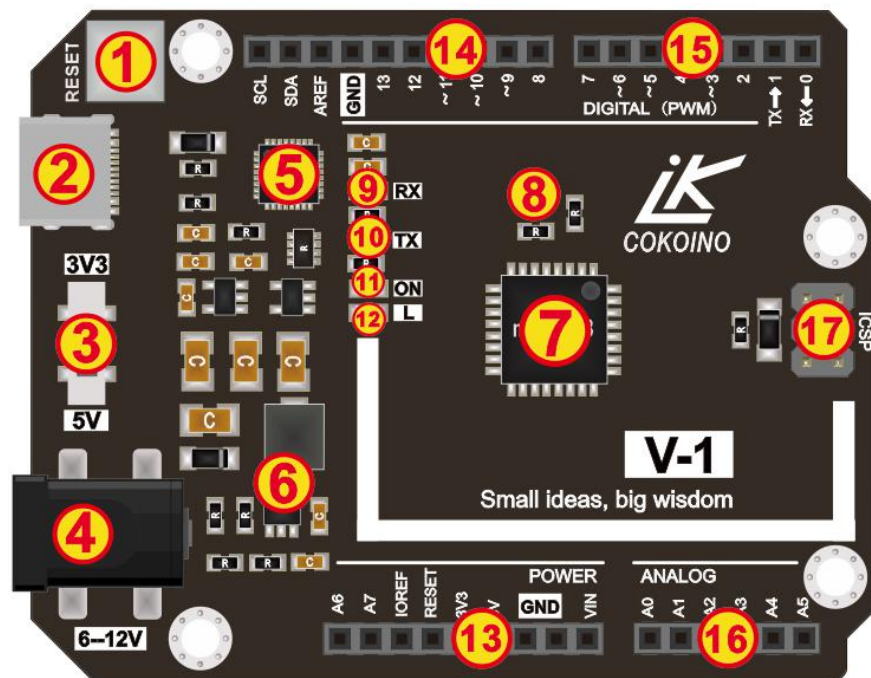
### **Features:**

Open source software - The Arduino software is published as open source tools, available for extension by experienced programmers. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.

Open source hardware - The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it.


Cross-platform - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.

## 2.Introduction of LK COKOINO V-1 board



1	Reset Button	Press the system reset button to trigger a reset (reset the main control IC).
2	Type C USB Port	Connect the PC to the board for uploading the code.
3	System Voltage Selection Switch	Select the system to work with 3.3V or 5V voltage.
4	External Power Input DC Socket	Model DC005, external DC power input, voltage range DC6-18V, recommended: DC6-12V.
5	USB to Serial System	The CP2102 chip enables the motherboard to communicate with the PC, such as uploading code and serial communication.
6	5V Voltage Stabilization System	DC to DC 5V voltage stabilization system, convert DC external power supply to DC 5V \ 2A.
7	Master IC	Mega328P model, the core processor of the development board, like the human brain can handle events such as receiving, sending, judging, interrupting, timing, driving.
8	Crystal Oscillator	The 16M crystal oscillator provides stable clock timing for the main control IC, just like the human heart.
9	Serial Tort Receiving Indication LED	When the serial port of the board receives data, this LED will flash.
10	Serial Port Sending Indication LED	When the serial port of the board sends data, this LED will flash.
11	Power Indicator	When the USB port or DC socket is connected to the power supply, this LED will light.

12	Onboard LED	This LED light is controlled by the 13 IO port of the board, and a simple program can be written to control it to turn on and off.
13	Power, Reset, Analog Port	VIN: Output the power that the DC socket provides GND: GND. 5V: DC5V \ 2A power output port 3V3: DC3.3V \ 500mA power output port RESET: Reset input port, active low (reset the main control IC) IOREF: DC5V output port A7: Analog input port (main control IC analog input port) A6: Analog input port (main control IC analog input port)
14	Digital IO Port	8: Digital IO port. 9: Digital IO port, multi-function IO port, also used for PWM output port (using timer1). 10: Digital IO port, multi-function IO port, also used for PWM output port (using timer1) and SPI SS. 11: Digital IO port, multi-function IO port, also used for PWM output port (using timer2) and MOSI of SPI. 12: Digital IO port, multi-function IO port, also used for SPI MISO. 13: Digital IO port, multi-function IO port, also used for SCK of SPI. GND: GND. AREF: Analog input reference voltage SDA: IIC data line (multi-function IO port, common pin with A4). SCL: IIC clock line (multi-function IO port, common pin with A5).
15	Digital IO Port	0: Digital IO port, multi-function IO port, also used for serial port receiving port 1: Digital IO port, multi-function IO port, also used for serial port 2: Digital IO port, multi-function IO port, also used for external interrupt 0 3: Digital IO port, multi-function IO port, also used for external interrupt 1 and PWM output port (using timer2) 4: Digital IO port 5: Digital IO port, multi-function IO port, also used for PWM output port (using timer0) 6: Digital IO port, multi-function IO port, also used for PWM output port (using timer0) 7: Digital IO port

16	Digital IO Port	A0: Analog input port, multi-function IO port, also used for digital IO port
		A1: Analog input port, multi-function IO port, also used for digital IO port
		A2: Analog input port, multi-function IO port, also used for digital IO port
		A3: Analog input port, multi-function IO port, also used for digital IO port
		A4: Analog input port, multi-function IO port, also used for digital IO port, common with SDA
		A5: Analog input port, multi-function IO port, also used for digital IO port, common pin with SCL
17	SPI Port 	MISO: Master output, slave output port, common pin with IO 12 pin.
		MOSI: Master input, slave output port, common pin with IO 11
		SCK: Clock line, common pin with IO 13
		REST: Reset IO port
		VCC: 5V voltage output port
		GND: GND

## 2.1 Specification

Microcontroller: ATmega328P

Operating Voltage : 5V

Input Voltage (recommended): 6-12V

Input Voltage (limit): 6-18V (Is not recommended to use)

Digital I/O Pins: 14 (of which 6 provide PWM output)

PWM Digital I/O Pins: 6 (D3, D5, D6, D9, D10, D11)

Analog Input Pins: 6 (A0~A7)

DC Current per I/O Pin : 20 mA

DC Current for 3.3V Pin: 500mA

Flash Memory: 32 KB (ATmega328P) of which 0.5 KB used by bootloader

SRAM: 2 KB (ATmega328P)

EEPROM: 1 KB (ATmega328P)

Clock Speed: 16 MHz

LED\_BUILTIN: 13 (IO)

size: 69.5mm\*55.0mm

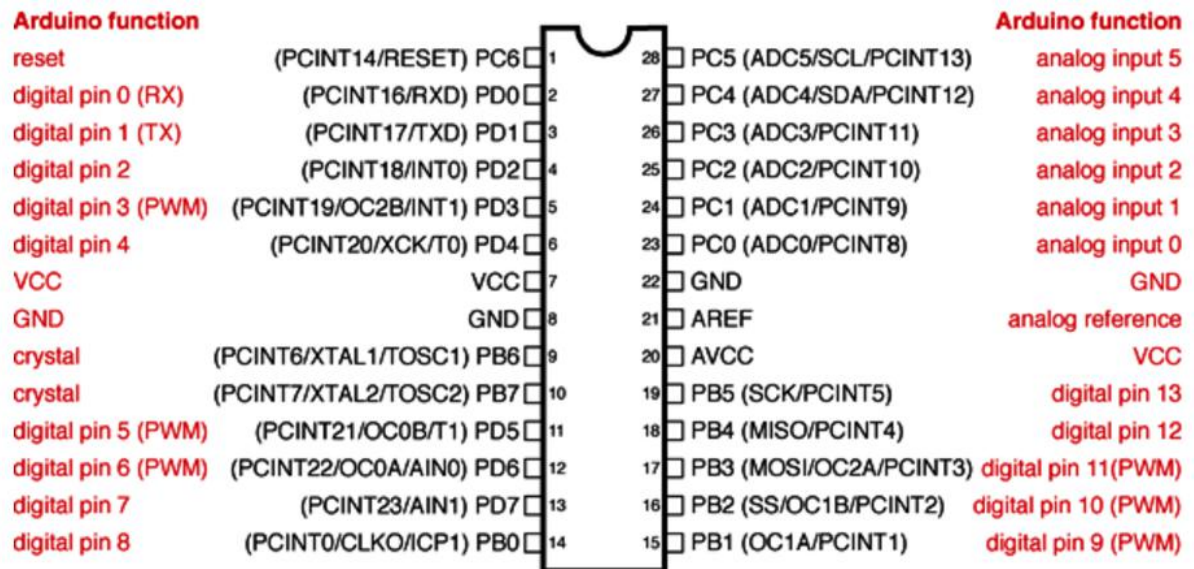
Bootloader: UNO REV3

dimensional drawing:

<https://github.com/Cokoino/CKD0002/tree/master/Drawing>

## 2.2 Input and Output

See the mapping between Arduino pins and ATmega328P ports. The mapping for the Atmega8, 168, and 328 is identical.



Digital Pins 11, 12 & 13 are used by the ICSP header for MOSI, MISO, SCK connections (Atmega168 pins 17, 18 & 19). Avoid low-impedance loads on these pins when using the ICSP header.

Each of the 14 digital pins on the Uno can be used as an input or output, using `pinMode()`, `digitalWrite()`, and `digitalRead()` functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

### In addition, some pins have specialized functions:

**Serial:** 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.

**External Interrupts:** 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the `attachInterrupt()` function for details.

**PWM:** 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the `analogWrite()` function.

**SPI:** 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.

**LED:** 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

**I2C:** A4 or SDA pin and A5 or SCL pin. Support I2C communication using the `Wire` library.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though it is possible to change

---

---

the upper end of their range using the AREF pin and the `analogReference()` function. There are a couple of other pins on the board:

AREF. Reference voltage for the analog inputs. Used with `analogReference()`.

Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

## **Communication**

V1 board has a number of facilities for communicating with a computer, another Arduino/Genuino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An CP2102 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The CP2102 uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A `SoftwareSerial` library allows serial communication on any of the V1's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino Software (IDE) includes a `Wire` library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the `SPI` library.

## **Automatic (Software) Reset**

Rather than requiring a physical press of the reset button before an upload, the V1 board is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the CP2102 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino Software (IDE) uses this capability to allow you to upload code by simply pressing the upload button in the interface toolbar. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the V1 board is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the V1 board. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.



---

### 3. Install the Arduino Software (IDE) on Windows PCs

Get the latest version from this link (<https://www.arduino.cc/en/Main/Software>). You can choose between the Installer (.exe) and the Zip packages.



