



Adeept

Adeept New Ultimate Starter Learning Kit for Arduino UNO R3

Sharing Perfects Innovation



STEM



python

www.adeept.com

Contents

Arduino.....	3
Building the Arduino development environment.....	12
Arduino graphical programming.....	28
Lesson 1 How to Light the Blinking LED.....	50
Lesson 2 The Application of the Active Buzzer.....	75
Lesson 3 Controlling LED with Button.....	96
Lesson 4 Controlling LED with Relay Module.....	116
Lesson 5 Controlling LED with Potentiometer.....	133
Lesson 6 Making a Flowing LED with LED.....	149
Lesson 7 Controlling LED Bar with Potentiometer.....	164
Lesson 8 Making Breathing LED with LED.....	180
Lesson 9 The Application of the RGB LED.....	199
Lesson 10 The Application of the Passive Buzzer.....	216
Lesson 11 The Application of the LCD1602 and the IIC Interface.....	233
Lesson 12 Making Voltmeter.....	250
Lesson 13 The Application of the 7-segment Display.....	265
Lesson 14 Making A Simple Counter.....	282
Lesson 15 Controlling the Servo Motor.....	299
Lesson 16 Making a Digital Thermometer.....	316
Lesson 17 The Application of IR Remoter Controller.....	334
Lesson 18 The Application of DHT-11(Digital Temperature & Humidity Sensor).....	350
Lesson 19 The Application of the Ultrasonic Distance Sensor.....	369
Lesson 20 Application of MPU6050.....	385
Lesson 21 The Application of the 4*4 Matrix Keyboard.....	410
Lesson 22 Controlling the DC Motor.....	427
Lesson 23 The Application of the PS2 Joystick Module.....	441
Lesson 24 Controlling LED with Tilt Switch.....	457
Lesson 25 The Application of the 8*8 Dot-matrix Module.....	471
Lesson 26 Controlling the Stepper Motor.....	487
Lesson 27 Using Photoresistor to Measuring Light intensity.....	501
Lesson 28 Making Light Tracking System.....	517
Lesson 29 Making Frequency Meter.....	532
Lesson 30 The Application of the PIR Movement Sensor.....	549
Lesson 31 Using IR Remoter Controller to control Relay.....	565
Lesson 32 Controlling RGB LED with IR Remoter Controller.....	579
Lesson 33 Controlling Stepper Motor with IR Remoter Controller.....	595
Lesson 34 Controlling the Size of a Circle by Potentiometer.....	610
Lesson 35 Controlling the 3D Model by PS2 Joystick.....	619
Lesson 36 Snake Game.....	622
Lesson 37 Star Wars.....	625

Arduino

1. Arduino

Arduino is an easy-to-use open source electronic prototype platform. The name Arduino is derived from the inherent, originally intended as "strong friends". It was born in 2005 in Italy Interaction Design Institute (Interaction Design Institute Ivrea). One of the main founders of the project is called Massimo Banzi.

The original intention of the Arduino project was to lower the threshold for students in the Interactive Design Institute to create interactive works. In addition to reducing the cost of students, it also takes into account the difficulty of learning. That is to say, it only takes a little money and requires a very small amount of programming knowledge to create different works according to the unexpected creativity.

In the spirit of open source: for the benefit of more enthusiasts, from the beginning of the creation of the Arduino project, it has complied with the agreement of open source hardware. Anyone not only is allowed to get the design drawings of the Arduino hardware circuit board for free, but also do mass produce according to their own design process.

In recent years, with the rise of smart hardware, a wave called Maker has emerged all over the world. They use Arduino and various hardware modules around it to quickly build a variety of fun products. Such as personalized air quality detectors, fire alarms. People who love cats make smart cat feeders. Some people make all kinds of weird small robots. Flying enthusiasts design various aircrafts to complete a variety of thrilling aerial photography.

Next let us learn about Arduino!

2.Features of Arduino

At present, there are many other single-chip and single-chip platforms on the market, such as 51 single-chip and STM32 single-chip. However, they have relatively high thresholds for ordinary developers. They not only ask for certain basic knowledge of programming and hardware, but also the internal registers of them are more complicated.

Arduino not only simplifies the process of working with a microcontroller, but also provides teachers, students and hobbyists with some advantages and features that other systems do not have:

Openness: Arduino is an early open source hardware project. Its various open source projects have been widely recognized and widely used. Its hardware circuit and software development environment are completely open, and anyone can use, modify, and share it without engaging in commercial use. This not only allows users to better understand the circuit principle of Arduino, but also can modify according to their needs. For example, due to space constraints, we need to design a special-shaped circuit board, or design our own expansion circuit and main control circuit to connect together.

Ease of use: for those who are interested, regardless of the basics knowledge, however within one hour they can get to run the first simple program on Arduino. The connection between Arduino and PC uses the most popular USB connection. You can connect the Arduino directly to the computer just like using a smart phone without installing any additional drivers. And the development environment software of Arduino is also very simple. The clear menu only provides the necessary toolbar, eliminating all the elements that may dazzle beginners. You can even compile and download routines without reading the tutorial.

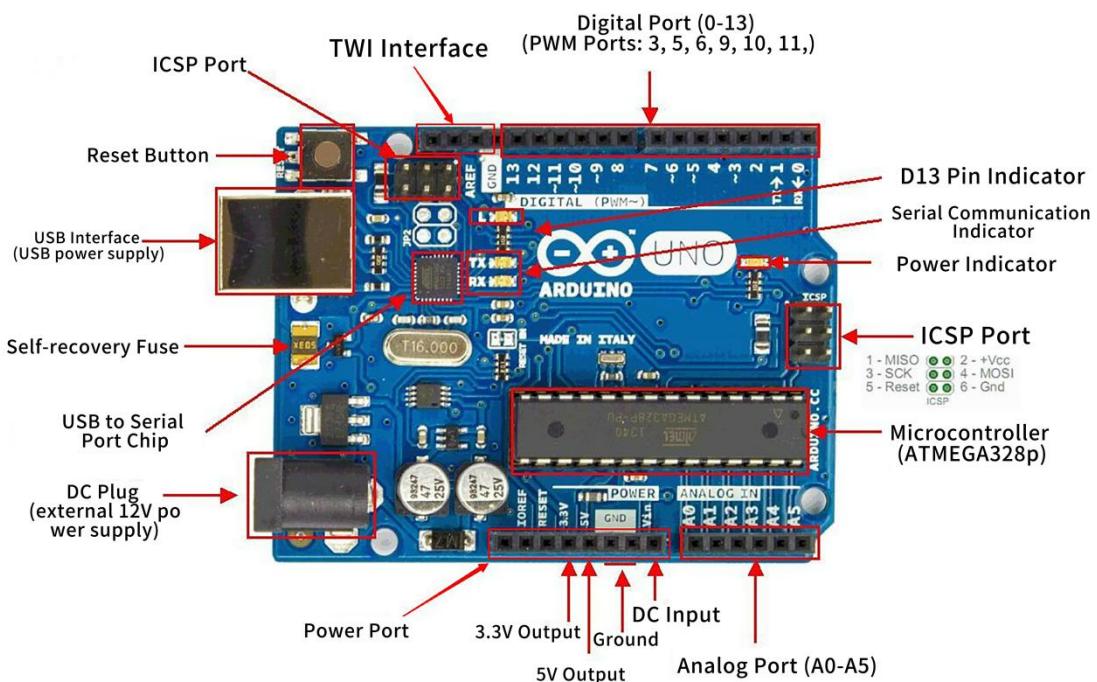
Communication: For beginners, communication and display are very effective ways to stimulate learning enthusiasm. But sometimes, for beginners who do not have a particularly deep understanding of single chip microcomputer, I am afraid it will be

difficult to communicate between them. But Arduino has defined a relatively unified framework, and some of the underlying initialization has adopted a unified method, and the ports used for digital and analog signals are also calibrated, which is helpful for beginners to communicate.

Rich third-party resources: Arduino hardware and software are all open source, you can have a deep understanding of all the underlying mechanisms, and it also reserves a very friendly third-party library development interface. Adhering to the openness and sharing of the open source community, many enthusiasts will bring their own hardware and software to share with you after successfully implementing their own designs. For latecomers, you can easily find some basic function modules you want to use in the Arduino community, such as servo control, PID speed regulation, A/D conversion, etc. Some function module suppliers are also paying more and more attention to the Arduino community. They provide libraries and related tutorials for their products under Arduino. These are extremely convenient for Arduino developers. You can focus on the functional design that you want to do without having to stick to the writing of basic functions.

3. Introduction of Arduino Development Board

Since the development of Arduino, various models and many derivative controllers have been introduced, including Arduino Uno, Arduino Nano, Arduino Yún, etc. Our course uses the Arduino Uno R3 development board. As shown below:



The following is a detailed introduction to the development board of Arduino Uno R3 model:

(1) Power port:

Arduino UNO has three power supply methods:

1. Power supply via USB interface. The voltage is 5 V;
2. Power is supplied with the DC power input interface, and the voltage requires 7 to 12 V;
3. Power is supplied with the 5 V or VIN port at the power supply interface. The power supply at the 5 V port must be 5 V, and the power supply at the VIN port is 7 to 12 V.
4. IOREF: Provide working reference voltage for the expansion board.
5. RESET: reset pin. Restart the Arduino and start the program from the beginning.
6. GND: Ground terminal.
7. Vin: provides external power supply voltage, which can be directly connected to external power supply by this port.

(2) Analog port (A0~A5):

Arduino Uno has 6 analog pins. Pins A0-A5 can read analog voltages. They are used as ADCs (analog-to-digital converters). ADCs convert the voltages into bits that the microprocessor can understand. These pins are used as analog inputs, but they can also be used as digital inputs or digital outputs. They are often used to read sensor data.

(3) Digital port (0~13):

Pins 0 to 13 of Arduino Uno are used as digital input/output pins. Among them, pin 13 is connected to the on-board LED indicator (D13 indicator); digital pins can be turned on or off. When they are on, they are at a high level of 5V. When they are off, they are at a low level of 0V status. On the Arduino, when the digital pins are configured as outputs, they are set to 0 or 5V. When the digital pin is configured as an input, the voltage is provided by an external device. This voltage can be varied between 0-5V and converted to a digital 0 or 1 to indicate low or high level. To determine this, there are 2 thresholds:

1. If it is lower than 0.8v, it is regarded as 0.
2. If it is higher than 2.0v, it is regarded as 1.

When connecting components to digital pins, make sure that the logic levels match. If the voltage is between the thresholds, the return value will be undefined.

Pins 3, 5, 6, 9, 10, 11 have PWM function.

Pins 10, 11, 12, and 13 can be used for SPI communication.

0 and 1 are Rx and Tx pins. These two pins are generally used as serial ports. Non-serial devices should not occupy this pin as much as possible. (Rx send, Tx receive).

Pin 2 and 3 can input an external interrupt signal.

A4 (SDA), A5 (SCL) and TWI interface can be used for TWI communication, being compatible with I²C communication. You can use the official Wire library to

operate.

The AERF pin can be used for analog input reference voltage.

【Attention】 :

1. Each pin can provide/receive up to 40mA of current. But the recommended current is 20mA.
2. The absolute maximum current provided by all pins is 200mA.

(4) ICSP port:

The ICSP (In-Circuit Serial Programming) port is an online program programming port, which is a 2*3 pin header on the Arduino development board, and its 6 pins are connected to the microcontroller on the Arduino development board , corresponding to 5V, MISO, MOSI, SCK, GND and RESET respectively. These pins enable users to program the firmware on the Arduino development board. Generally, when programming a program, the ICSP port close to the microcontroller is used.

(5) Indicator light:

Arduino Uno has 4 LED indicators. The functions are as follows:

1. ON: power indicator. When the Arduino is powered on, the ON light will be on.
2. TX: serial port sending indicator. When the USB is connected to the computer and the Arduino transmits data to the computer, the TX light will be on.
3. RX: serial port receiving indicator. When using USB to connect to the computer and the Arduino receives the data from the computer, the RX light will be lit up.
4. L: D13 pin indicator light, also known as programmable control indicator light. The LED is connected to pin 13 of the Arduino by a special circuit. When the pin 13 is high or high impedance, the LED will be lit; when it is low, it will not be lit. The LED can be controlled to be on or off by a program or an external input signal.

(6) DC external plug:

Using an external power supply of 7-12V to power the Arduino.

(7) USB interface:

It is used to connect the USB interface of the computer to achieve serial communication, uploading programs, and serial port monitoring between the Arduino and the computer.

(8) Reset button:

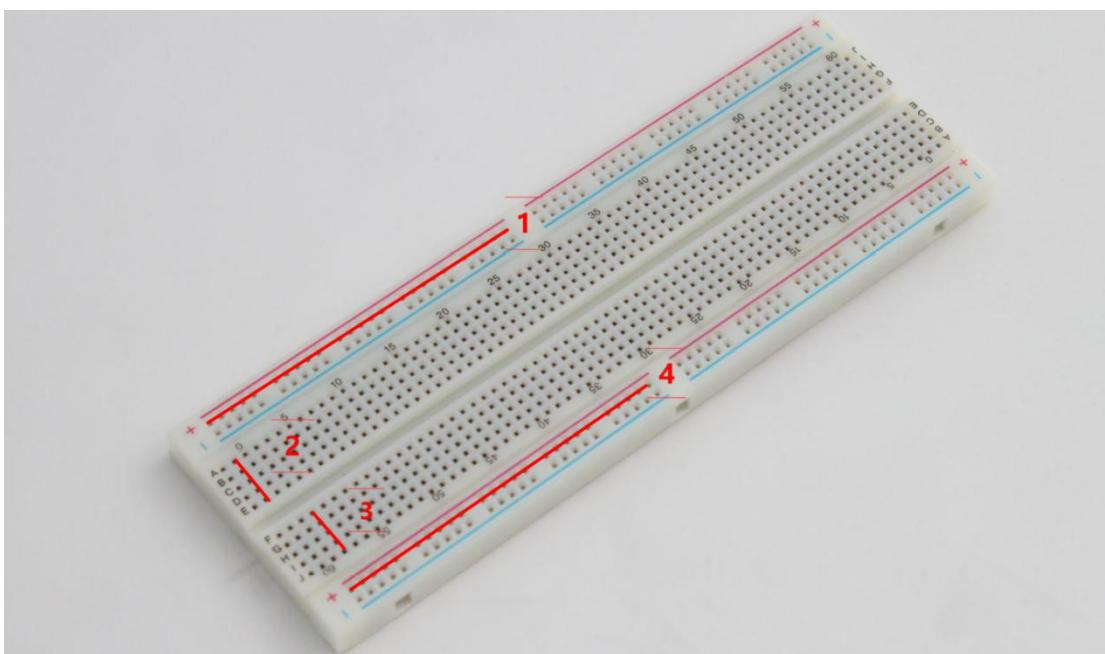
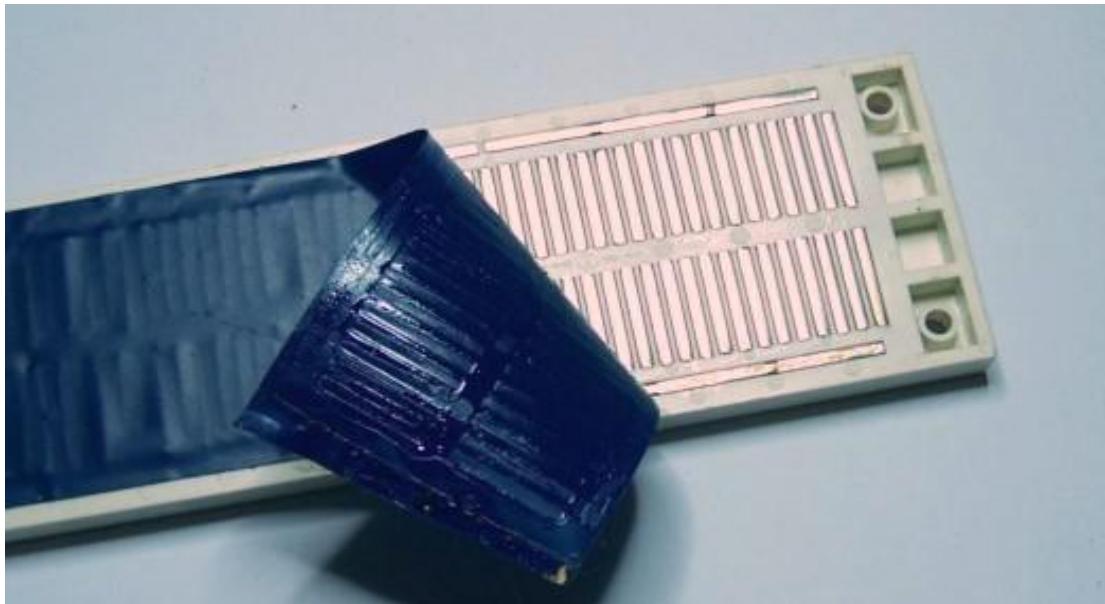
Restart the Arduino and start the program from the beginning.

(9) Microcontroller:

The "brain" of Arduino is based on the smallest system of ATmega328P microcontroller.

4. the use of Breadboard (breadboard) in the circuit

Breadboard is a commonly used plug-in board with porous sockets in circuit experiments. When conducting circuit experiments, you can insert the pins and wires of electronic components into the corresponding holes according to the circuit connection requirements to make them contact with the elastic contact spring in the hole, thereby connecting into the required experimental circuit.



The internal circuit connectivity of the breadboard:

- (1) In area 1 of the figure, only five holes from left to right are connected, and the red line is drawn as shown in the figure. The upper and lower holes are not connected.
- (2) In area 2 in the figure, only five holes from top to bottom are connected, and the red line is drawn as shown in the figure. Left and right are not connected.
- (3) In area 3, only five holes from top to bottom are connected, and the red line is drawn as shown. Left and right are not connected.
- (4) In area 4 in the figure, only the five holes from left to right are connected, and

the red line is drawn as shown in the figure. The upper and lower holes are not connected.

【Attention】

Zone 1, zone 2, zone 3 and zone 4 are not connected to each other.

Building the Arduino development environment

1. Arduino development language

Arduino uses C/C++ to write programs, so before learning Arduino, you need to master the C/C++ language. Although C++ is compatible with the C language, these are two different languages. C is a process-oriented programming language, and C++ is an object-oriented programming language. The early Arduino core library was written in C language. Later, object-oriented ideas were introduced. At present, the latest Arduino core library is written in C and C++.

Generally speaking, the Arduino language refers to a collection of various Application Programming Interfaces (APIs) provided by the Arduino core library files. These APIs are formed by secondary packaging of the lower-level microcontroller support library. For example, the core library of Arduino using AVR microcontroller is the secondary packaging of AVR-LIBC (GCC-based AVR support library).

In the traditional development method, multiple registers need to be configured to achieve the corresponding functions. In Arduino, the complicated registers are encapsulated into simple APIs, which can be intuitively controlled, enhancing the readability of the program and improving the development efficiency.

2. Arduino program structure

The Arduino program structure is different from the traditional C/C++ program structure-there is no main() function in the Arduino program. In fact, it is not that there is no main() function in the Arduino program, but that the definition of the main() function is hidden in the core library file of the Arduino. In the development of Arduino, the main function is not directly operated, but the two functions of setup() and loop() are used instead.

3. The construction of the Arduino development environment

The IDE of the Arduino development environment can be downloaded from the official website. The download address of the Arduino IDE is:
<https://store.arduino.cc/usa/>

(1) Install Arduino IDE under Windows

We will teach you how to download and install:

1. Open Google Chrome and enter the URL in the address bar:
<https://store.arduino.cc/usa/>

After successfully opening the interface as shown below, we click DOWNLOADS under SOFTWARE.



2. After jumping to the following interface, slide the mouse to the middle to find the part marked in the red circle. You can find that the official website provides us with installation files for Windows, Mac OS X, and Linux systems.



Download the Arduino IDE



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 \(sha512\)](#)

3. We click the installation package of Windows ZIP file for non admin install. After the interface jumps, we select JUST DOWNLOAD. And then start the download. The download status will be displayed in the lower left of Google Chrome. Then we wait for the download to complete.

Download the Arduino IDE



ARDUINO 1.8.12

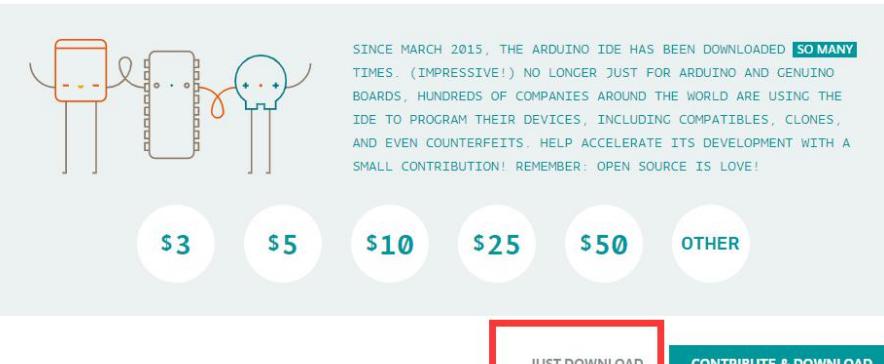
The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is

Windows Installer for Windows 7 and up
Windows ZIP file for non admin install

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

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.



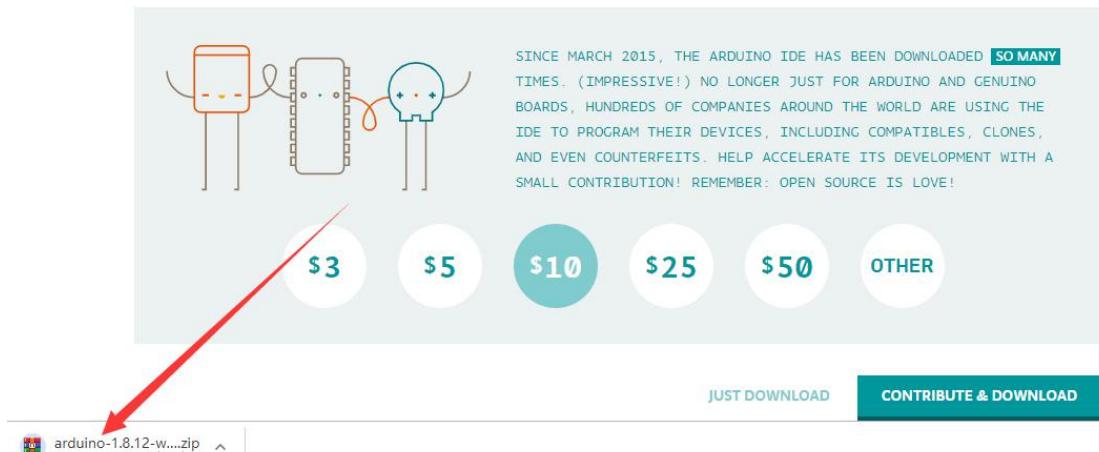
SINCE MARCH 2015, THE ARDUINO IDE HAS BEEN DOWNLOADED [SO MANY](#) TIMES. (IMPRESSIVE!) NO LONGER JUST FOR ARDUINO AND GENUINO BOARDS, HUNDREDS OF COMPANIES AROUND THE WORLD ARE USING THE IDE TO PROGRAM THEIR DEVICES, INCLUDING COMPATIBLES, CLONES, AND EVEN COUNTERFEITS. HELP ACCELERATE ITS DEVELOPMENT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

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

[JUST DOWNLOAD](#) [CONTRIBUTE & DOWNLOAD](#)

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.](#)



 arduino-1.8.12-w...zip 已下载 1.8/193 MB, 还需 1...

4. After the download is complete, open the folder. There are downloaded compressed installation files:

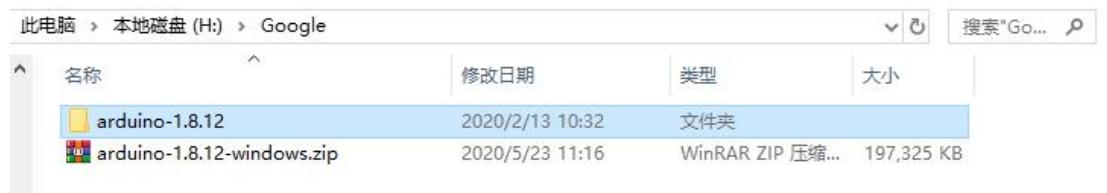
arduino-1.8.12-windows.zip



5. Double-click to open the file and unzip it.



6. The file arduino-1.8.12 appears after decompression. As shown follows;

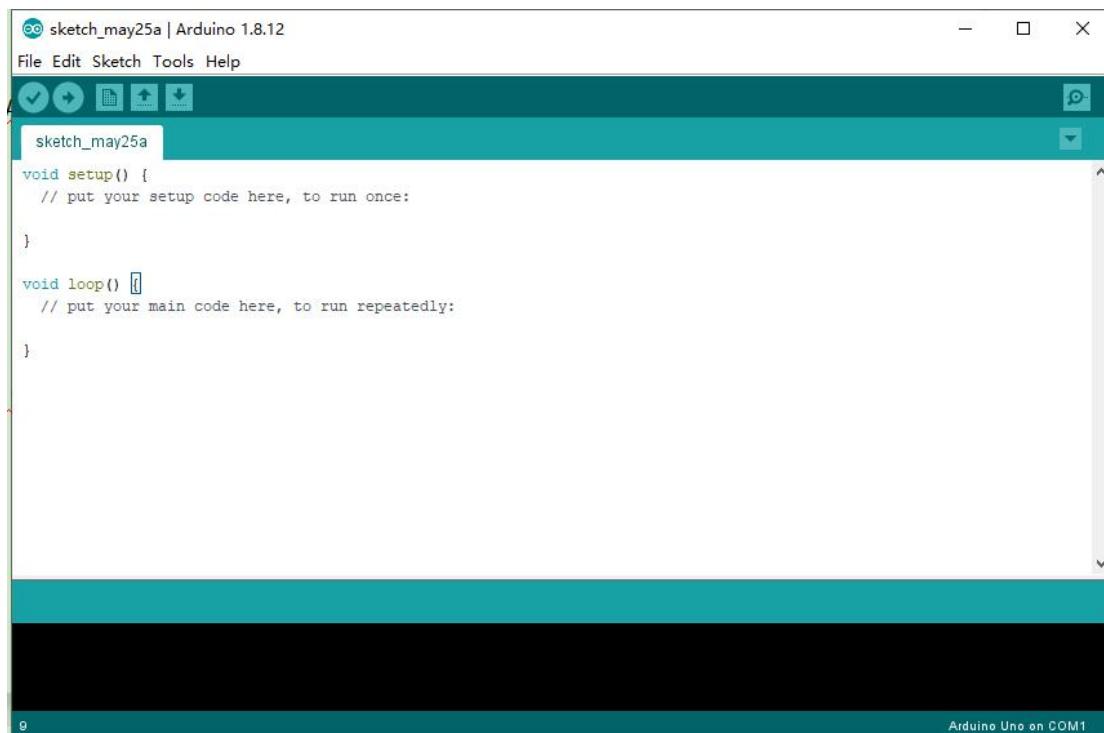


7. Open the arduino-1.8.12 folder and double-click arduino.exe to open the

software.

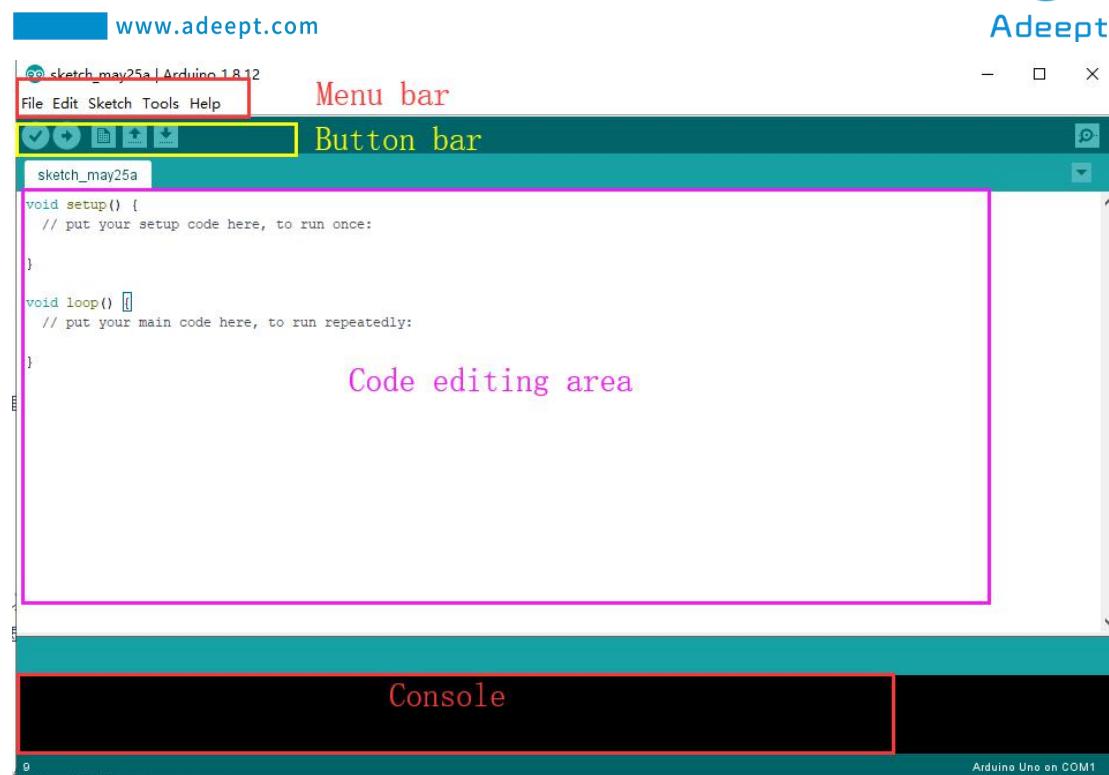
名称	修改日期	类型	大小
drivers	2020/2/13 10:32	文件夹	
examples	2020/2/13 10:32	文件夹	
hardware	2020/2/13 10:32	文件夹	
java	2020/2/13 10:32	文件夹	
lib	2020/2/13 10:32	文件夹	
libraries	2020/2/13 10:32	文件夹	
reference	2020/2/13 10:32	文件夹	
tools	2020/2/13 10:32	文件夹	
tools_builder	2020/2/13 10:32	文件夹	
arduino.exe	2020/2/13 10:32	应用程序	395 KB
arduino.l4j.ini	2020/2/13 10:32	配置设置	1 KB
arduino_debug.exe	2020/2/13 10:32	应用程序	393 KB
arduino_debug.l4j.ini	2020/2/13 10:32	配置设置	1 KB

8. The interface will show as follows after the Arduino software is opened, indicating that our software has been downloaded and installed successfully.



4. Introduction of Arduino software interface

The following figure is the interface introduction of Arduino software



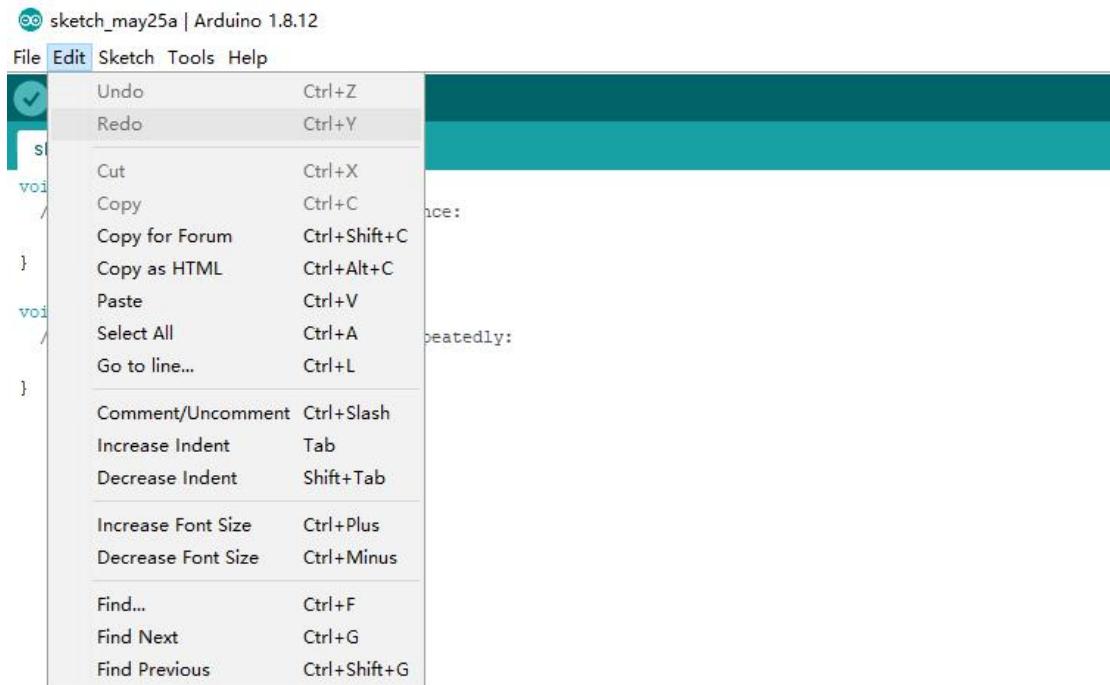
(1)Menu bar

Menu bar contains File, Edit, Sketch, Tools and Help.