**ARDUINO 1.8.12**

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, for Windows 7 and up  
**Windows** ZIP file for non admin install

**Windows app** Requires Win 8.1 or 10  
[Get](#)

**Mac OS X** 10.10 or newer

**Linux** 32 bits  
**Linux** 64 bits  
**Linux** ARM 32 bits  
**Linux** ARM 64 bits

[Release Notes](#)  
[Source Code](#)  
[Checksums \(sha512\)](#)

Click "JUST DOWNLOAD" to download

### Contribute to the Arduino Software

Consider supporting the Arduino Software by contributing to its development. (US tax payers, please note this contribution is not tax deductible). [Learn more on how your contribution will be used.](#)



SINCE MARCH 2015, THE ARDUINO IDE HAS BEEN DOWNLOADED **41,706,515** TIMES. (IMPRESSIVE!) NO LONGER JUST FOR ARDUINO AND GENUINO BOARDS, HUNDREDS OF COMPANIES AROUND THE WORLD ARE USING THE IDE TO PROGRAM THEIR DEVICES, INCLUDING COMPATIBLES, CLONES, AND EVEN COUNTERFEITS. HELP ACCELERATE ITS DEVELOPMENT WITH A SMALL CONTRIBUTION! REMEMBER: OPEN SOURCE IS LOVE!

**\$3** **\$5** **\$10** **\$25** **\$50** **OTHER**

**JUST DOWNLOAD** **CONTRIBUTE & DOWNLOAD**

#### Previous Releases

Download the previous version of the current release the classic Arduino 1.0.x, or the Arduino 1.5.x Beta version.

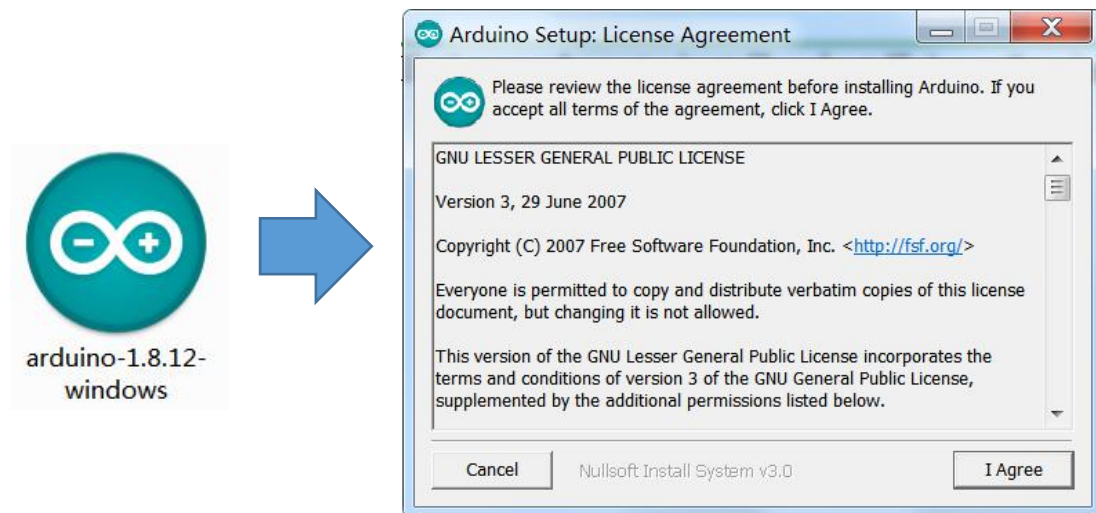
<https://www.arduino.cc/en/Main/OldSoftwareReleases#previous>



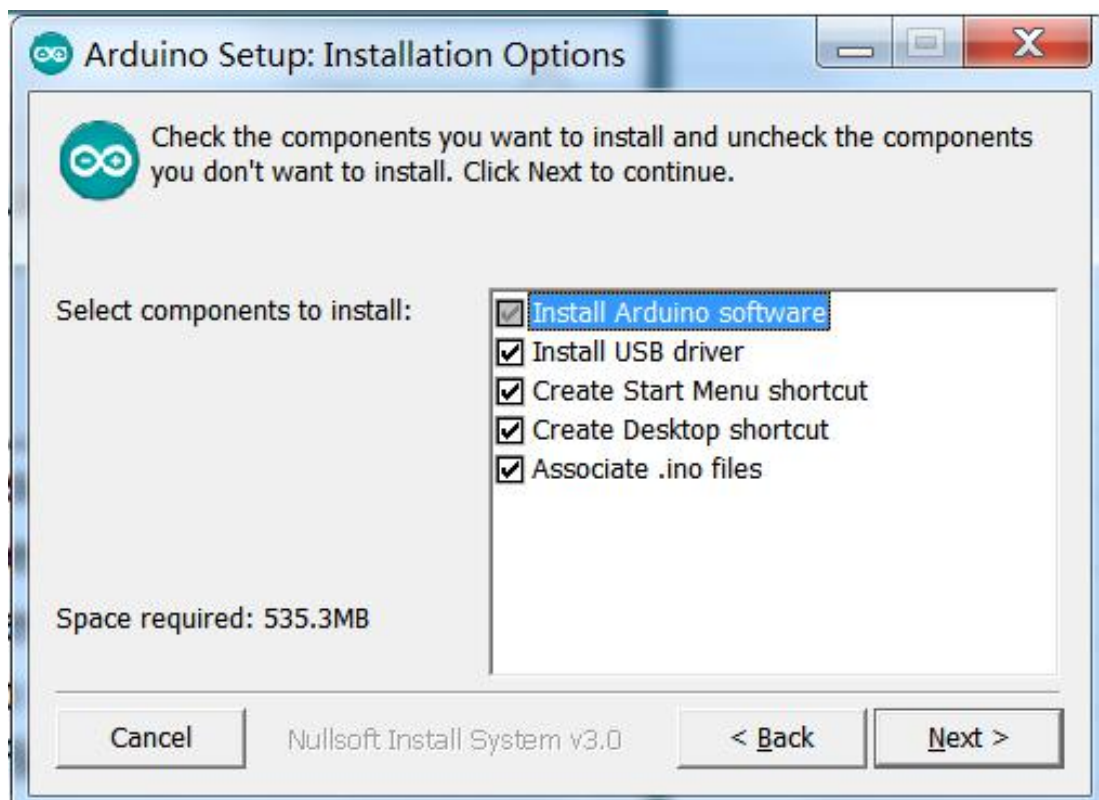
---

### 3.1 Download the Installer (.exe)

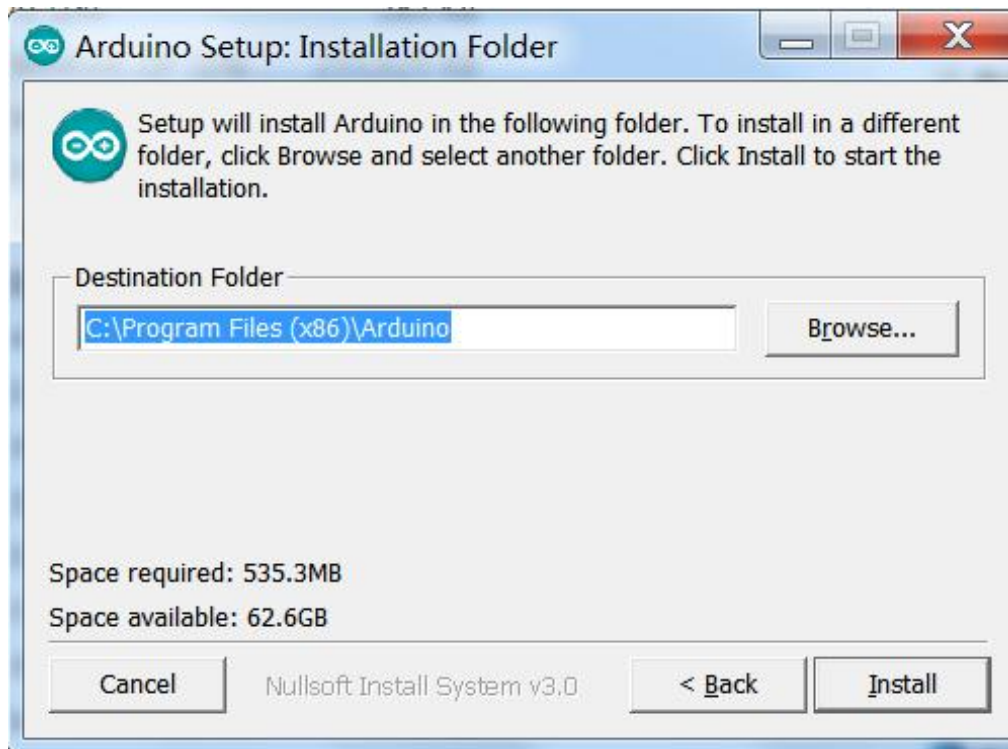
3.1.1 When the download finishes, double-click the installation package, click "I Agree" to install:



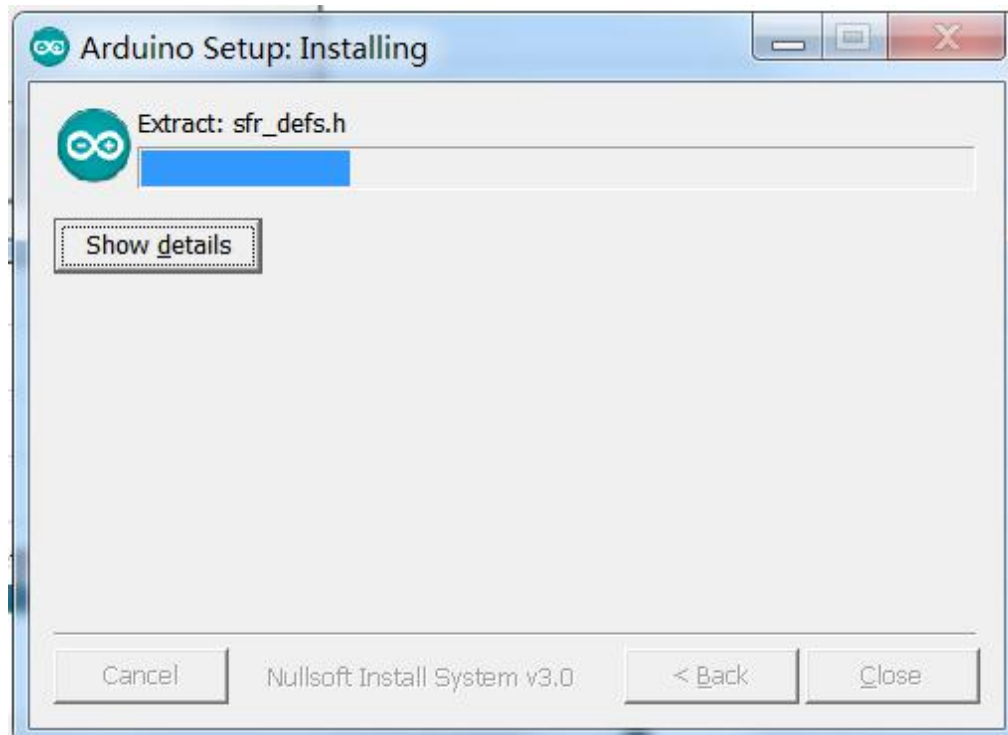
3.1.2 Click next



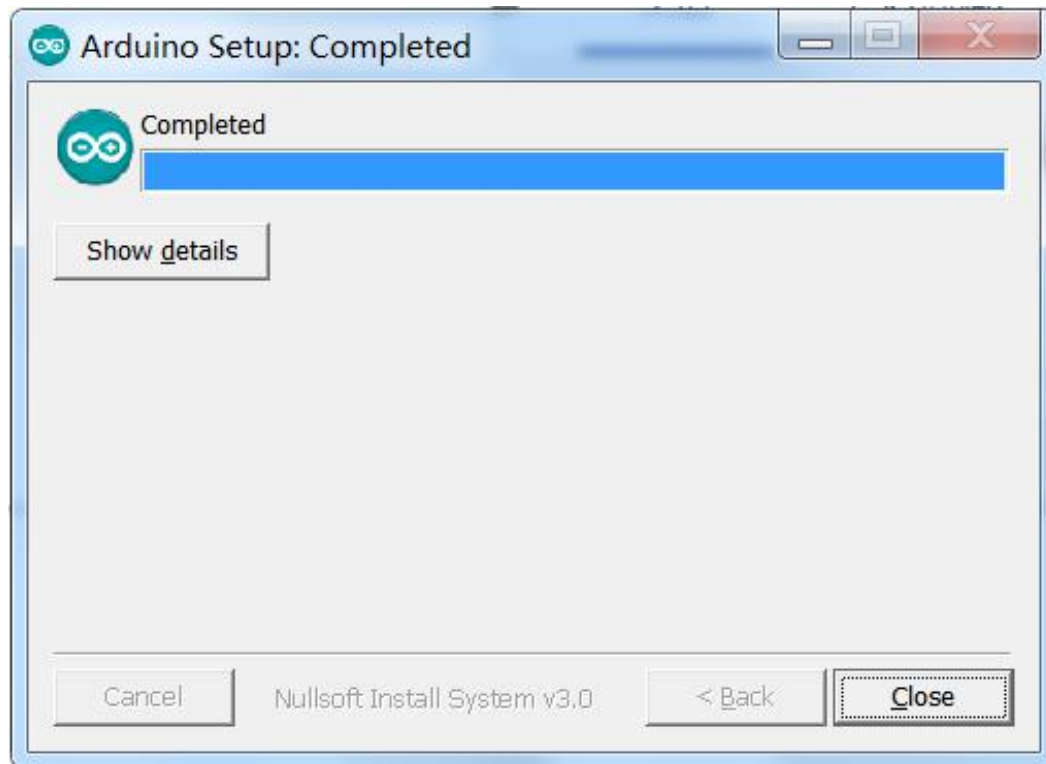
3.1.3 Choose the installation directory (we suggest to keep the default one) and click install



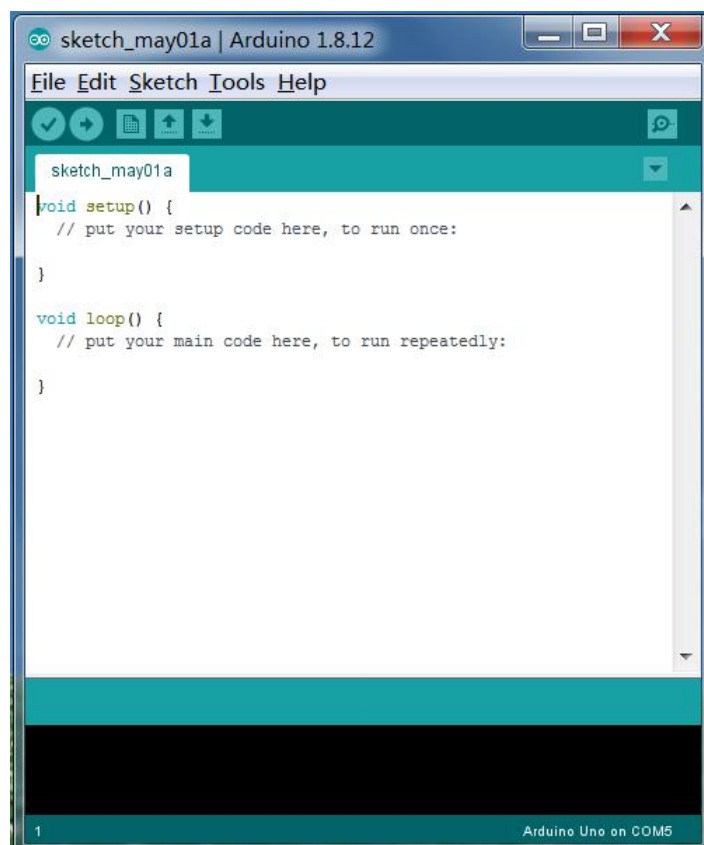
3.1.4 The process will extract and install all the required files to execute properly the Arduino Software (IDE) and please allow the installation process when you get a warning from the operating system.



### 3.1.5 Completed and click close

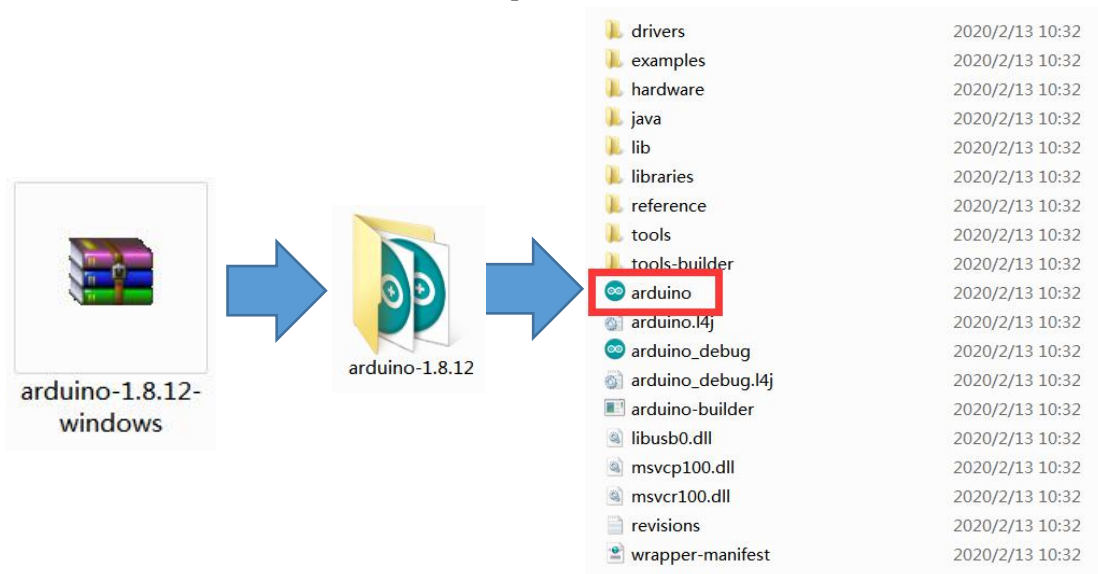


3.1.6 You can find an arduino icon on the computer desktop, double click this icon will pop up the interface of arduino IDE.



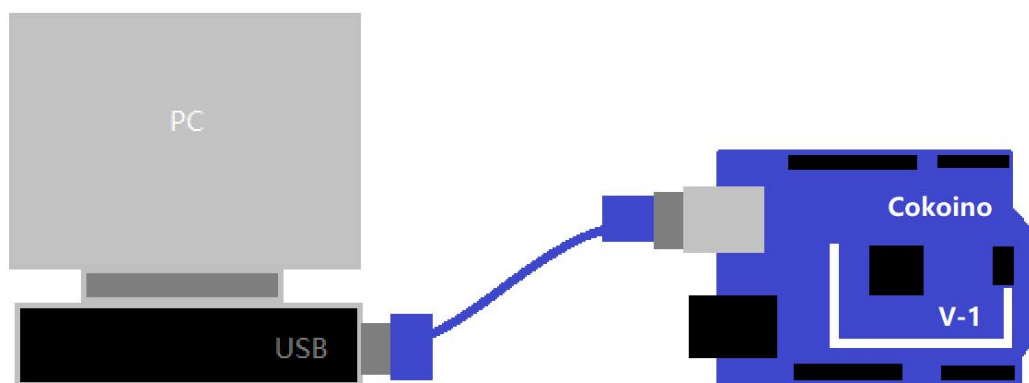
### 3.2 Download the Zip package

Unzip the downloaded zip file, put it in a system disk, double-click to open the unzipped folder, and then click the arduino icon to open the IDE, as shown below:



### 3.3 Install The Driver

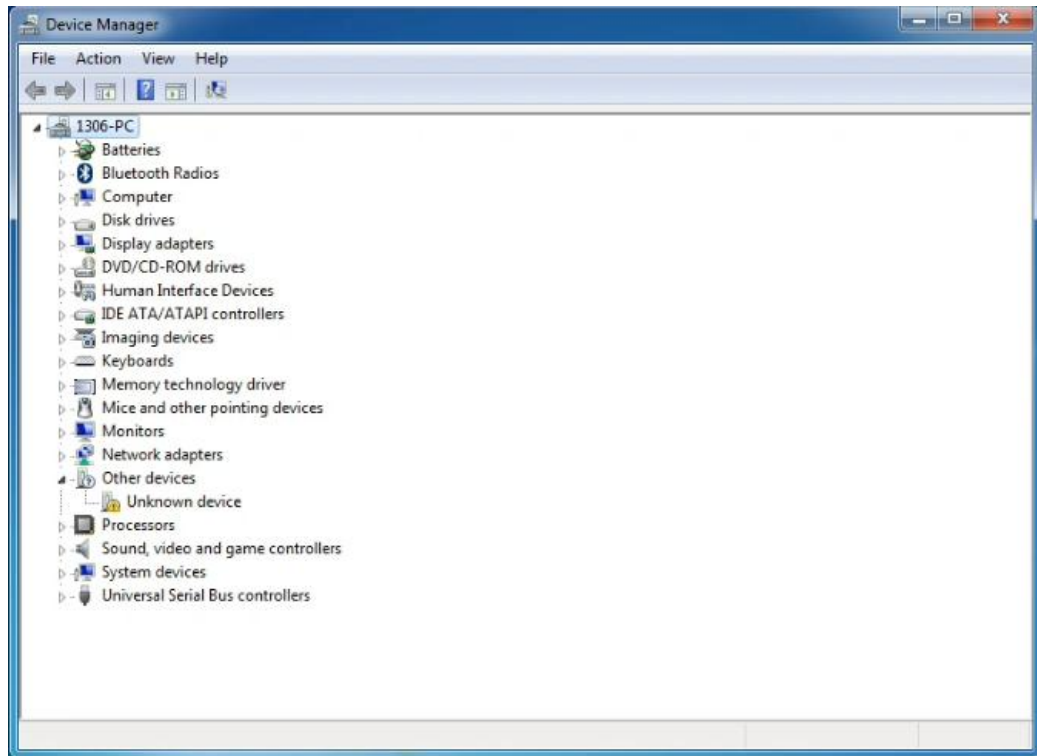
3.3.1 First, you need to connect the board to the computer with a USB cable.