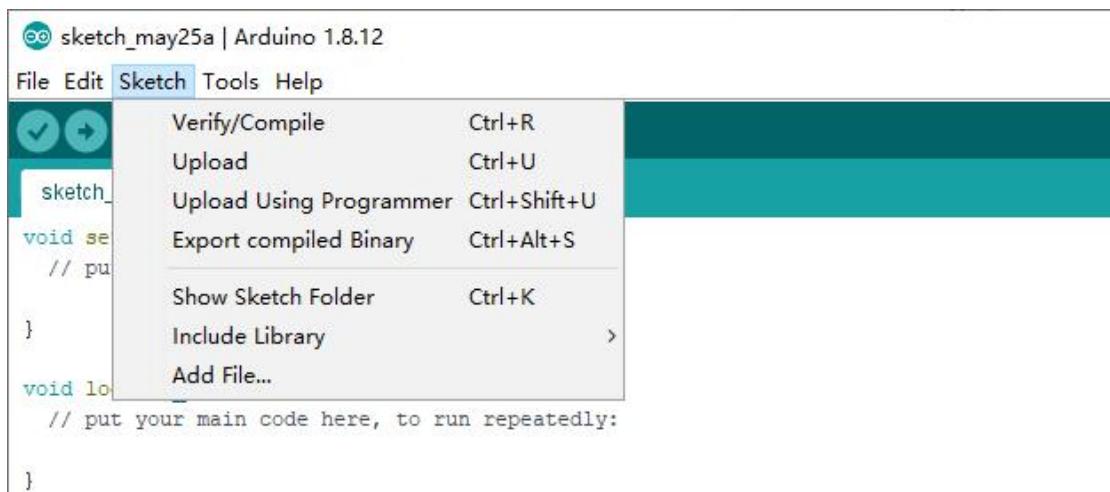
(1) "File" can operate new file, open file, save file, close file, save, etc. For the Examples, you can check the official sample program.



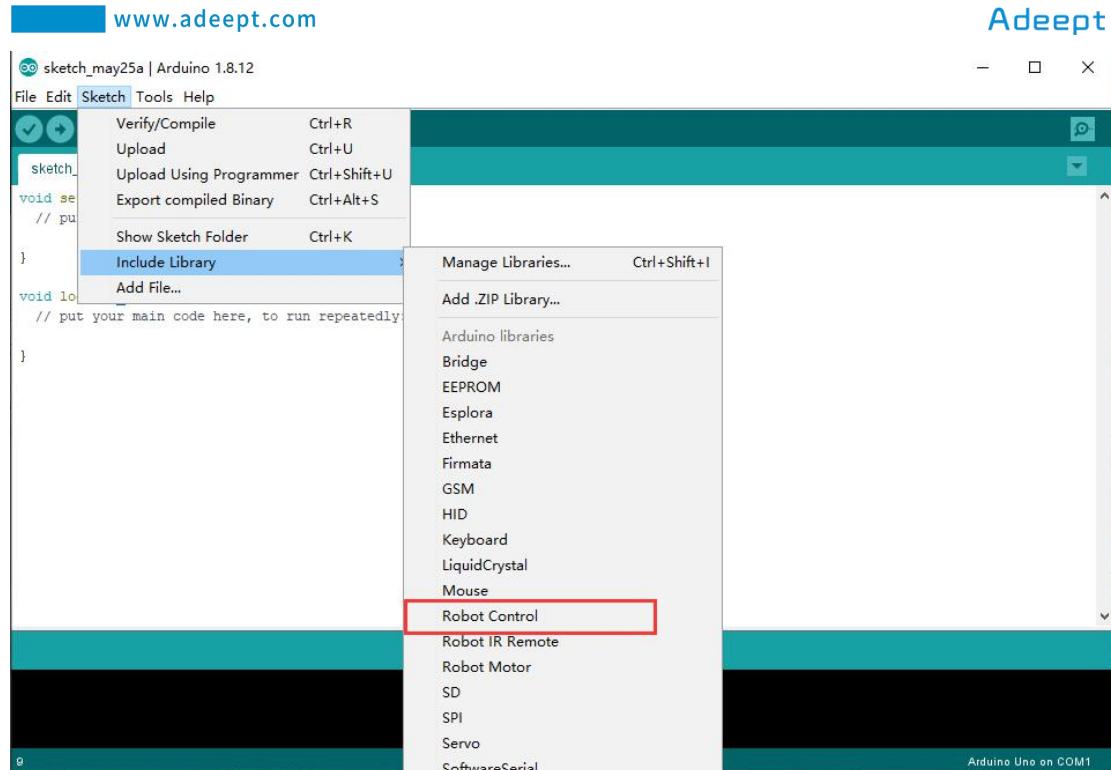
(2) "Edit" has the functions for the program code of editing, copying and pasting, commenting, indenting, searching, etc.



(3) Sketch can perform Verify/Compile, Upload and other operations on the written project.



The Include Library can load the library file. After selecting the library file in the list, the relevant header files are automatically added in the code editing area.



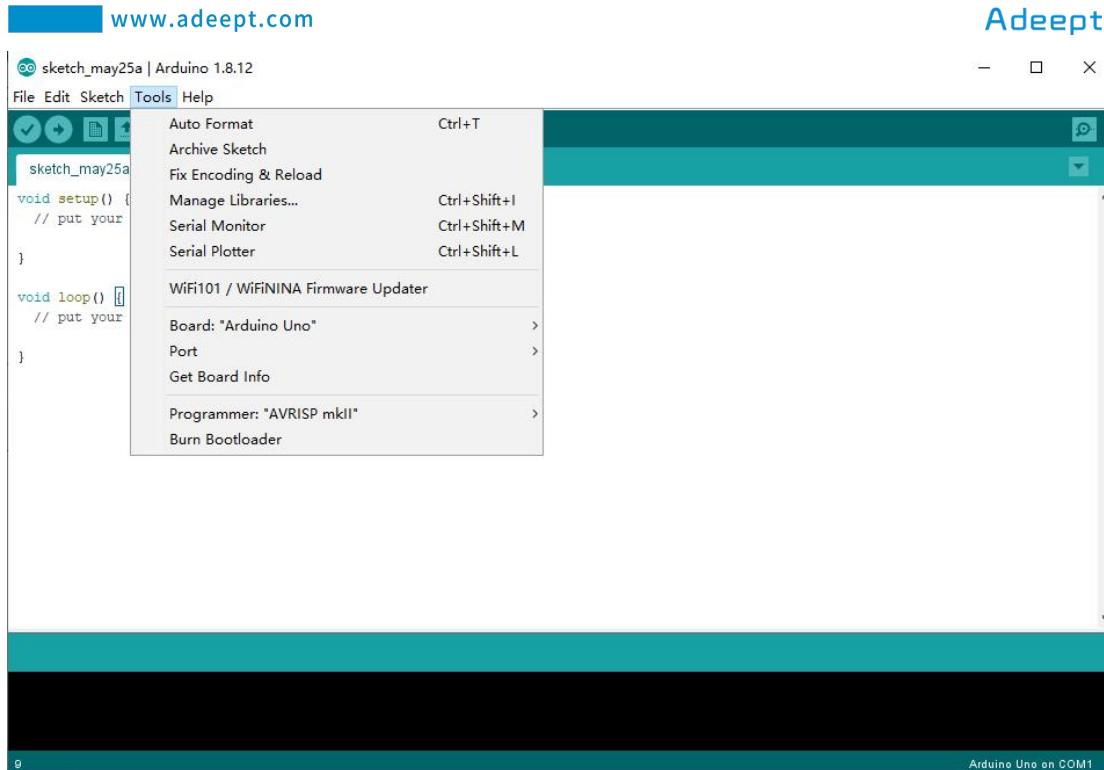

```
#include <ArduinoRobot.h>
#include <Arduino_LCD.h>
#include <Compass.h>
#include <EasyTransfer2.h>
#include <EEPROM_I2C.h>
#include <Fat16.h>
#include <Fat16Config.h>
#include <Fat16mainpage.h>
#include <Fat16util.h>
#include <FatStructs.h>
#include <Multiplexer.h>
#include <SdCard.h>
#include <SdInfo.h>
#include <Squawk.h>
#include <SquawkSD.h>

void setup() {
  // put your setup code here, to run once:
}

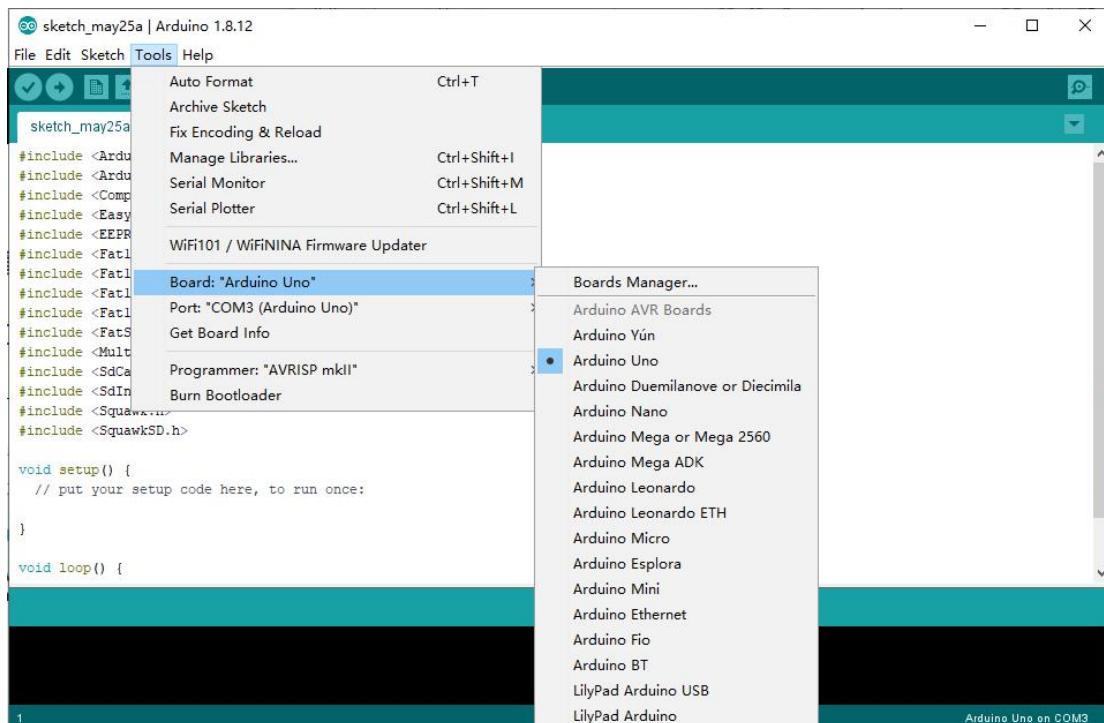

```

The code editor displays the sketch "sketch_may25a". A red box highlights the first several lines of code, which are the #include statements for various Arduino libraries.

(4) Board and Port are often used in "Tools".

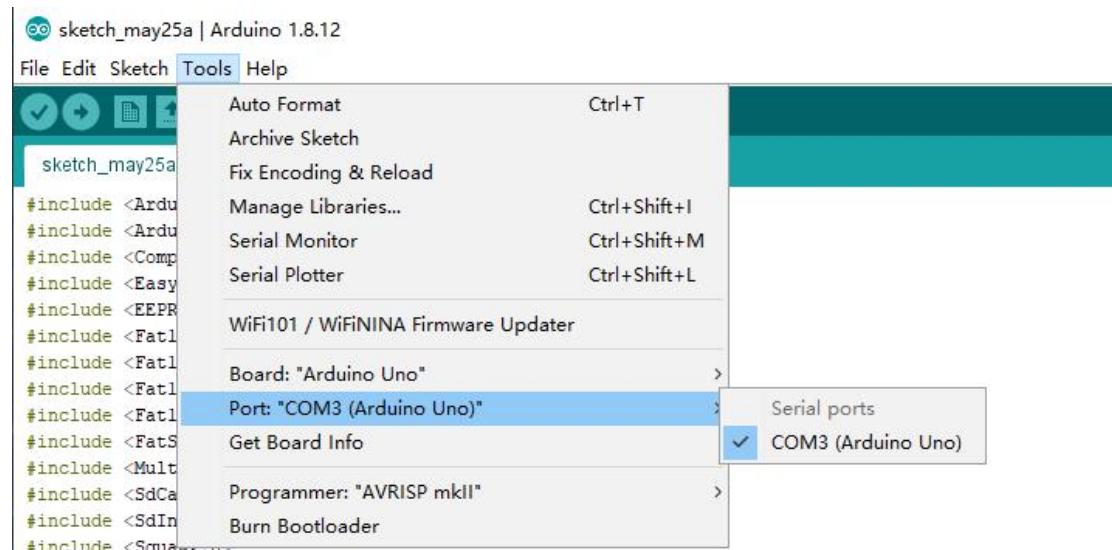


Board can choose different development boards. Our course uses Arduino Uno development board, so we need to choose Arduino Uno. The list contains many Arduino development board models. We choose the corresponding ones according to the model.



Port can set the port used by Arduino IDE to download the program, that is, the

port number of the development board connected to the computer. The port display of each computer is different. When we use the Arduino Uno to connect to the computer, it displays the COM3 port number.



(2) Button bar

Button bar includes functions of Verifying,Uploading,Building New,Opening and Saving.

(1) Verify :

Checking and compilation. This button is used to check the correctness of your "syntax" or code. If your code has any syntax errors or undefined variables, an error message will appear at the bottom of the IDE screen. At the same time, the line of error code will be marked with a red background color for easy modification. But if it is correct, you will see the message that the compilation is complete.

(2) Upload :

Download the program code to the Arduino development board. It is better to click Verify first, and then click Upload.

(3) New :

Open a new program editing window to create a new project.

(4) Open :

This button can open an existing draft file. You will use it when you need to open a file that you have downloaded or used before.

(5) Save :

Save the program file being edited.

(3)Code editing area

The code editing area is where to write program code and code comments.

(4)Console

The debug window will output information showing various compilation and debugging results. For example, if your code is written incorrectly, you will be prompted about what went wrong.

5.Connecting the Arduino development board and the computer

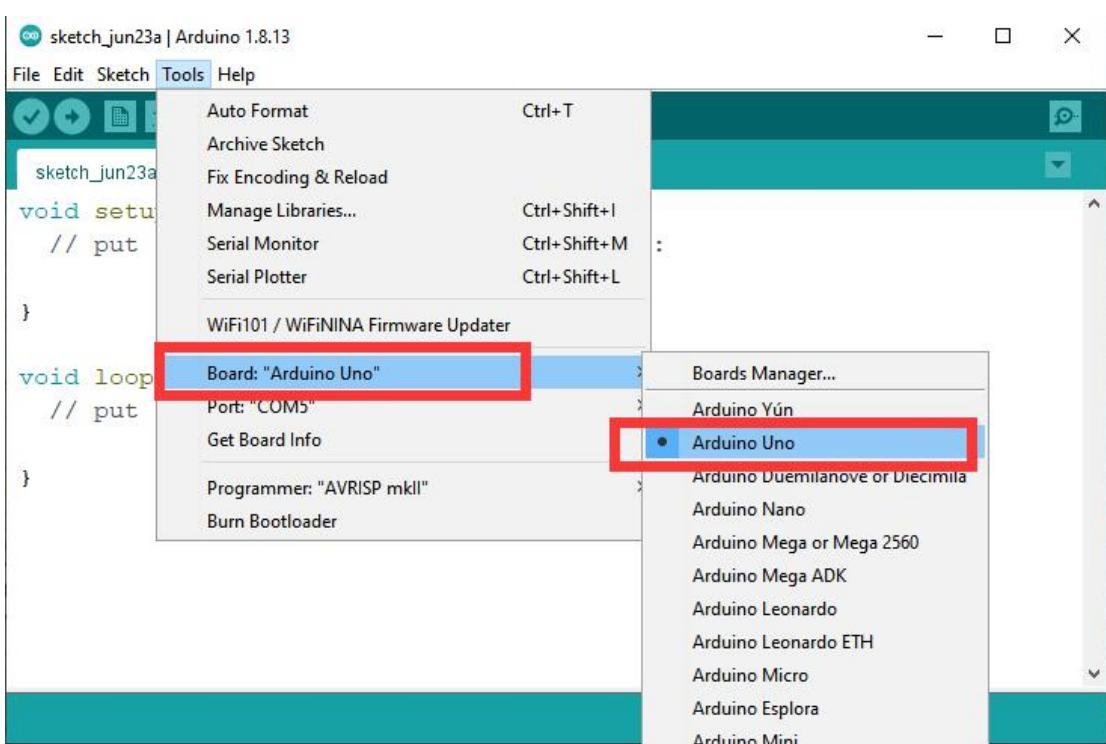
(1)Connecting the Arduino development board and the computer

You need to use USB Cable to connect the Arduino to the computer.As shown below:



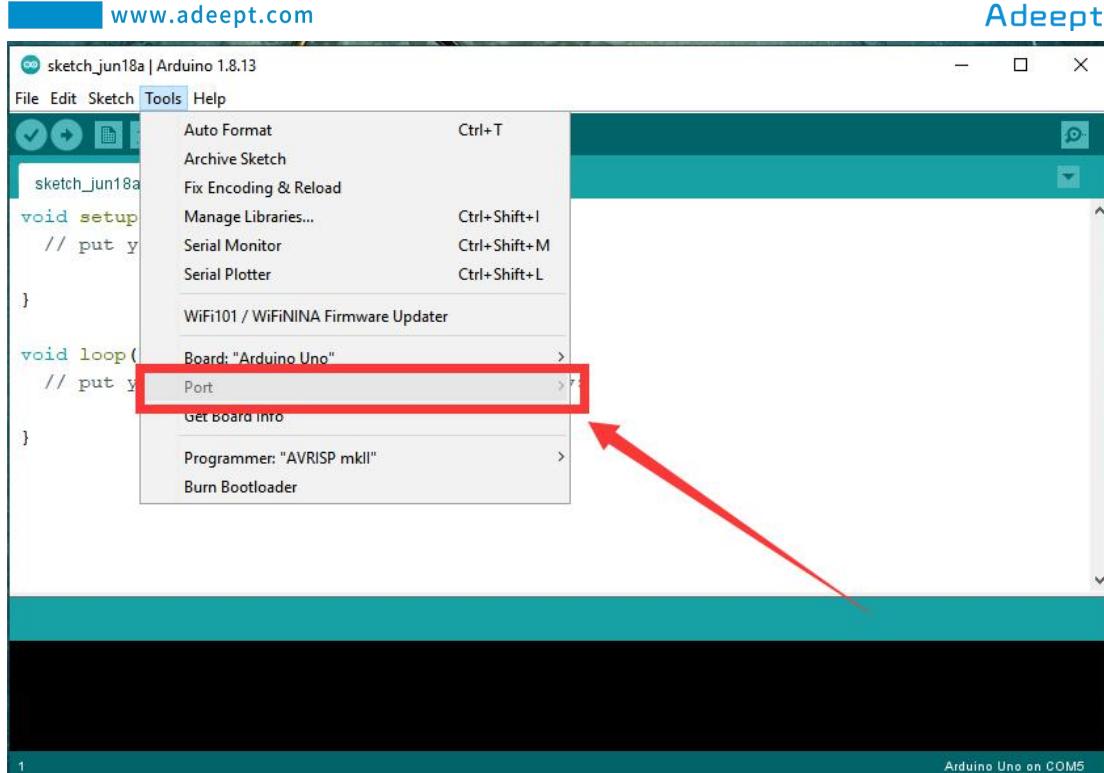
(2) Select the Arduino Uno development board in Tools

Open Arduino IDE under Tools—>Board. Select Arduino UNO in the list.



(3) Install CH341SER driver

1. Open Arduino IDE, you will see the serial port is not accessible, meaning that you have not installed the serial port driver.



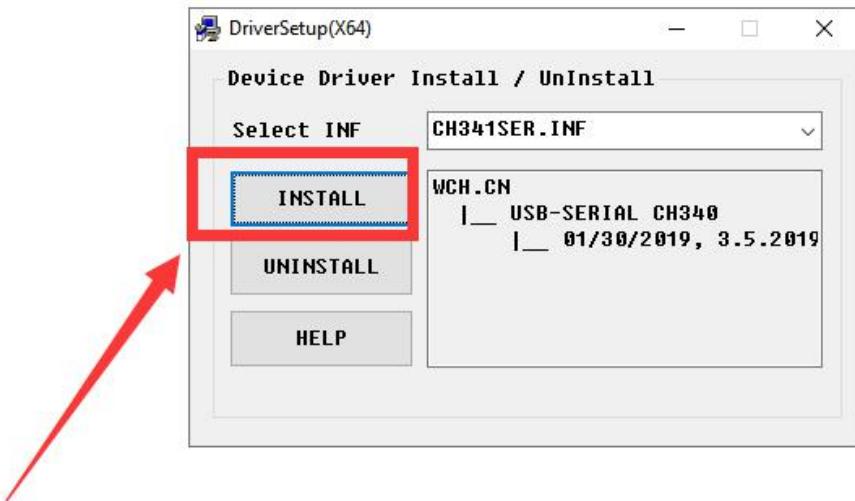
2. You need to find the user folder provided by Adeept: Adeept_Ultimate_Kit_For_Arduino_V2_0, find the Adeept driver folder and open it. If you are using a Windows system, you can double-click directly to open CH341SER_Windows.EXE and install corresponding driver according to the computer operating system.

Name	Date modified	Type	Size
Adeept driver introduction.txt	6/17/2020 2:18 PM	Text Document	1 KB
CH341SER_ANDROID.ZIP	6/17/2020 2:18 PM	WinRAR ZIP 压缩...	2,360 KB
CH341SER_LINUX.ZIP	6/17/2020 2:18 PM	WinRAR ZIP 压缩...	9 KB
CH341SER_MAC.ZIP	6/17/2020 2:18 PM	WinRAR ZIP 压缩...	149 KB
CH341SER_Windows.EXE	6/17/2020 2:18 PM	Application	277 KB

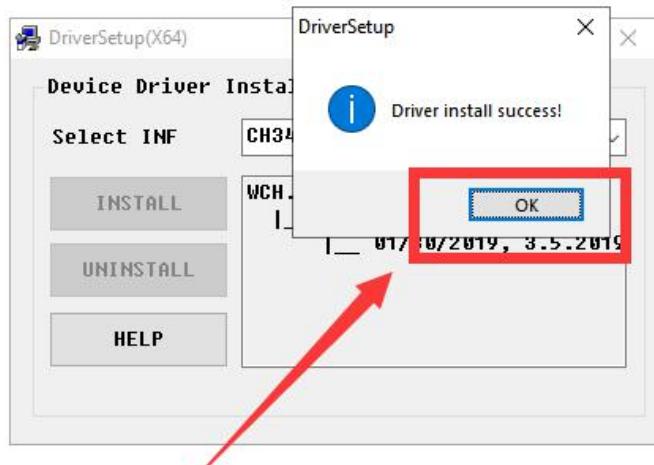
A red box highlights the "CH341SER_Windows.EXE" file in the list, and a red arrow points from below the table towards this highlighted file.

3. Click INSTALL. Wait for the installation to succeed. And click OK.

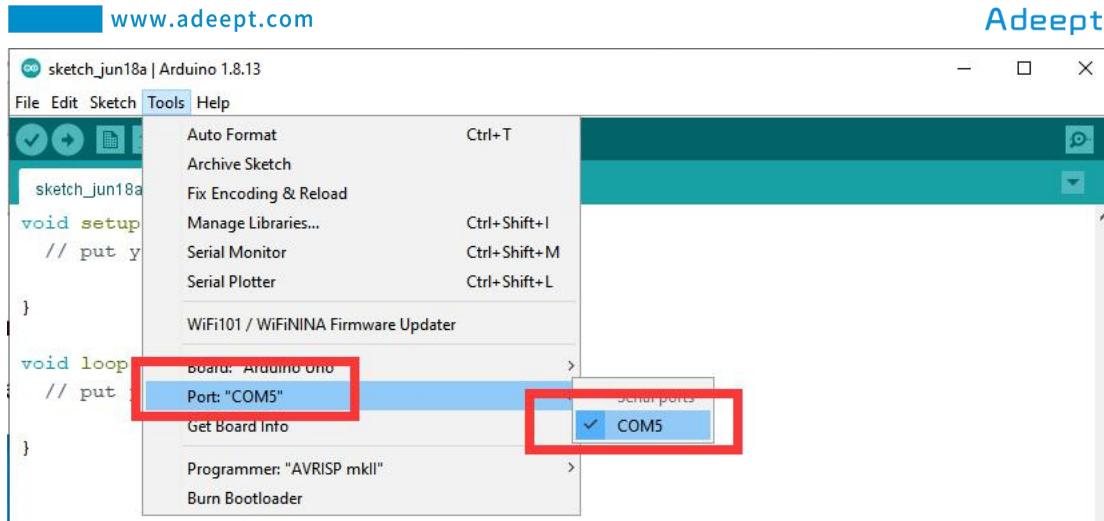
Name	Date modified	Type	Size
Adeept driver introduction.txt	6/17/2020 2:18 PM	Text Document	1 KB
CH341SER_ANDROID.ZIP	6/17/2020 2:18 PM	WinRAR ZIP 压缩...	2,360 KB
CH341SER_LINUX.ZIP	6/17/2020 2:18 PM	WinRAR ZIP 压缩...	9 KB
CH341SER_MAC.ZIP	6/17/2020 2:18 PM	WinRAR ZIP 压缩...	149 KB
CH341SER_Windows.EXE	6/17/2020 2:18 PM	Application	277 KB



Adeept driver introduction.txt	6/17/2020 2:18 PM	Text Document	1 KB
CH341SER_ANDROID.ZIP	6/17/2020 2:18 PM	WinRAR ZIP 压缩...	2,360 KB
CH341SER_LINUX.ZIP	6/17/2020 2:18 PM	WinRAR ZIP 压缩...	9 KB
CH341SER_MAC.ZIP	6/17/2020 2:18 PM	WinRAR ZIP 压缩...	149 KB
CH341SER_Windows.EXE	6/17/2020 2:18 PM	Application	277 KB



4. Now you will find the Arduino serial port is accessible (different computer configuration has different serial port). It means that the Arduino UNO development board has been successfully connected to the computer. You will need to pay attention to this connection step in the following course.



6.The solution for situation that Arduino IDE cannot be opened

When opening the Arduino IDE, you will suddenly encounter a situation that it cannot be opened.



【Solution】

You need to find the Arduino15 folder in the `\Users\ASUS\AppData\Local\Arduino15` directory of the C drive. As shown below:

Name	Date modified	Type	Size
cache	5/21/2020 6:35 PM	File folder	
logs	5/21/2020 6:34 PM	File folder	
library_index.json	6/10/2020 10:45 AM	JSON File	12
library_index.json.sig	6/10/2020 10:45 AM	SIG File	
package_index.json	6/10/2020 10:45 AM	JSON File	
package_index.json.sig	6/10/2020 10:45 AM	SIG File	
preferences.txt	6/10/2020 10:43 AM	Text Document	

You need to delete the package_index.json file, and then reopen the Arduino IDE.

cache	5/21/2020 6:35 PM	File folder
logs	5/21/2020 6:34 PM	File folder
library_index.json	6/10/2020 10:58 AM	JSON File
library_index.json.sig	6/10/2020 10:58 AM	SIG File
package_index.json	6/10/2020 10:58 AM	JSON File
package_index.json.sig	6/10/2020 10:58 AM	SIG File
package_index.txt	6/10/2020 10:45 AM	Text Document
preferences.txt	6/10/2020 10:58 AM	Text Document

Arduino graphical programming

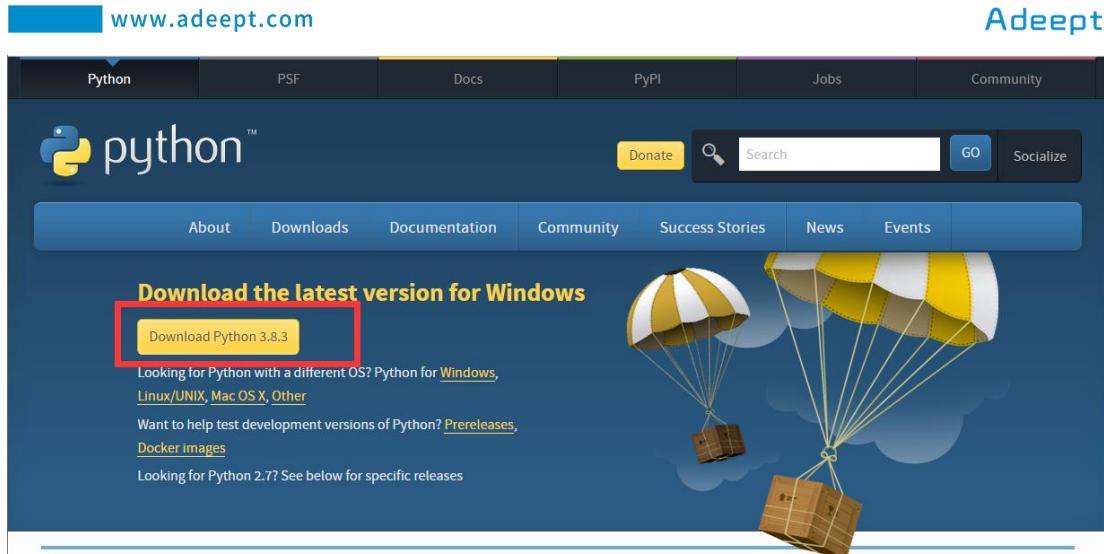
We creatively provide users with Arduino graphical programming tools-GwBlock. Using graphical program instruction blocks to achieve control of Arduino with the Web page. Compared with the traditional pure character interface code programming platform, graphical programming is more conducive to learners who have not mastered C/C++. If you have studied Scratch, then you will be able to easily master the graphical programming of Arduino. Next we will teach you how to build graphical programming of Arduino.