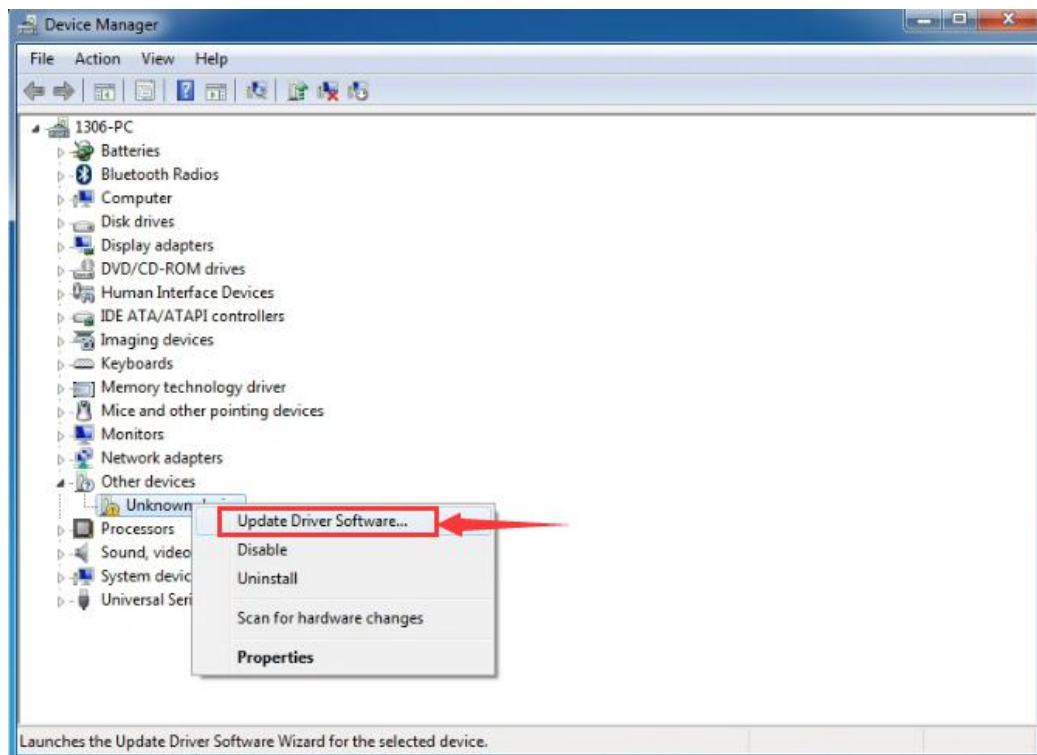
3.3.2 Click on the Start Menu, and open up the Control Panel.

While in the Control Panel, navigate to System and Security. Next, click on System. Once the System window is up, open the Device Manager.

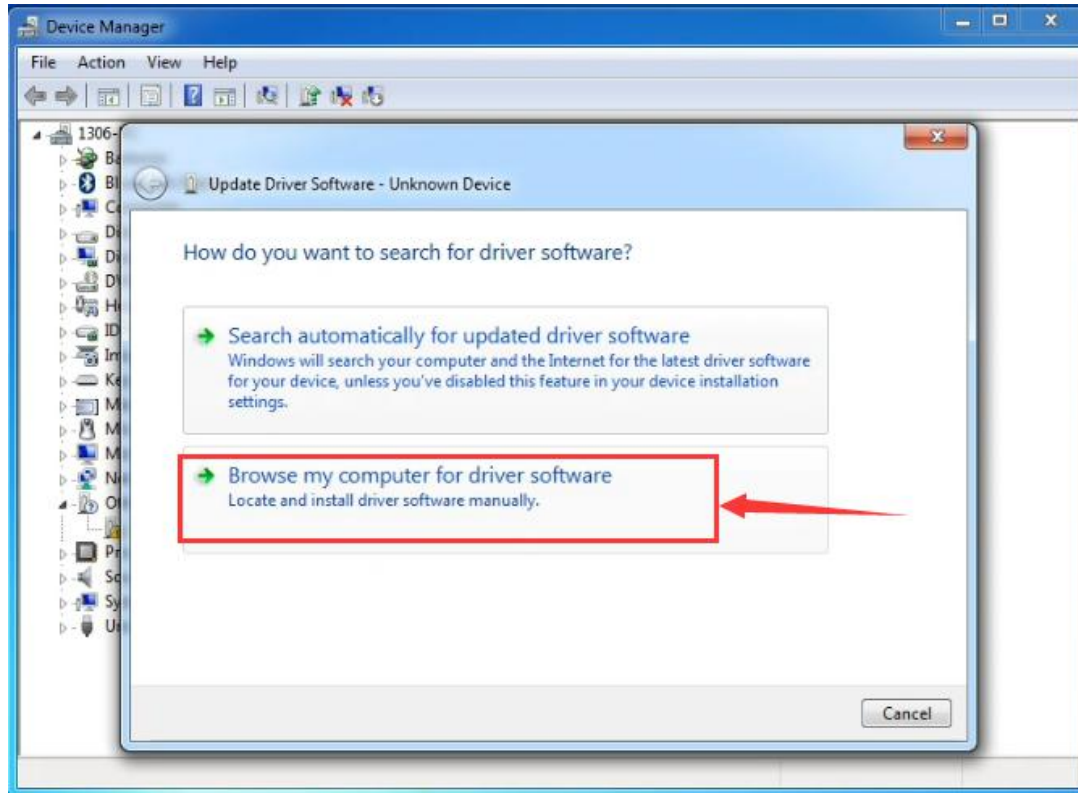
Look under "Other Devices" for "Unknown Device".



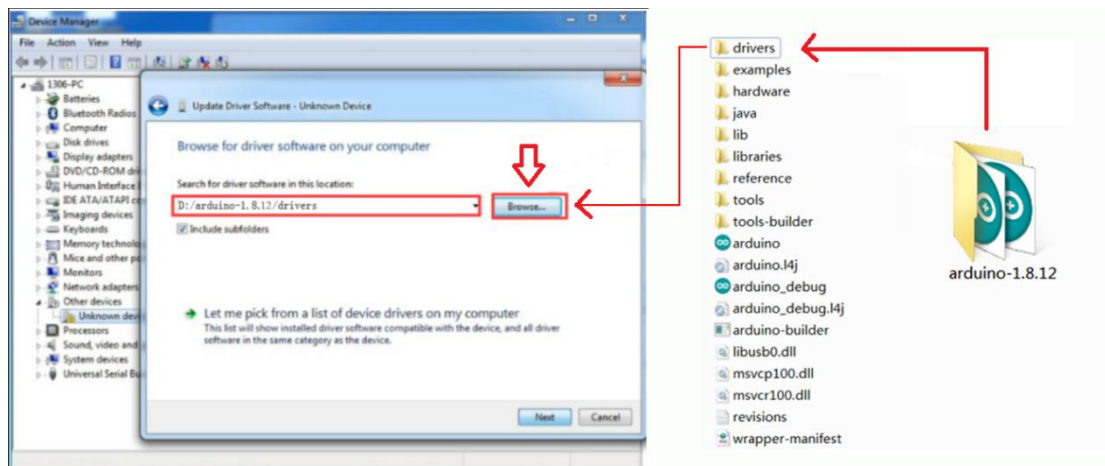
3.3.3 Right click on the "Unknown Device" port and choose the "Update Driver Software" option.