1. Downloading and installing Python

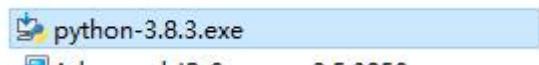
(1) Log in to the official website by browser: <https://www.python.org/downloads/>



(2) Click the "Download Python 3.8.3" button to download and wait for the download to complete:



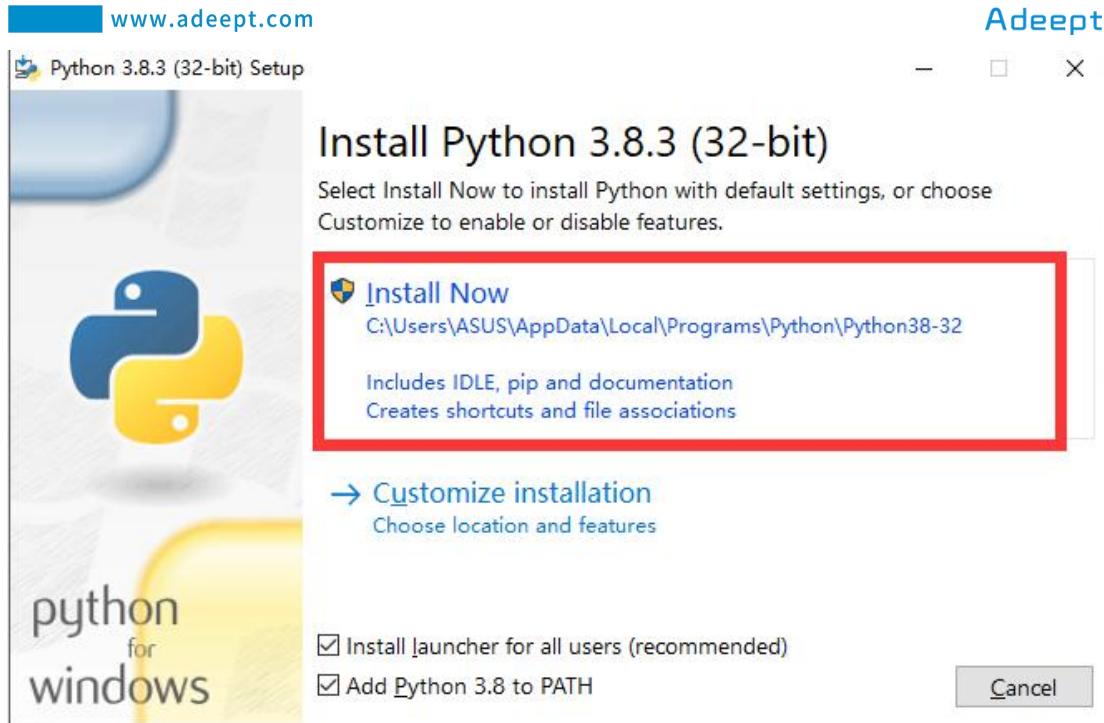
(3) Open the downloaded file, double-click to open it to install:



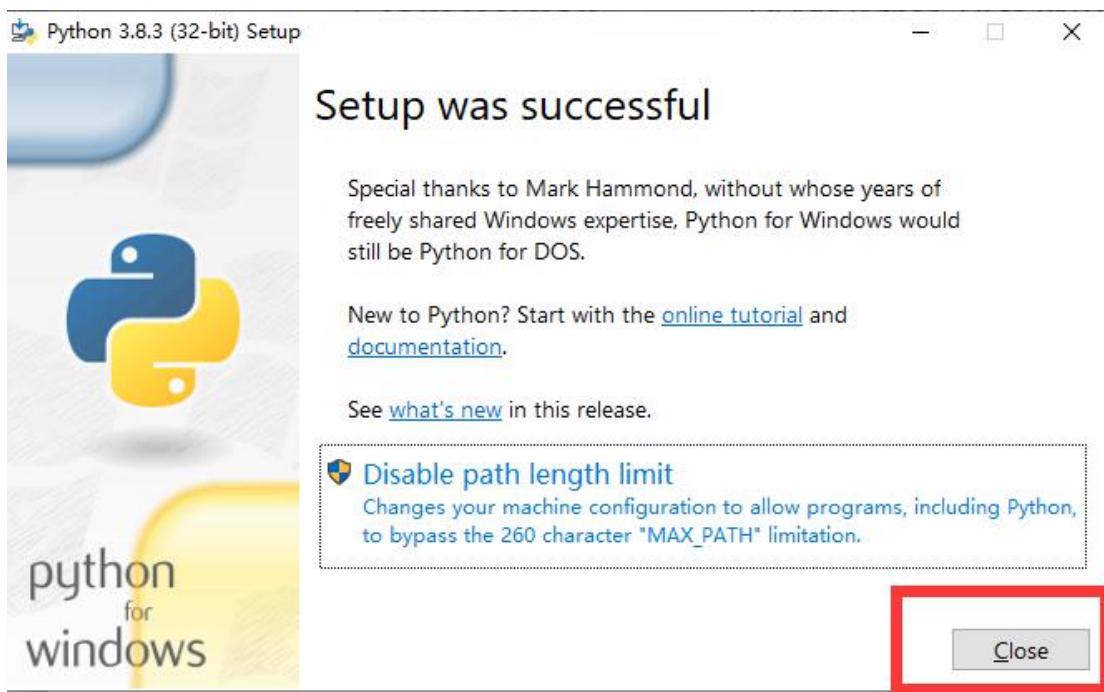
(4) Select the "Add Python 3.8 to PATH" option:



(5) Then click "Install Now" to install.



(6) Wait for the Python installation to complete and click "Close" to close.



2. Installing pySerial and connecting GwBlock graphical editor

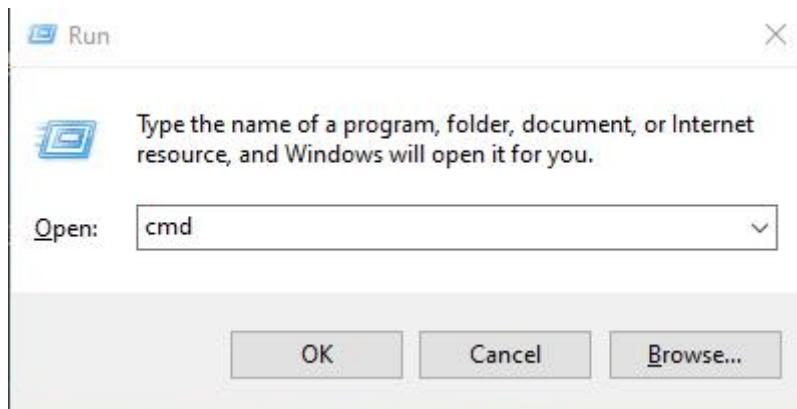
pySerial encapsulates the serial communication module, supporting Linux, Windows, BSD (may support all operating systems that support POSIX), Jython (Java)

and IconPython (.NET and Mono). The pyserial module encapsulates access to the serial port. The port number starts from 0 by default. There is no need to know the port name in the program. APIs like file reading and writing, read and write (readline, etc. are also supported), support binary transmission, no null elimination, no cr-lf conversion. All programs are completed by Python In addition to the standard library, it does not depend on other packages, except pywin32 (windows), JavaComm (Jython). POSIX (Linux, BSD) only depends on the Python standard library. APIs like file read and write, read, write (readline, etc. are also supported), support binary transmission, no null elimination, no cr-lf conversion, all programs are all done by Python, and do not depend on other packages except the standard library, except pywin32 (windows), JavaComm (Jython). POSIX (Linux, BSD) only depends on the Python standard library.

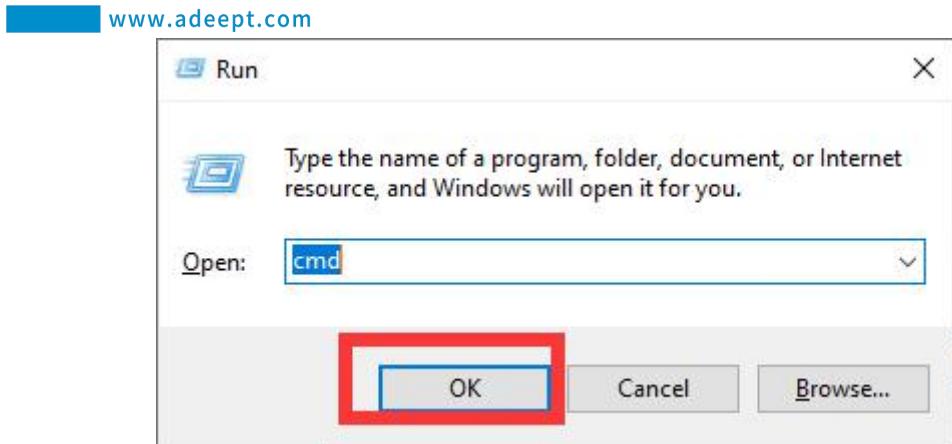
Before downloading and installing, you need to connect the Arduino development board to the computer.



(1) Press Win+R shortcut key to open CMD under Windows 10:



(2) Click "OK":



(3) Enter the command in the window:

pip install pyserial

Press the Enter and wait for the installation to complete.



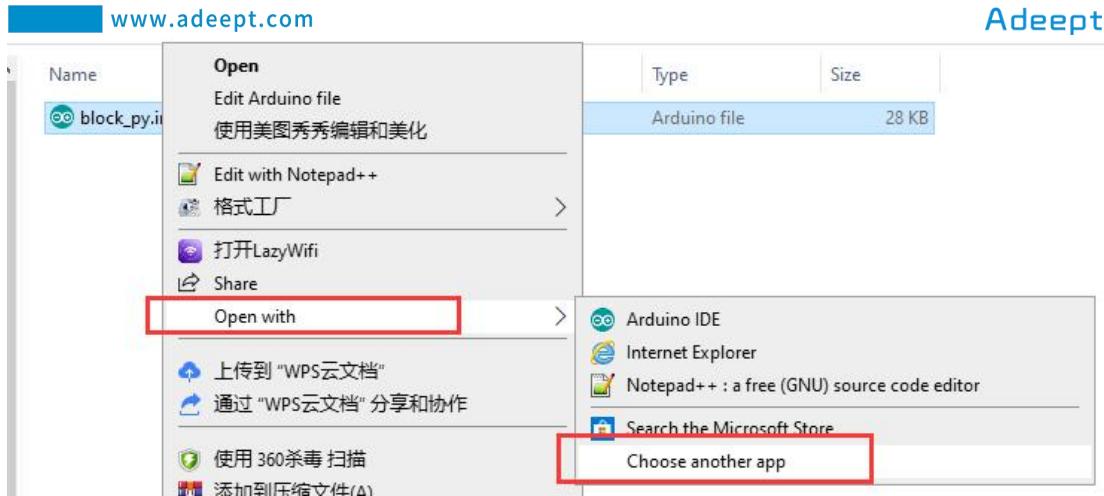
```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 10.0.18362.836]
(c) 2019 Microsoft Corporation。保留所有权利。
C:\Users\ASUS>pip install pyserial
Collecting pyserial
  Downloading https://files.pythonhosted.org/packages/0d/e4/2a744dd9e3be04a0c0907414e2a01a7c88bb3915cbe3c8cc06e209f59c30/pyserial-3.4-py2.py3-none-any.whl (193kB)
[██████████| 194kB 3.2kB/s]
Installing collected packages: pyserial
Successfully installed pyserial-3.4
WARNING: You are using pip version 19.2.3, however version 20.1.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
C:\Users\ASUS>
```

(4) Open the folder "Adeept_Ultimate_Kit_For_Arduino_V2_0" provided by Adeept to the user → "block_py" and find this file: "block_py.ino".

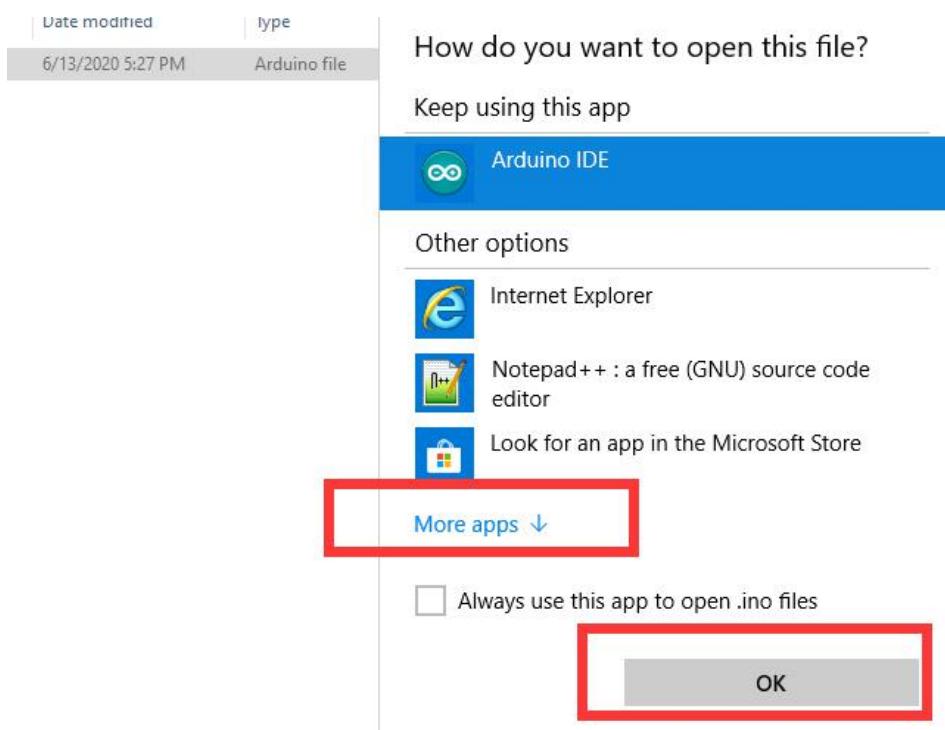
名称	修改日期	类型
Adeept_Ultimate_Kit_For_Arduino_C_C...	2020/6/4 15:52	文件夹
Arduino libraries	2020/6/4 15:52	文件夹
block_py	2020/6/4 15:52	文件夹
websocket	2020/6/5 11:31	文件夹

名称	修改日期	类型
block_py.ino	2020/5/25 17:55	INO 文件

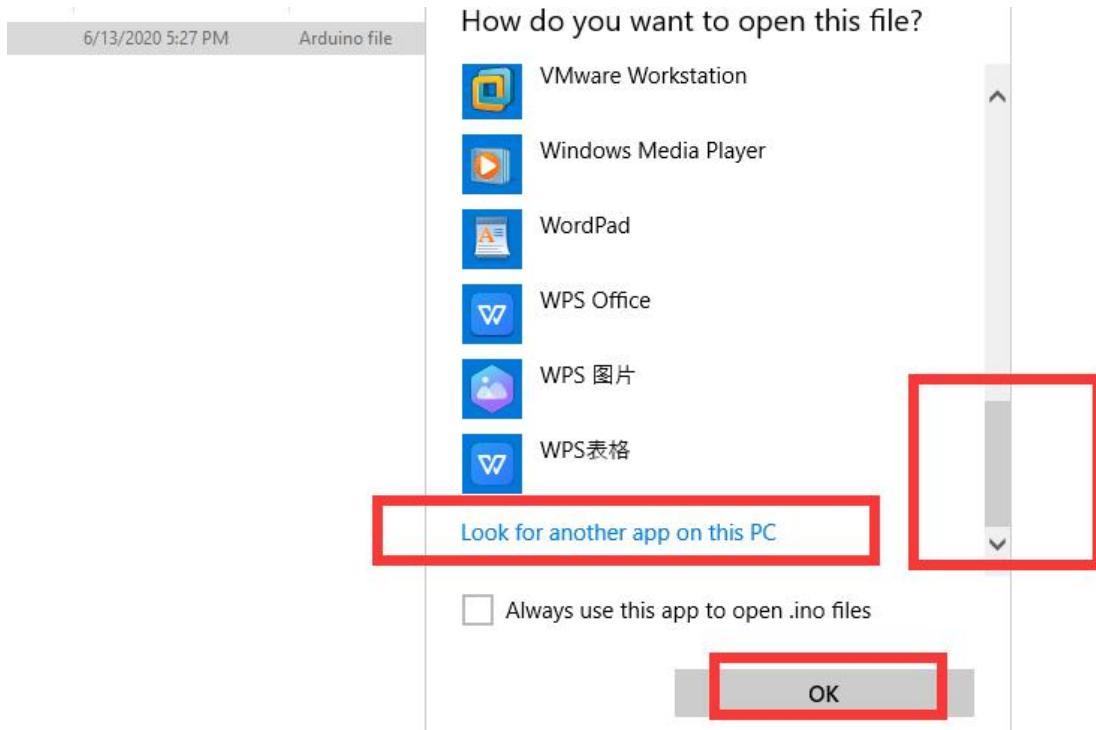
(5) Then right-click the file: "block_py.ino". Select "Open with" → "Choose another app".



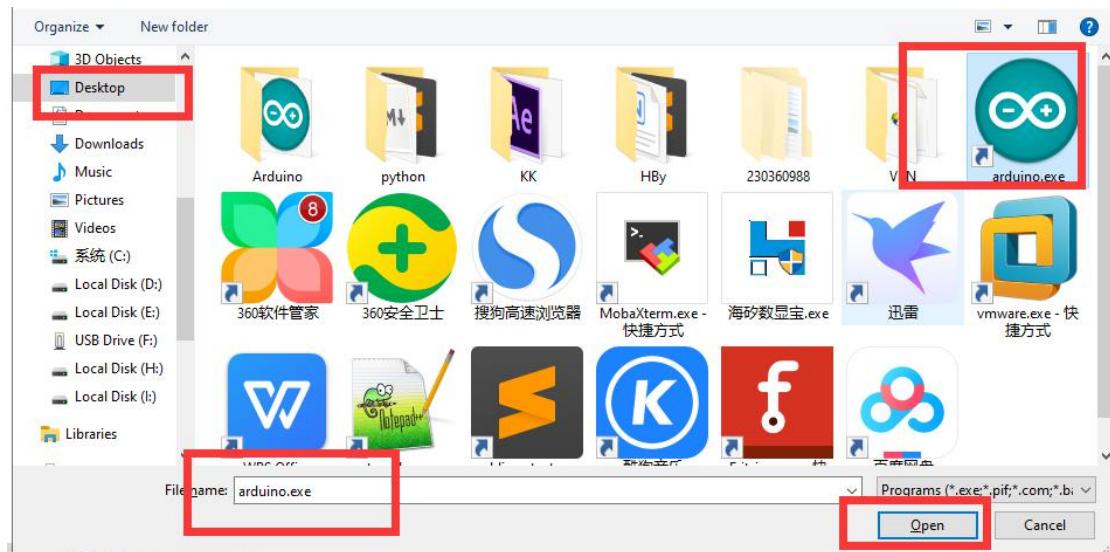
(6) Click "More apps", then click "OK".



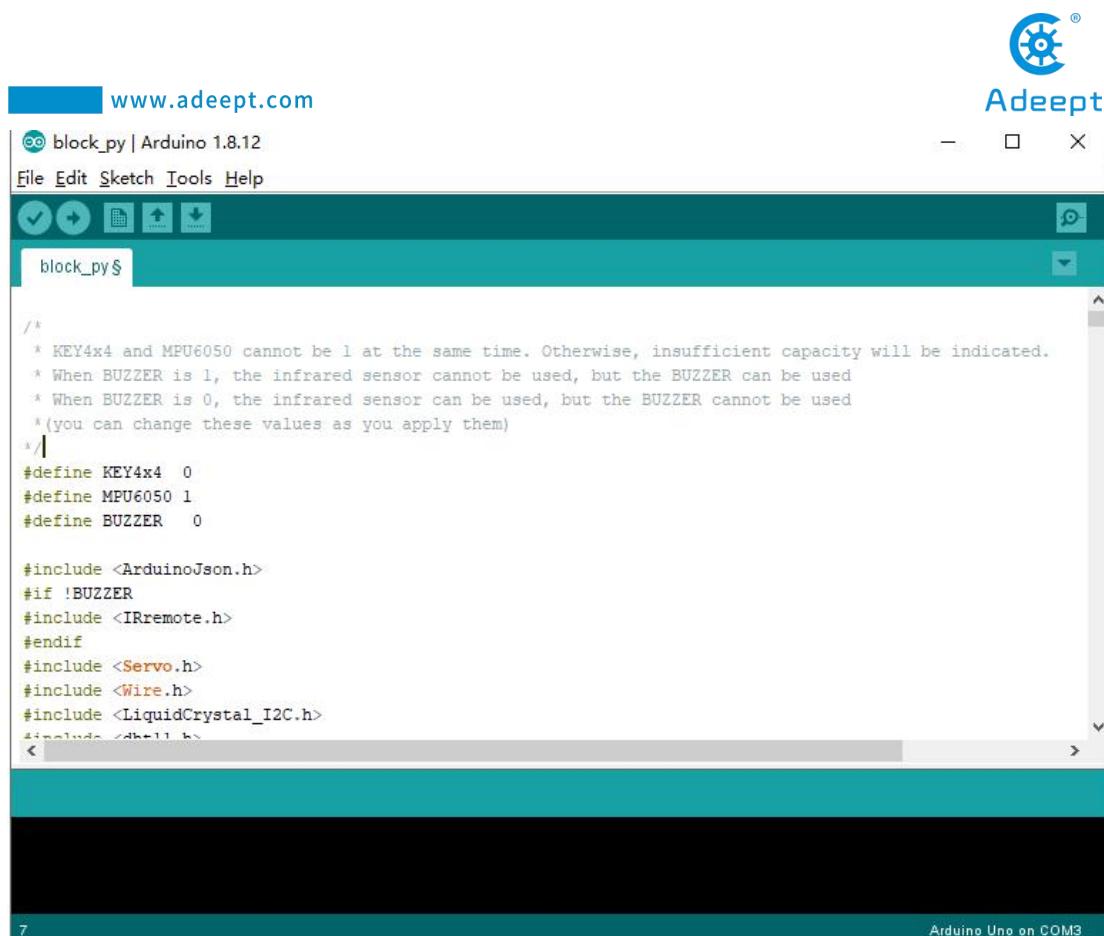
(7) Slide the mouse down, click "Look for another app on this PC", and then click "OK".



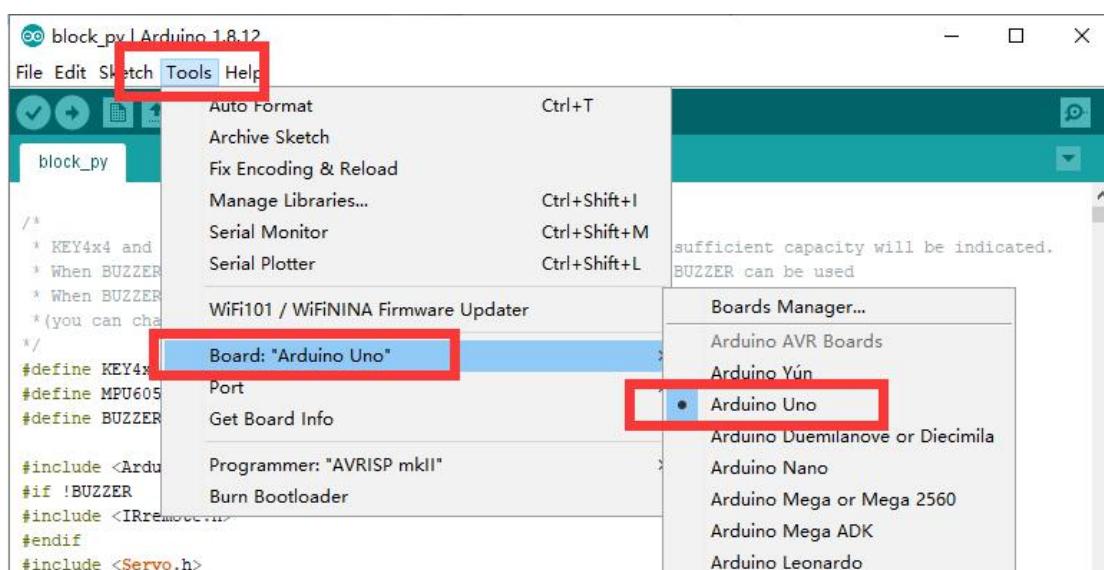
- (8) Find the Arduino software on Desktop or the place where you installed the Arduino software, select it and click "Open".



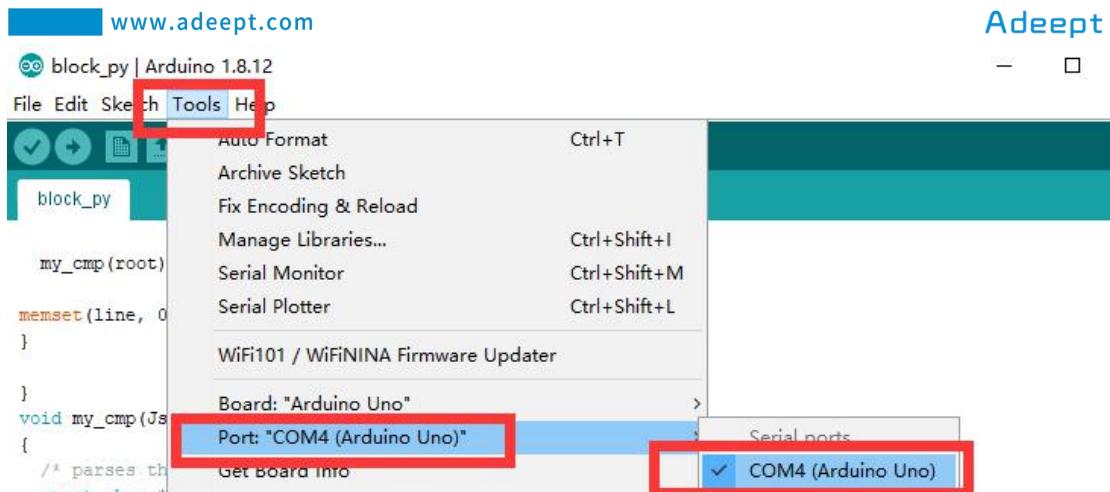
- (9) Then the Arduino software opens the file "block_py.ino".



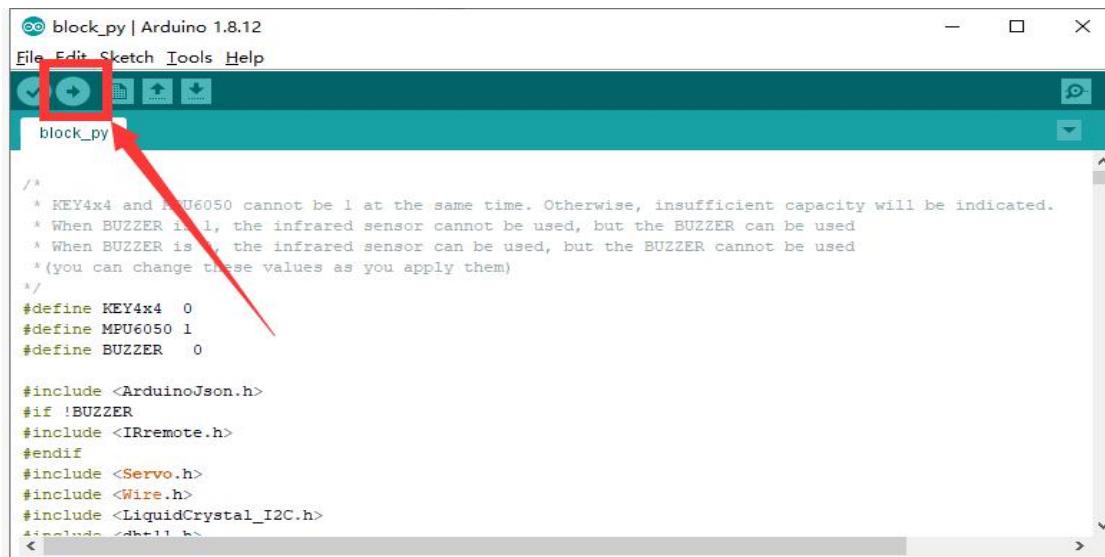
(10) First select the Arduino development board as UNO version with Tools.



(11) Then continue to select the "Port" of the Arduino connected to the computer with Tools



(12) Click the Upload button  to download the code program to the Arduino development board.



(13) After downloading, an error warning will be displayed because the Arduino software we downloaded and installed from the official website is missing some related libraries.

www.adeept.com



```

block_py

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */

#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
<

ArduinoJson.h: No such file or directory

```

Copy error messages

(14) Solution: We first close the Arduino software. Open the folder we provide to users: Adeept_Ultimate_Kit_For_Arduino_V2_0. Then open the "Arduino libraries" folder inside. As shown below:

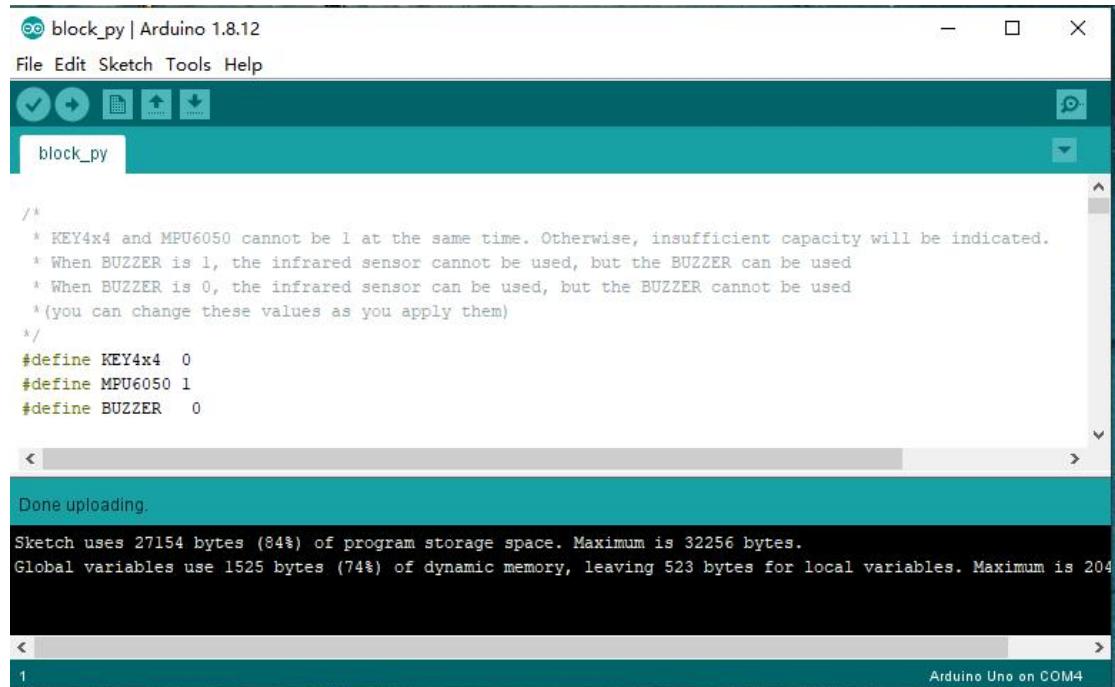
名称	修改日期	类型	大小
ArduinoJson	2020/6/4 15:52	文件夹	
Dht11	2020/6/4 15:52	文件夹	
IRremote	2020/6/4 15:52	文件夹	
Keypad	2020/6/4 15:52	文件夹	
LiquidCrystal_I2C	2020/6/4 15:52	文件夹	

(15) We need to copy all the folders inside to the libraries folder under the folder where we installed the Arduino software. As shown below after copying:

名称	修改日期	类型	大小
Adafruit_Circuit_Playground	2020/2/13 10:32	文件夹	
ArduinoJson	2020/6/4 15:41	文件夹	
Bridge	2020/2/13 10:32	文件夹	
Dht11	2020/6/4 15:41	文件夹	
Esplora	2020/2/13 10:32	文件夹	
Ethernet	2020/2/13 10:32	文件夹	
Firmata	2020/2/13 10:32	文件夹	
GSM	2020/2/13 10:32	文件夹	
IRremote	2020/6/4 15:41	文件夹	
Keyboard	2020/2/13 10:32	文件夹	
Keypad	2020/6/4 15:41	文件夹	
LiquidCrystal	2020/2/13 10:32	文件夹	
LiquidCrystal_I2C	2020/6/4 15:41	文件夹	
Mouse	2020/2/13 10:32	文件夹	

(16) We use the Arduino software to open the "block_py.ino" file again, and then click

the Upload button again to download the code program to the Arduino development board. After the download is successful, the following figure is shown below:



The screenshot shows the Arduino IDE interface with the sketch named 'block_py'. The code includes defines for KEY4x4, MPU6050, and BUZZER. The status bar at the bottom indicates 'Done uploading.' and shows memory usage details: Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes. Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 204. The serial monitor window shows 'Arduino Uno on COM4'.