3.3.4 Next, choose the "Browse my computer for Driver software" option.

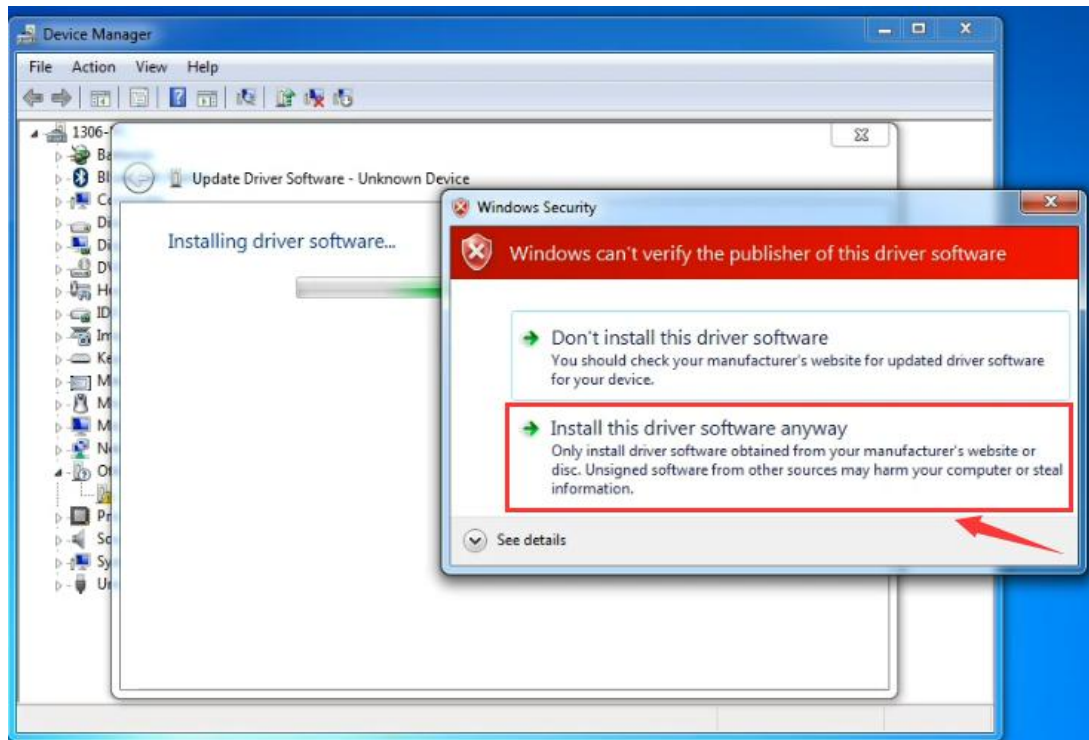


3.3.5 Finally, navigate to and select the "Drivers" folder of the Arduino Software download .

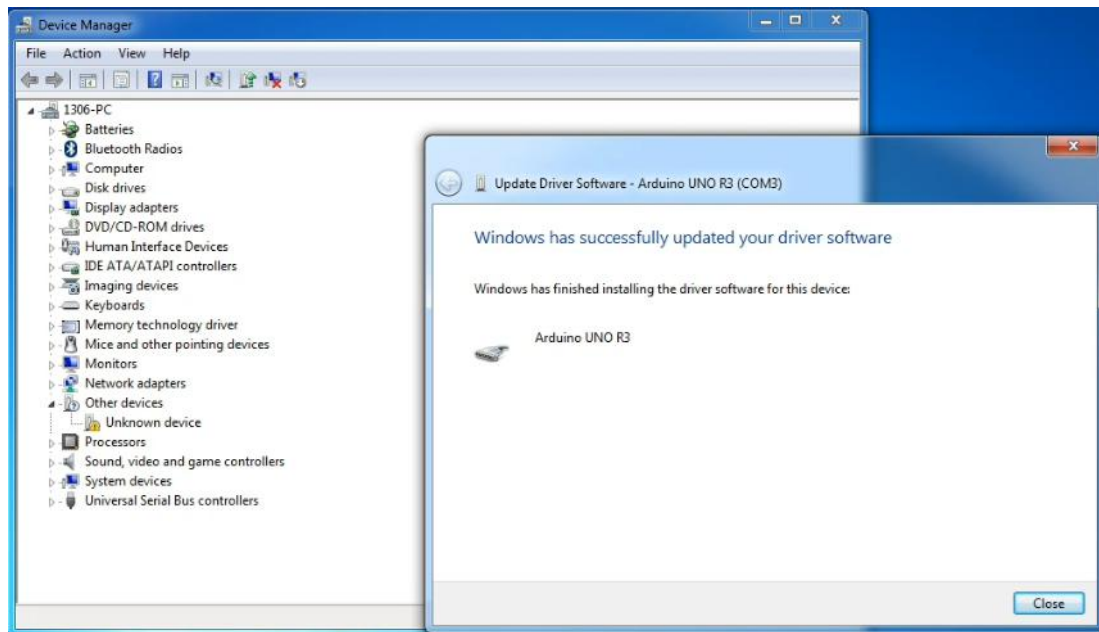


3.3.6 Install this driver software anyway

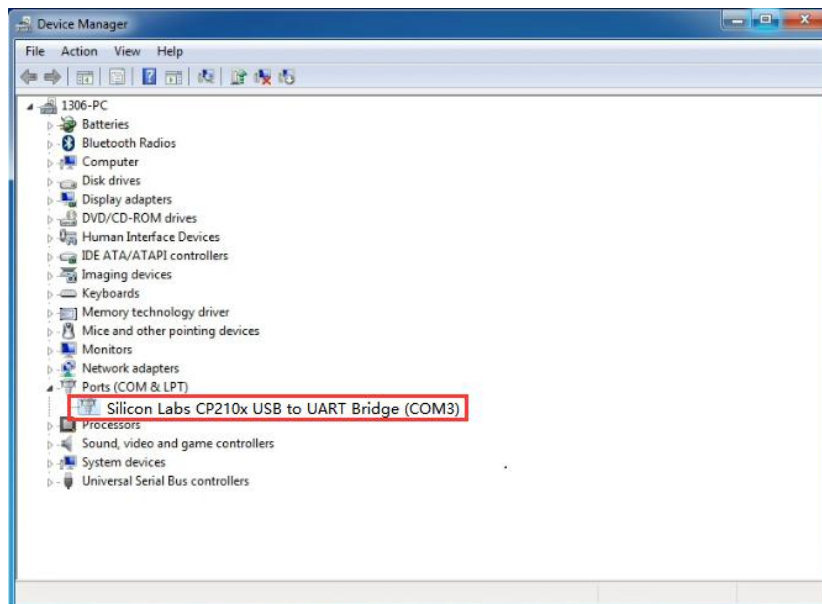




3.3.7 After the installation driver is complete, click Close.









3.3.8 Open the Device Manager, "Unknown Device" has been recognized, then you can start programming with arduino.



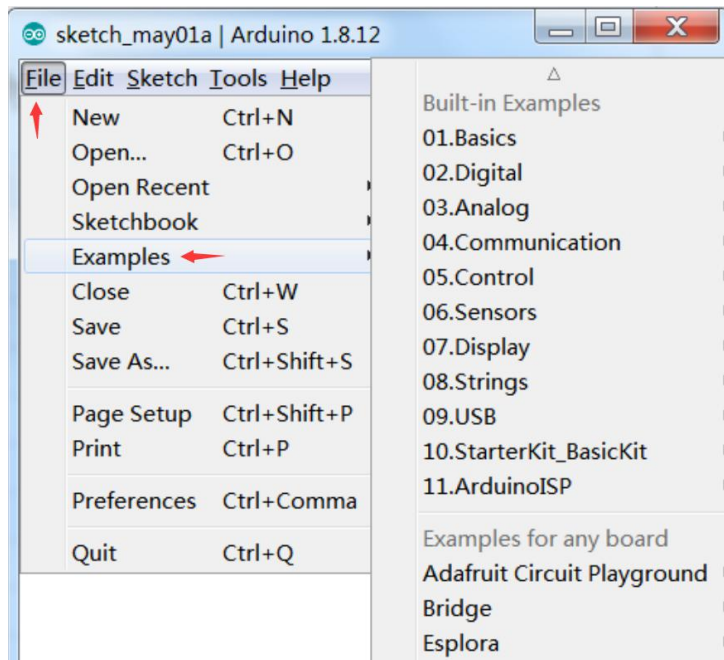
## 4.Introduction to the Arduino IDE

The functions of each button on the Toolbar are listed below:

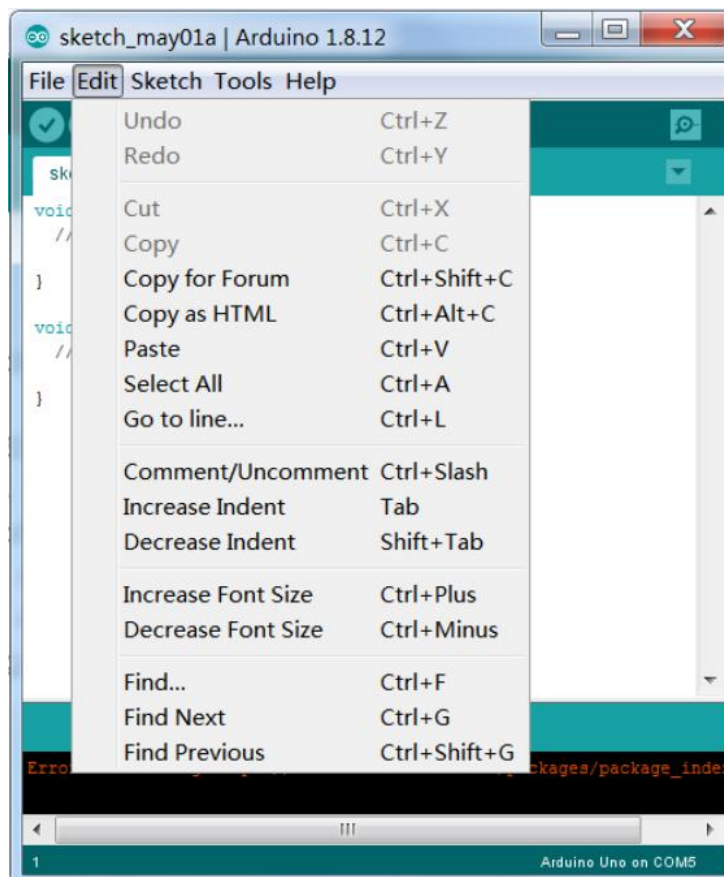


	Check the code for errors
<b>Verify/Compile</b>	
	Upload the current Sketch to the Arduino
<b>Upload</b>	
	Create a new blank Sketch
<b>New</b>	
	Show a list of Sketches
<b>Open</b>	
	Save the current Sketch
<b>Save</b>	
	Display the serial data being sent from the Arduino
<b>Serial Monitor</b>	

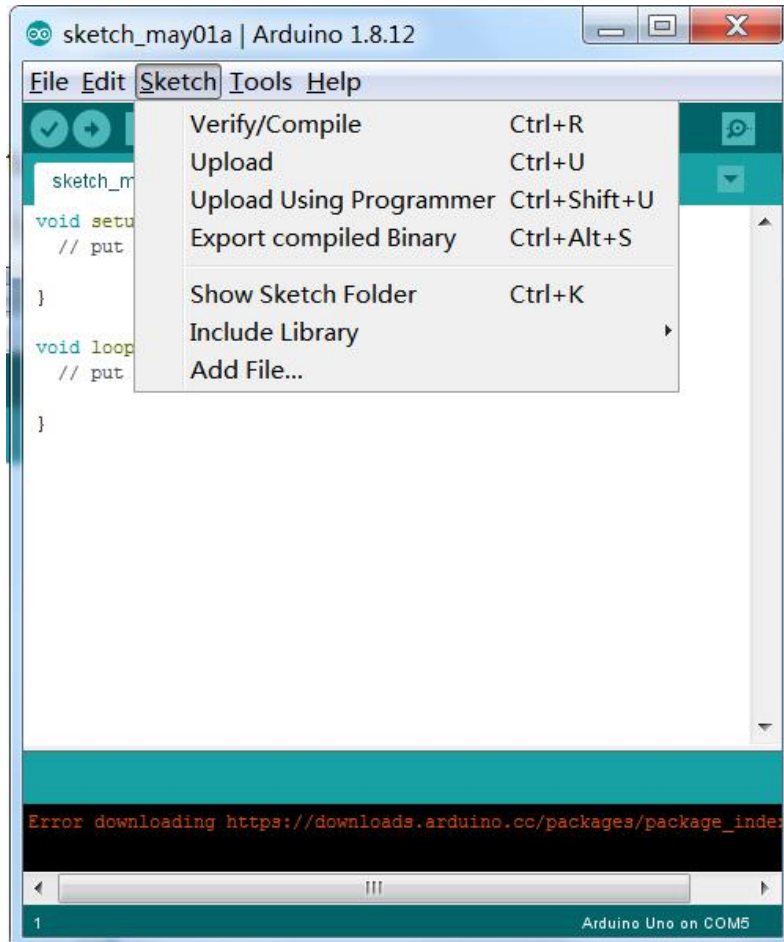
**File** : New sketch, open existing sketch, etc. There are many ready-made sample sketches under Examples, as shown below:



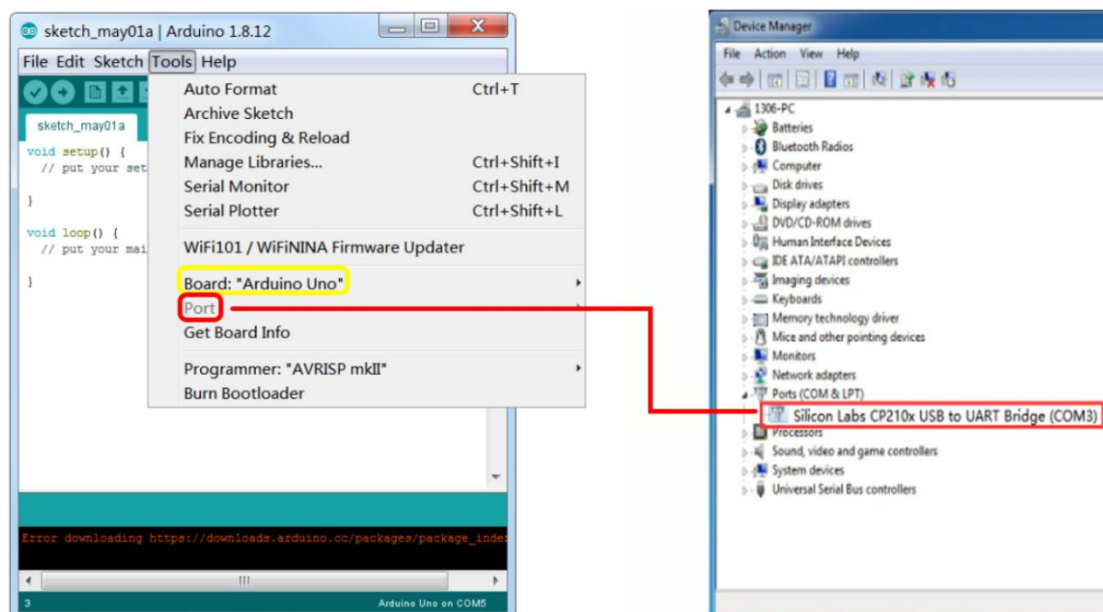
**Edit** :Undo and restore the sketch. When you write the wrong sketch, you want to restore the original sketch. This function is often used.