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

```

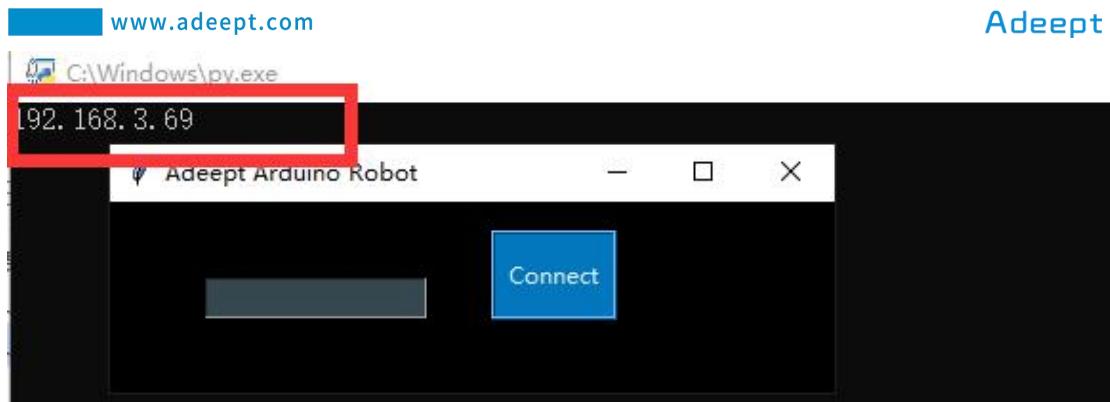
(17) Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provided to the user again, find the websocket folder and open it.

名称	修改日期	类型	大小
Adeept_Ultimate_Kit_For_Arduino_C_C...	2020/6/4 15:52	文件夹	
Arduino libraries	2020/6/4 15:52	文件夹	
block_py	2020/6/4 15:52	文件夹	
websocket	2020/6/5 11:31	文件夹	

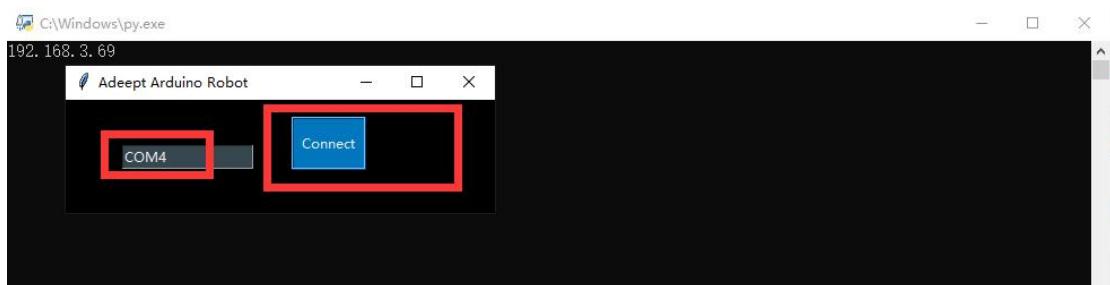
(18) Double-click to open this file: GUI info v1.0.py

名称	修改日期	类型	大小
Adeept.py	2020/5/25 17:55	Python File	4 KB
GUI info v1.0.py	2020/5/25 17:55	Python File	8 KB

(19) After opening the file, the following interface will appear. We need to record this IP address, which will be used later: 192.168.3.69



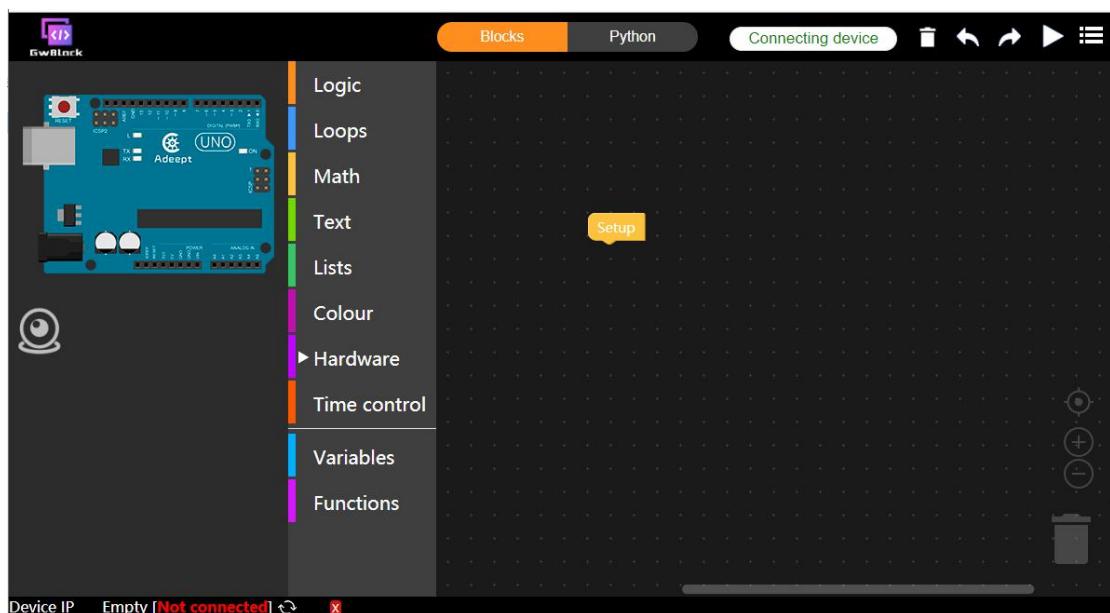
(20) In the input box, enter the port we set in step (11). Everyone's port is different. The port of my Arduino development board is: COM4. After entering, click the Connect button.



(21) Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

After successfully entering the website, the interface is as follows:

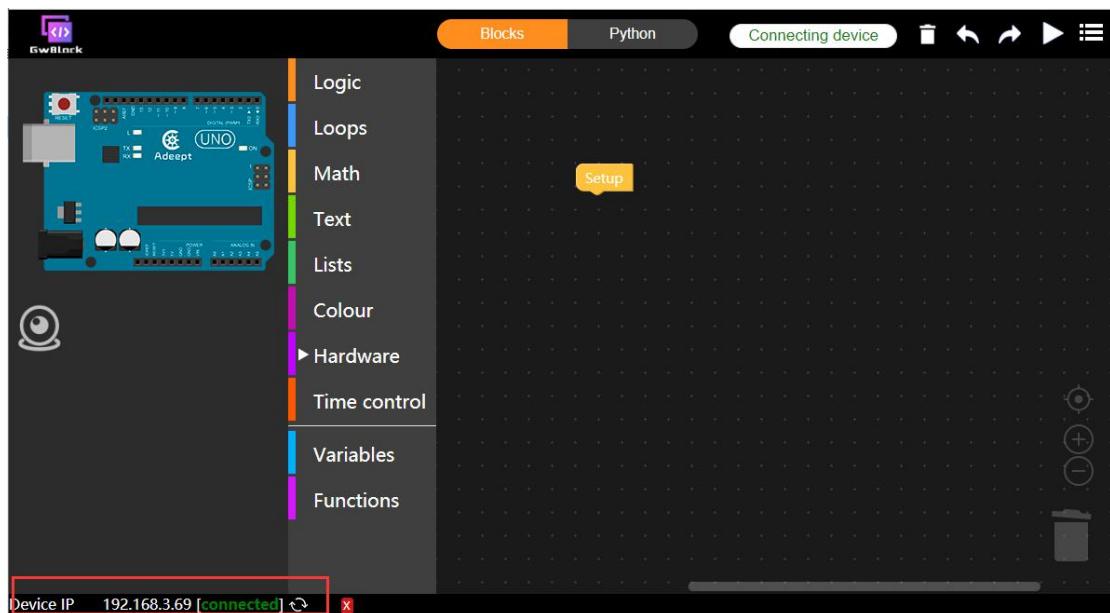


(22) Click the "Connecting device" button in the upper right corner. In the pop-up box, enter the IP address we recorded in step (19), as shown below, and then click the

Connecting button.



(23) After the connection is successful, a green "connected" will appear in the lower left corner of the interface, indicating that the connection with the Arduino is successful.



3.The method of reconnecting the GwBlock graphical editor

In the following cases, you need to reconnect to the GwBlock graphical editor:

- [1] When you close the GwBlock editor.

[2] When you close the Adeept Arduino Robot window.

[3] When you restart the computer.

[4] When you log in to the GwBlock website again.

[5] When you close the Arduino IDE.

(1) Before doing the experiment, you must first connect the Arduino development board to the computer.

(2) First, open the folder we provide to the user:

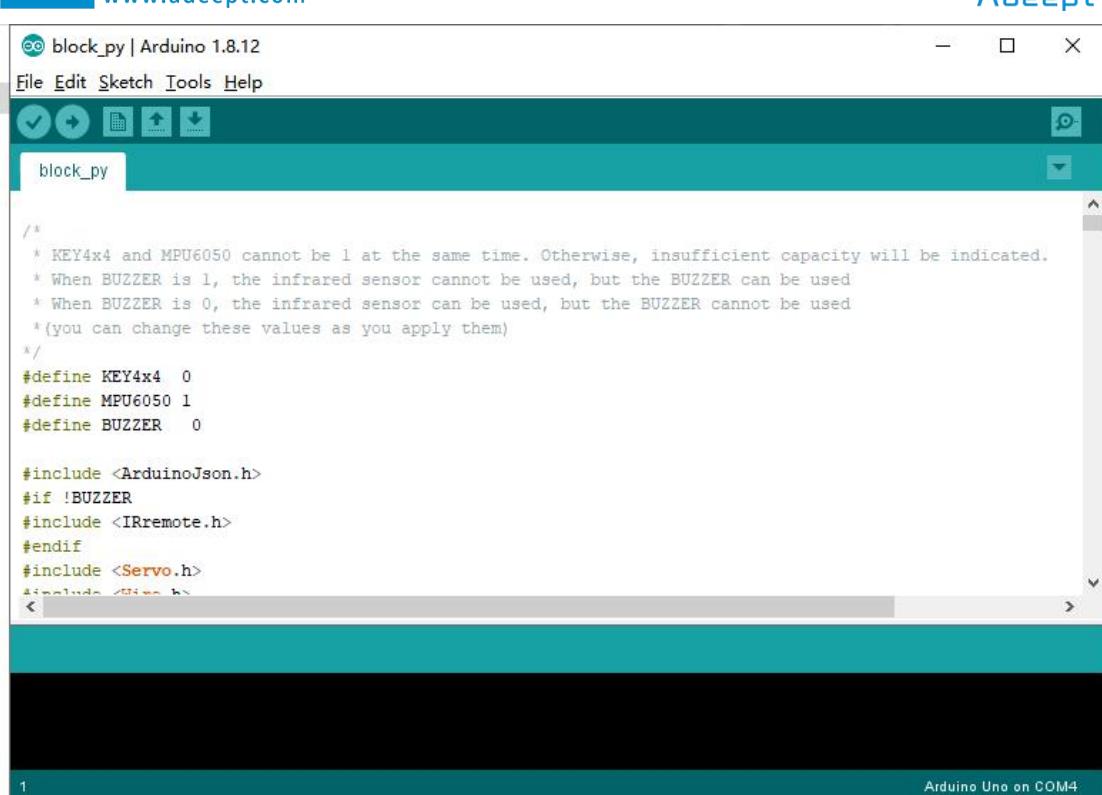
Adeept_Ultimate_Kit_For_Arduino_V2_0, after opening, as shown below:

名称	修改日期	类型	大小
Adeept_Ultimate_Kit_For_Arduino_C_C...	2020/6/4 15:52	文件夹	
Arduino libraries	2020/6/4 15:52	文件夹	
block_py	2020/6/4 15:52	文件夹	
websocket	2020/6/5 11:31	文件夹	

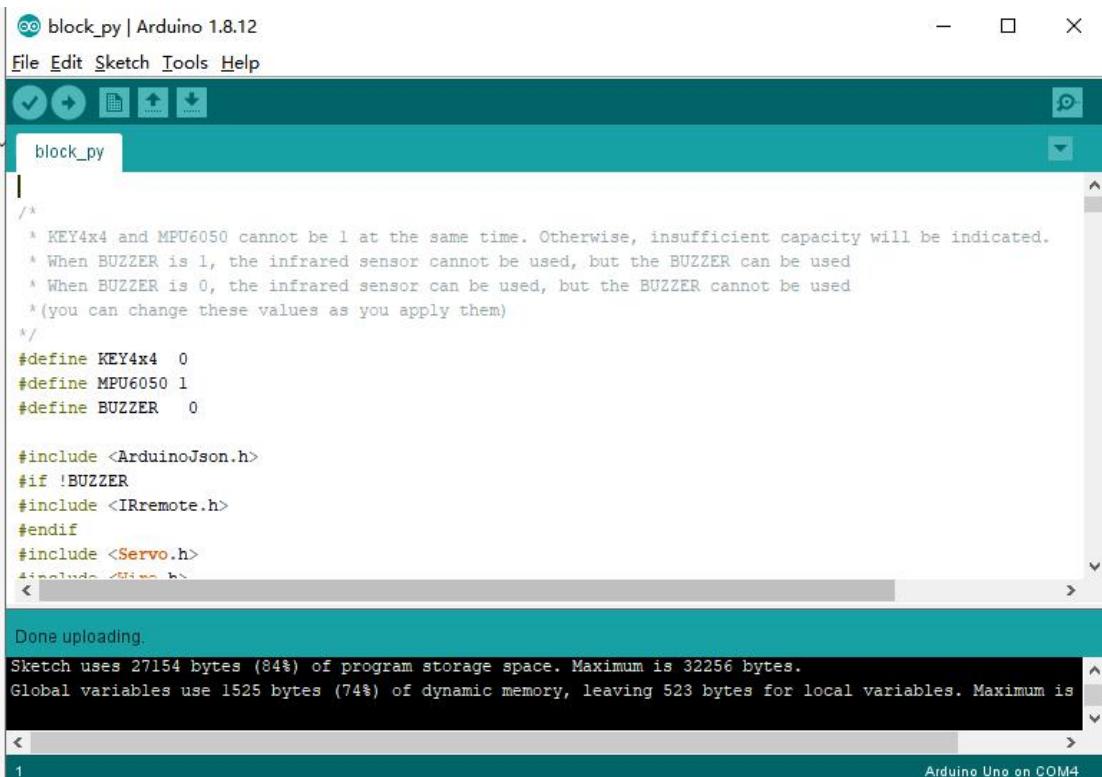
(3) Open this folder: block_py again, and find a block_py.ino file inside, as shown below:

名称	修改日期	类型	大小
block_py.ino	2020/5/25 17:55	INO 文件	2

(4) Double-click to open this block_py.ino file (use Arduino to open!), as shown below:



(5) First select COM4 as the Arduino port in Tools, click the icon  in the upper left corner to download the program to the Arduino development board, and the following picture will be prompted after successful download:



(6) Next, open the folder we provide to the user:

`Adeept_Ultimate_Kit_For_Arduino_V2_0.`

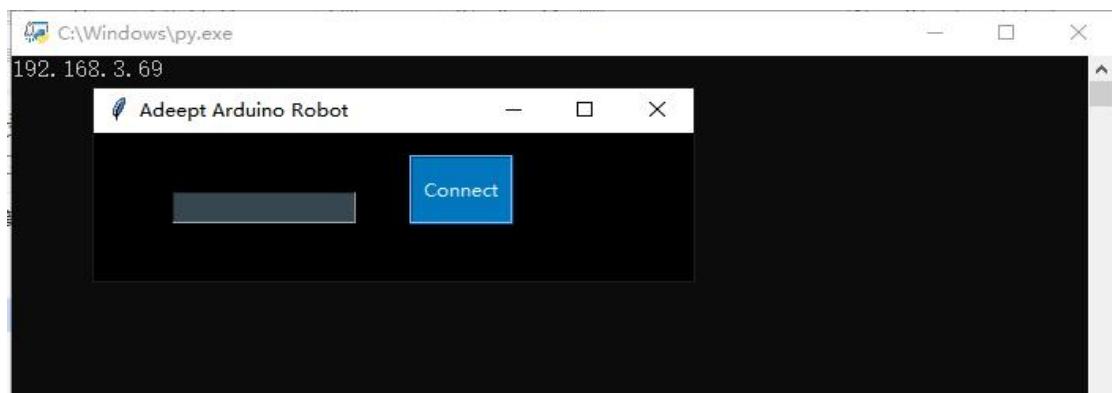
Then find the websocket folder. As shown below:

 Adeept_Ultimate_Kit_For_Arduino_C_C...	2020/6/4 15:52	文件夹
 Arduino libraries	2020/6/4 15:52	文件夹
 block_py	2020/6/4 15:52	文件夹
 websocket	2020/6/5 11:31	文件夹

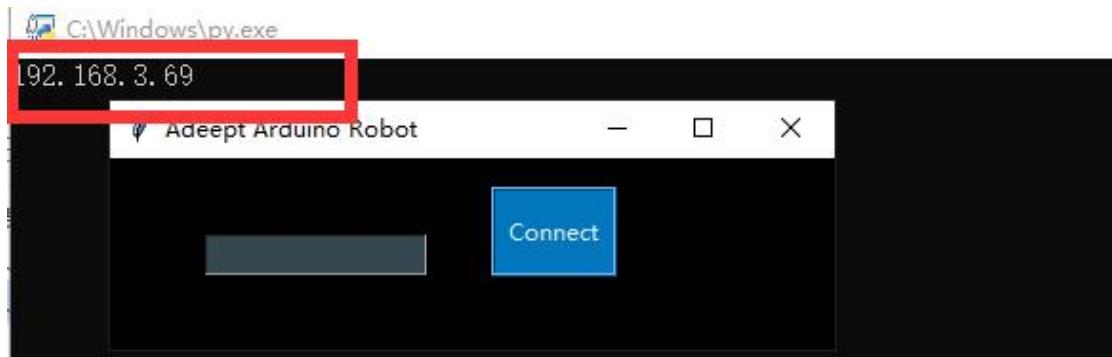
(7) Open the websocket folder. There is a file inside: `GUI info v1.0.py`.

名称	修改日期	类型	大小
 Adeept.py	2020/5/25 17:55	Python File	
 GUI info v1.0.py	2020/5/25 17:55	Python File	

(8) Double-click to open the `GUI info v1.0.py` file, and the following picture will appear after opening:

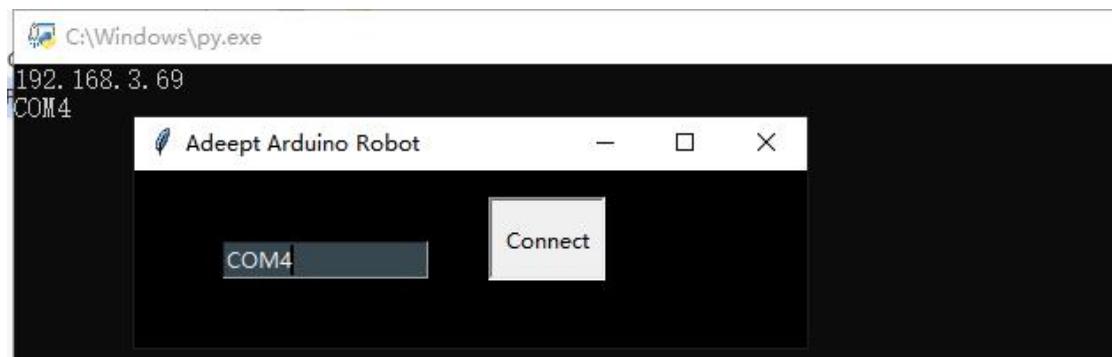