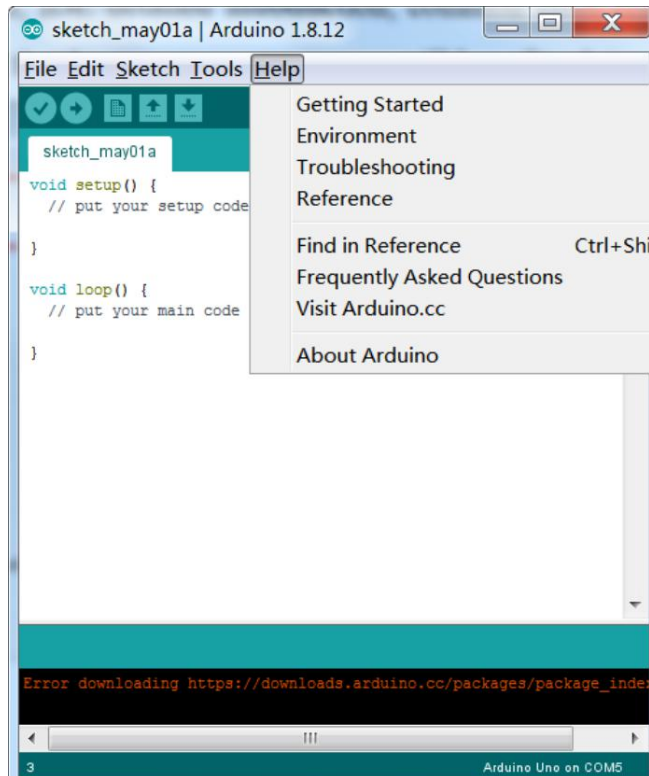
**Sketch** : compile and download the sketch



**Tools** : When uploading the program, you need to select the corresponding board type and port, otherwise the program will not be burned into the board, as shown below:

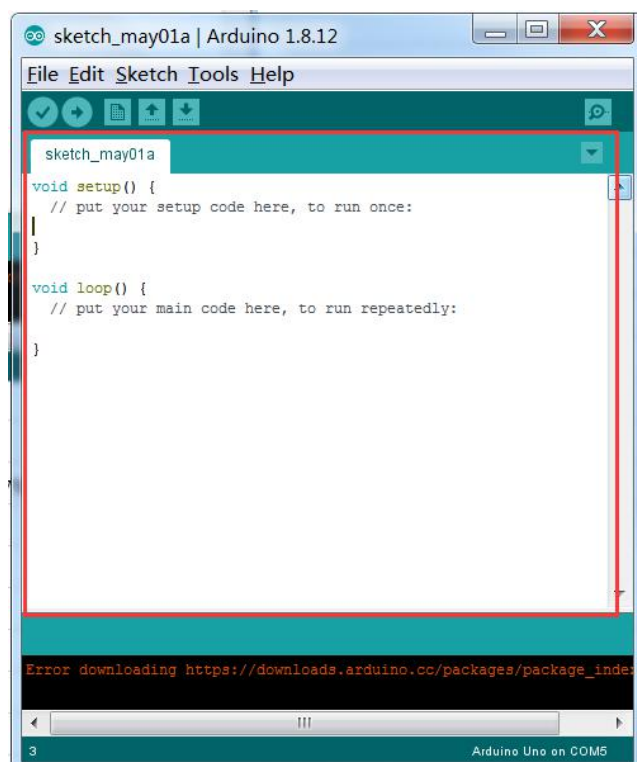


**Help** :About the official tutorial, IDE detailed introduction, common error resolution, programming language learning, etc., it is recommended that beginners must see, you will benefit a lot.

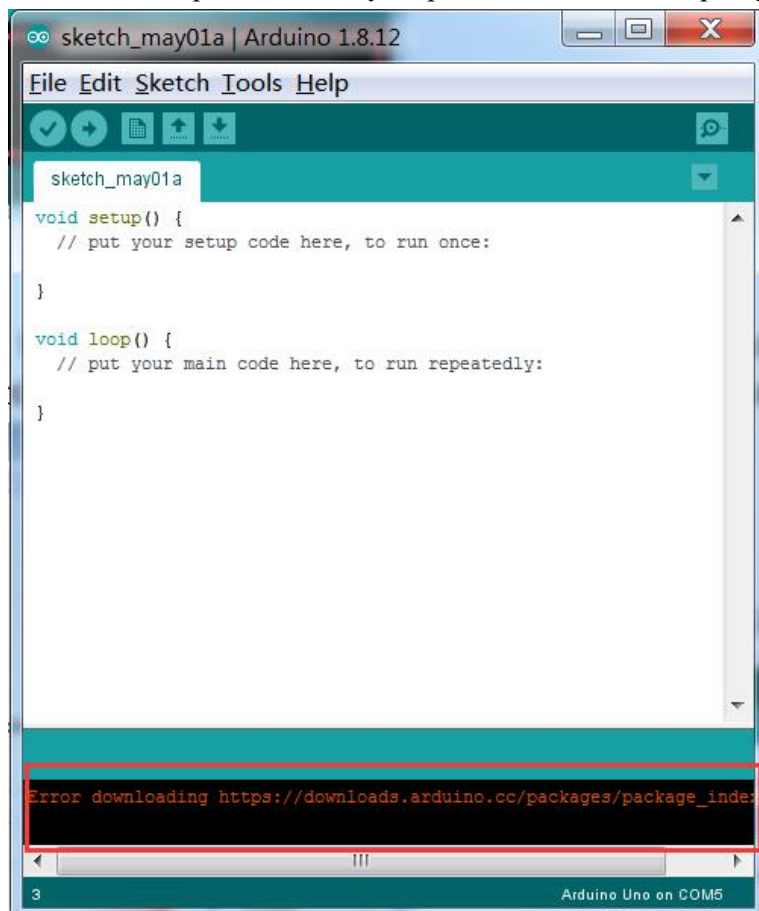


The sketch writing area, as shown below:

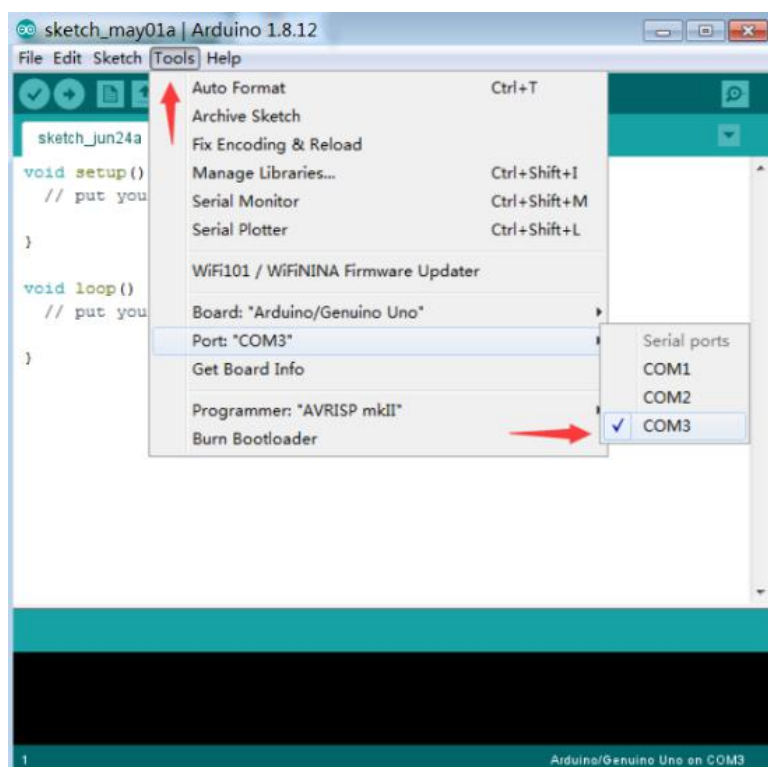
(Programming Learning Website: <https://www.arduino.cc/reference/en/>)



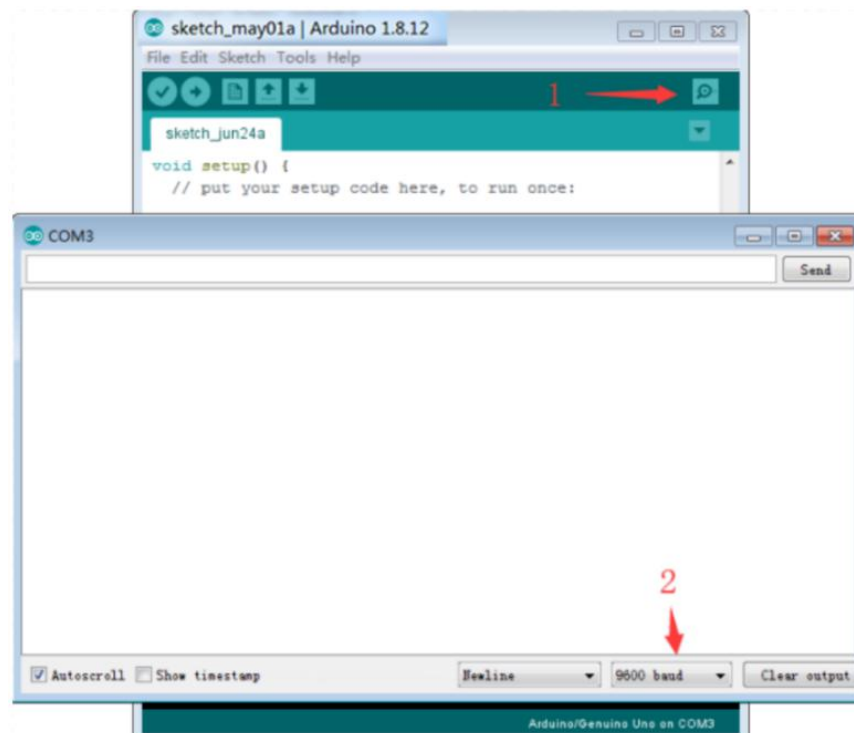
Information output area, mainly output information of compiling and uploading sketch.



Serial monitor, you can open the serial monitor after selecting the port corresponding to the board in "tools"---->"Port". The V1 board selects the COM3 port and sets the baud rate to 9600.

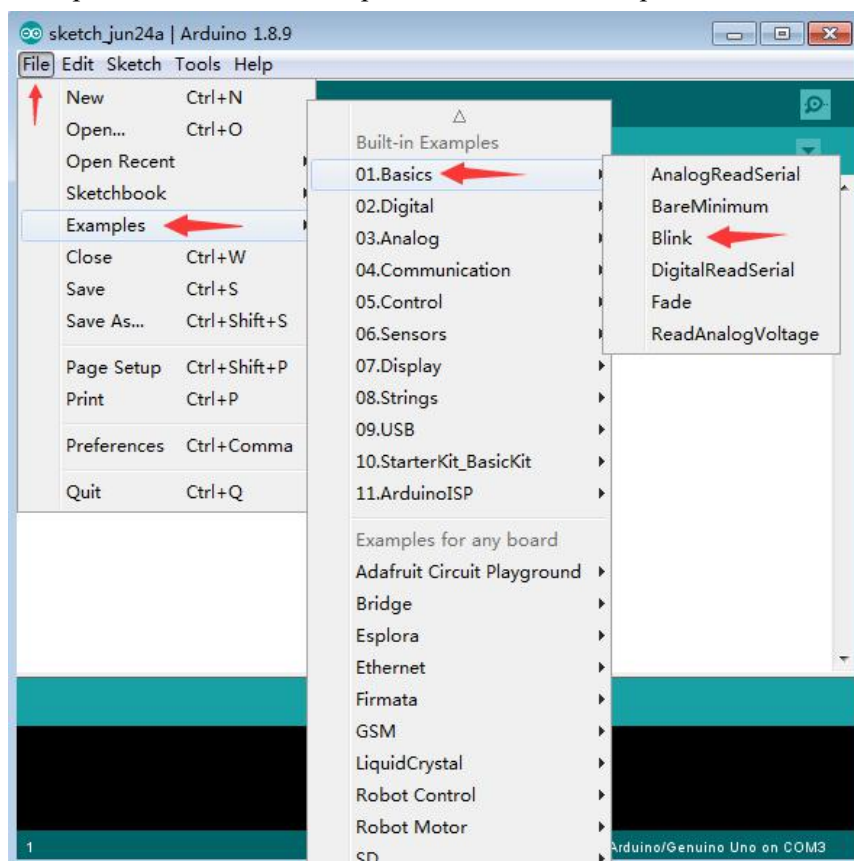


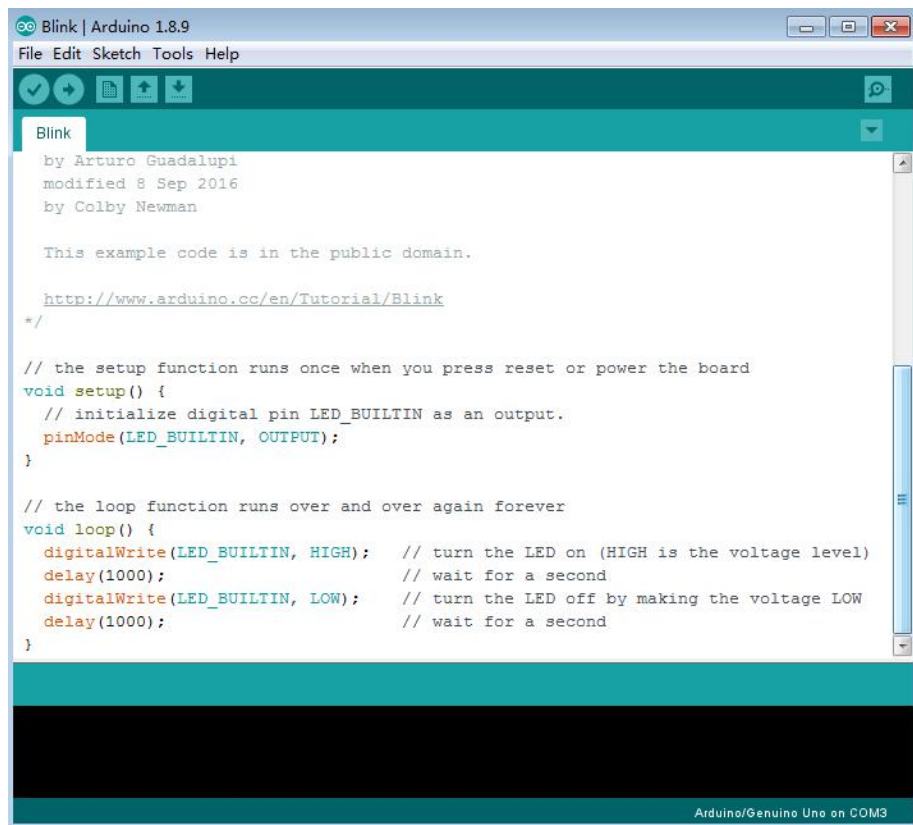




## 5.Upload your first sketch

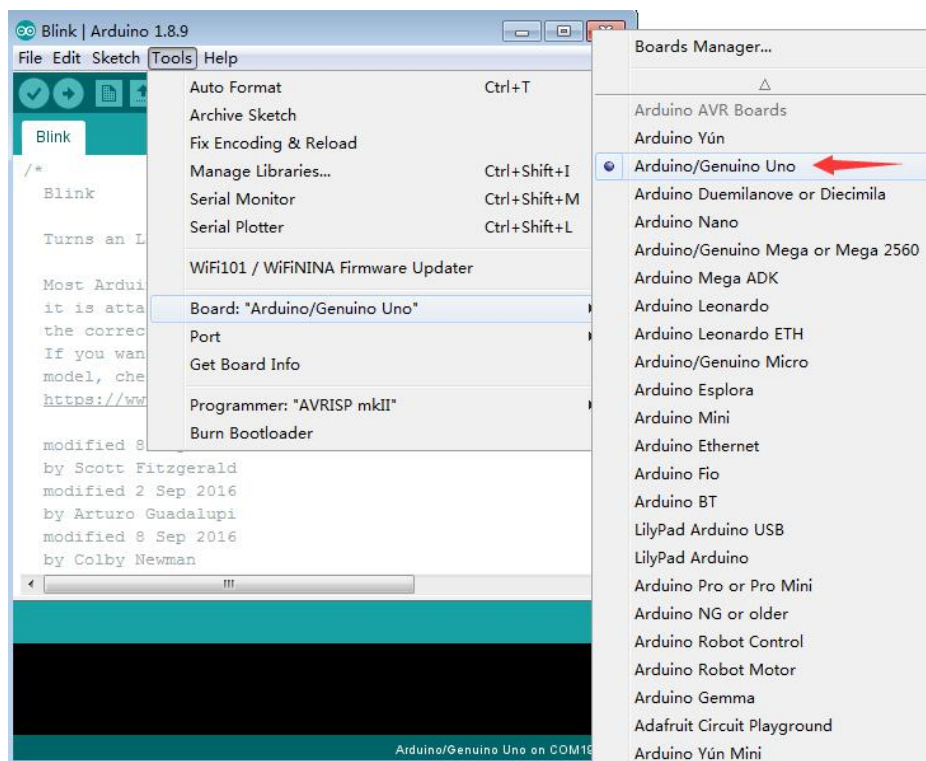
5.1 Open the LED blink example sketch: File > Examples >01.Basics > Blink.



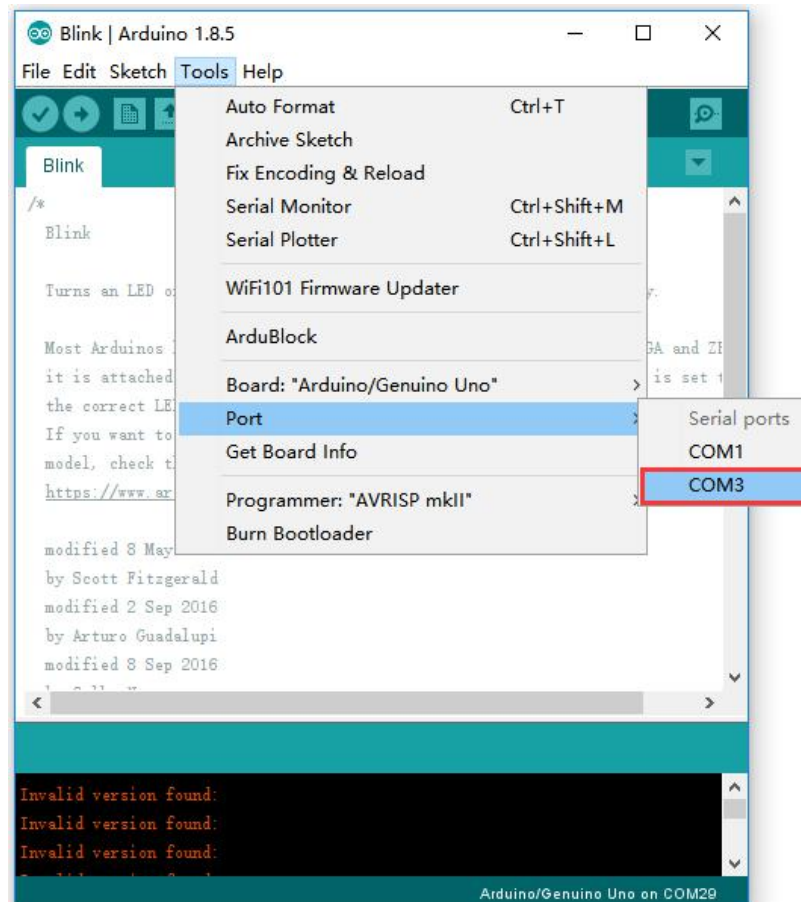


## 5.2 Select your board type and port

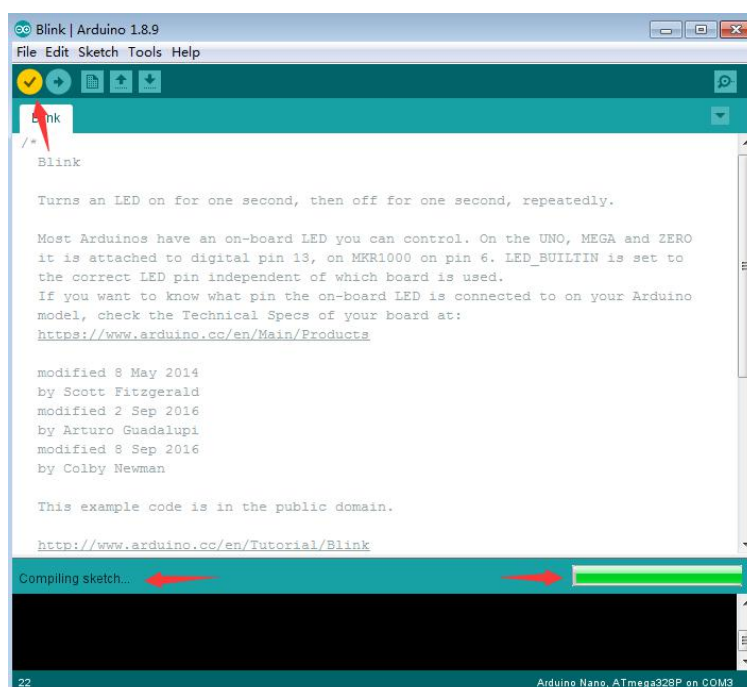
1) Click “tools”--->“Board”--->“Arduino/Genuino UNO”



2) Select the serial device of the board from the Tools | Serial Port menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your board and re-open the menu; the entry that disappears should be the port you should select. Reconnect the board and select that serial port.

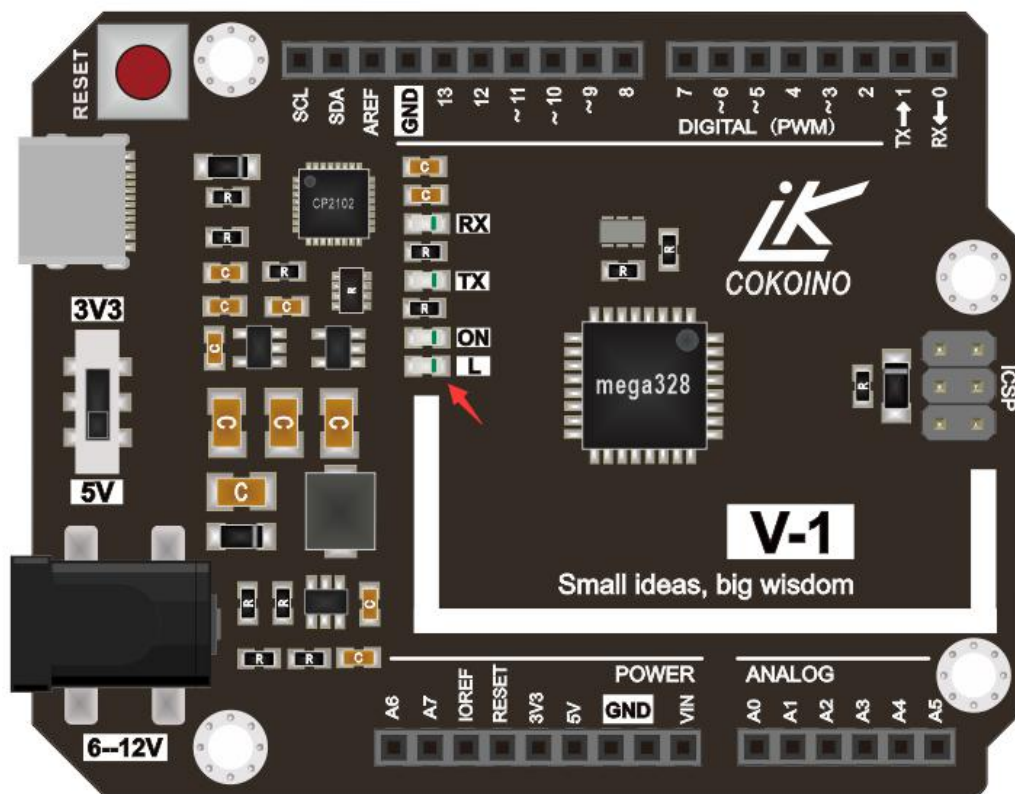


5.3 Now, simply click the "Upload" button. Wait a few seconds - you should see the RX and TX leds on the board flashing.



---

5.4 Result: The LED on V1 board which marked L flashes every 1 second, as shown below



## 6.Arduino library file

### What is the arduino library file

Libraries are files written in C or C++ (.c, .cpp) which provide your sketches with extra functionality (e.g. the ability to control an LED matrix, or read an encoder, etc.).

To use an existing library in a sketch simply go to the Sketch menu, choose "Import Library", and pick from the libraries available. This will insert an `#include` statement at the top of the sketch for each header (.h) file in the library's folder. These statements make the public functions and constants defined by the library available to your sketch. They also signal the Arduino environment to link that library's code with your sketch when it is compiled or uploaded.

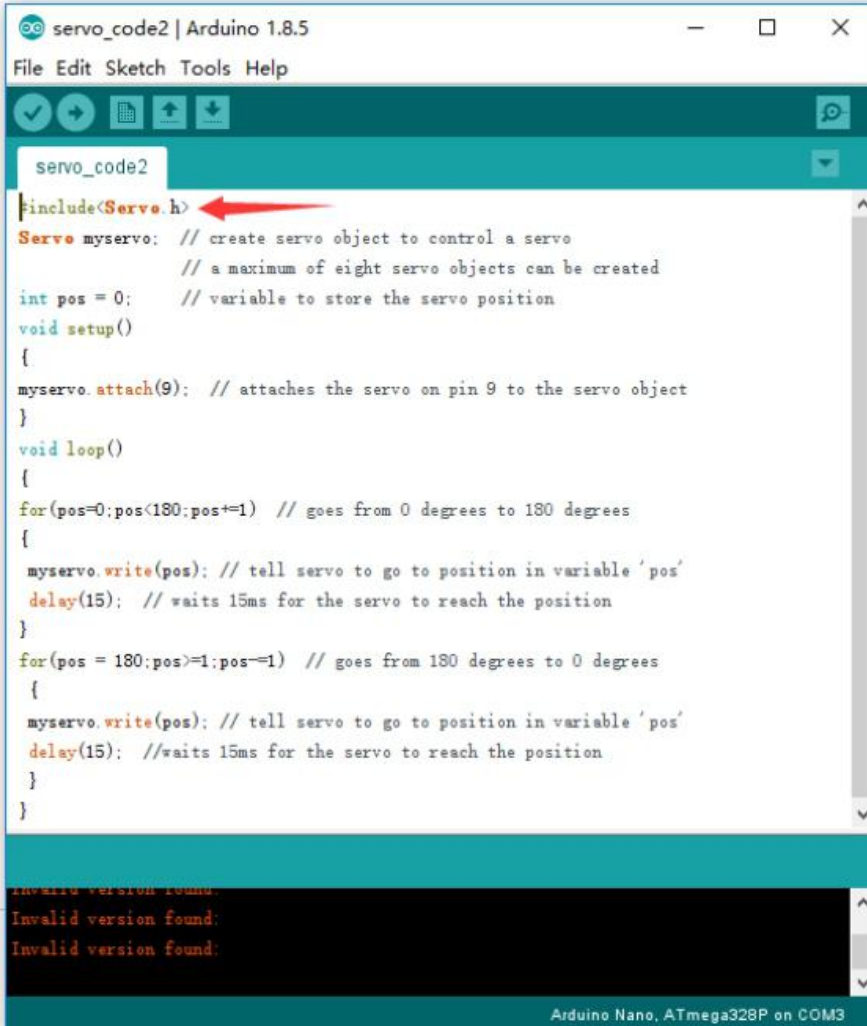
To install your own library, create a folder inside `ARDUINO/hardware/libraries` with the name of your library. The folder should contain a C or C++ file with your code and a header

---

file with your function and variable declarations. It will then appear in the Sketch | Import Library menu in the Arduino IDE.

Because libraries are uploaded to the board with your sketch, they increase the amount of space used by the ATmega8 on the board. See the FAQ for an explanation of various memory limitations and tips on reducing program size. If a sketch no longer needs a library, simply delete its `#include` statements from the top of your code. This will stop the Arduino IDE from linking the library with your sketch and decrease the amount of space used on the Arduino board.

**The following figure shows the library function of the servo motor:**



```
servo_code2 | Arduino 1.8.5
File Edit Sketch Tools Help

servo_code2
#include<Servo.h>
Servo myservo; // create servo object to control a servo
               // a maximum of eight servo objects can be created
int pos = 0;    // variable to store the servo position
void setup()
{
  myservo.attach(9); // attaches the servo on pin 9 to the servo object
}
void loop()
{
  for(pos=0;pos<180;pos+=1) // goes from 0 degrees to 180 degrees
  {
    myservo.write(pos); // tell servo to go to position in variable 'pos'
    delay(15); // waits 15ms for the servo to reach the position
  }
  for(pos = 180;pos>=1;pos-=1) // goes from 180 degrees to 0 degrees
  {
    myservo.write(pos); // tell servo to go to position in variable 'pos'
    delay(15); //waits 15ms for the servo to reach the position
  }
}

Invalid version found:
Invalid version found:
Invalid version found:

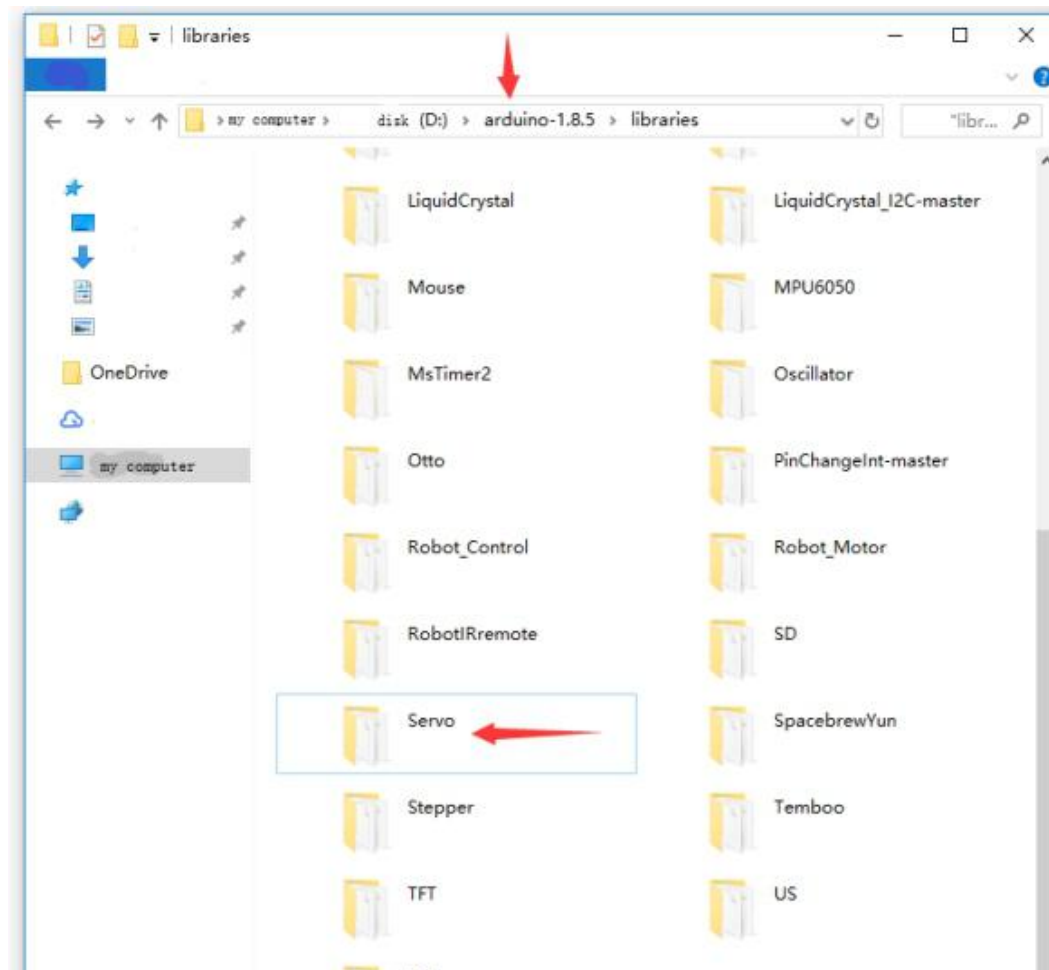
Arduino Nano, ATmega328P on COM3
```

### Where is the library file placed?

The library file is a combination of some header files ending in ".h" and source files ending in ".c" in a folder.

For information on how to write arduino library files, please learn about it in the Google or arduino community.

We only need to put the existing or self-written library files in the libraries folder of the Arduino IDE installation package, then the arduino ide will run it, otherwise arduino ide will report an error when compiling the program. The following figure shows the placement path of my computer arduino library files:



Remarks: For more information on arduino, please refer to the official website below.  
<https://www.arduino.cc/en/Reference/Libraries>