(9) After opening the file, the following interface will appear. We need to record this IP address, which will be used later: 192.168.3.69



(10) Enter the Arduino software download program in the input box of Adeept Arduino Robot. The connected port number: COM4. Click the Connect button. As

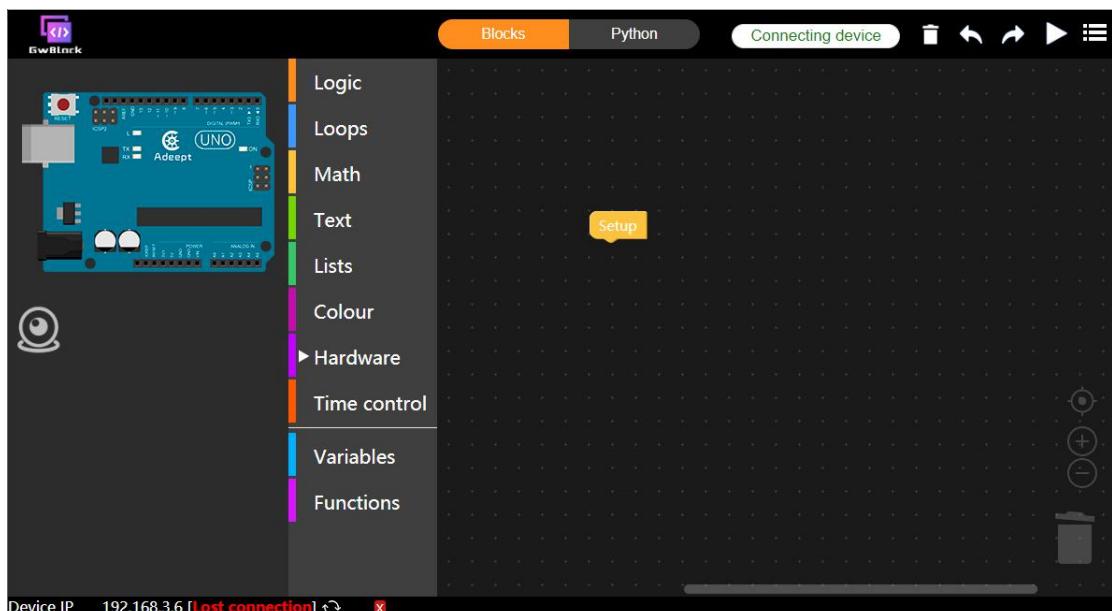
shown below:



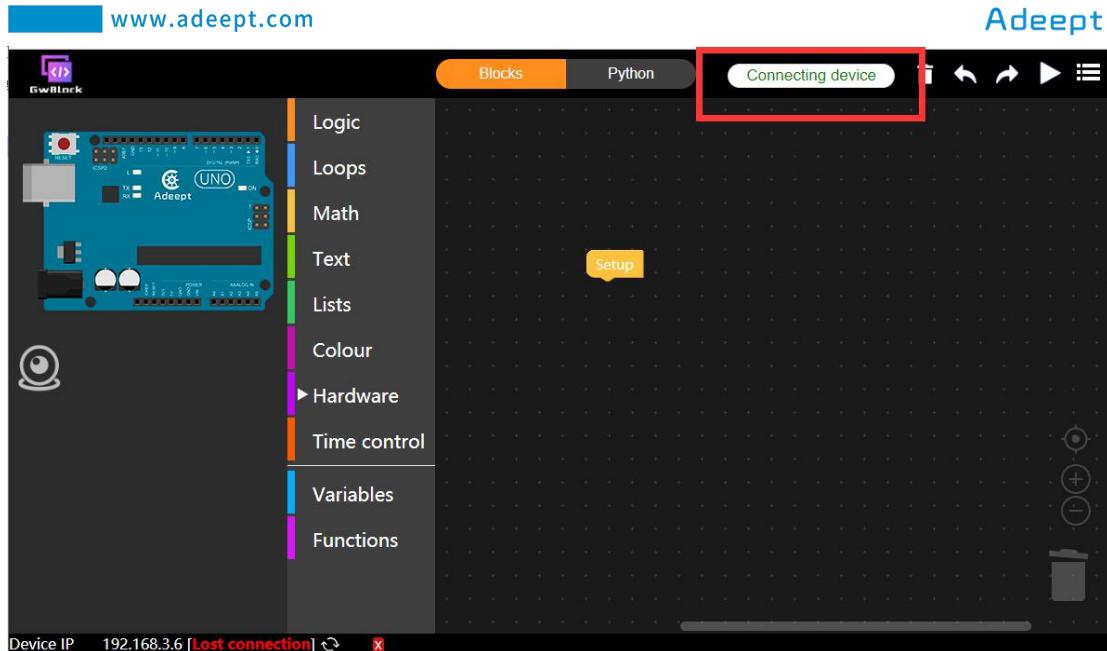
(11) Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

After successfully entering the website, the interface is as follows:



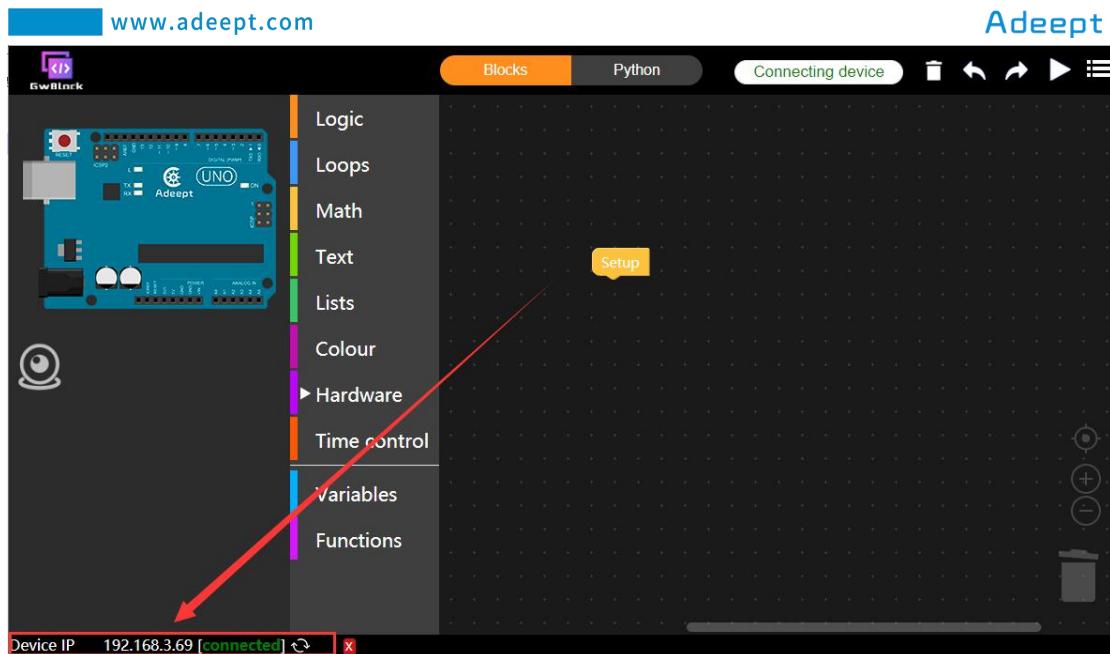
(12) Click the "Connecting device" button in the upper right corner. It will show as below:



- (13) In the pop-up box, enter the IP address in step (9): 192.168.3.69. And then click the Connecting button, as shown below:



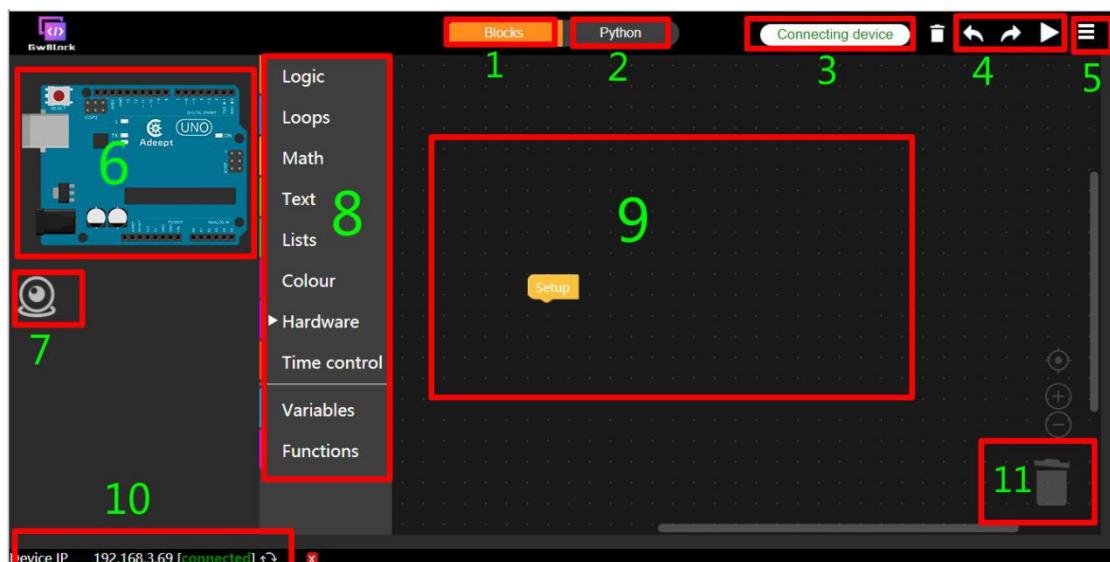
- (14) After a successful connection, as shown below, a green connected prompt will appear in the lower left corner. It means that we have successfully connected to the GwBlock graphical editor, with which we can realize the graphical programming of Arduino. Later I will teach you how to use GwBlock graphical editor.



4. Get to know about Arduino's graphical editor

GwBlock

The functions of the buttons on the main interface of the GwBlock editor will be described in detail below according to the function numbers in the picture. As shown below:



【1】 Blocks: Click this button to switch to the programming mode of the graphical code block.

【2】 Python: Click this button to display the edited graphical code block in the form

of Python code.

【 3 】 Connecting device: Click this button to connect to the Arduino development board, which requires you to enter the IP address.

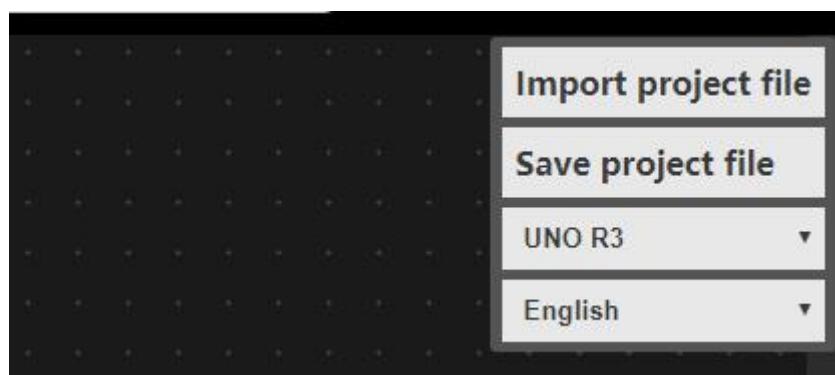


【 4 】 (1)  is the cancel button. Click it to return to the state of the previous operation (cancel this operation).

(2)  is the forward button. Click it to advance to the state of the next operation.

(3)  is the button to run the program. Click it to run the correct program we have compiled.

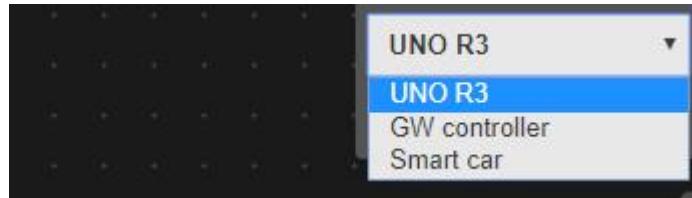
【 5 】  is a drop-down menu button:



Under the drop-down menu button, you can "Import project file" and "Save project file". In addition, by the drop-down button on the right of UNO R3, you can switch to the programming mode of different controllers. We are using Arduino UNO R3 version of the development board in the current course, so we choose UNO R3 mode to programmatically control Arduino.



You can also switch the language display mode of the editor by the drop-down button to the right of English. Currently, we only support English and Simplified Chinese.

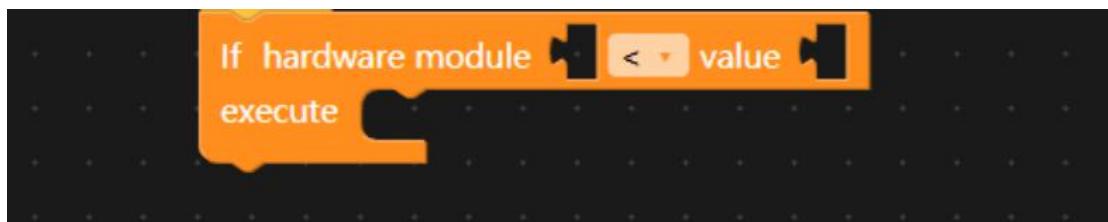


【6】 is the icon of the Arduino UNO development board, indicating that it is currently in the Arduino programming mode.

【7】 is the camera button, which is gray in the initial off state:  When you click it, the camera will turn red: , indicating that the camera is on. At this time, a screen window will appear in the editing area on the right. Click  to close the camera screen. The screen can be dragged to any position by clicking the camera. As shown below:



【8】 is the code instruction module toolbar. You can select the code instruction block you need here.



【9】 is the editing area (code area or work area), where we edit the code instruction block. Each code instruction block must be placed below **Setup**.



【10】 is the connection status of the device. There are two states:

(1) The following is displayed when the device is not connected:



(2) After the device is successfully connected, the display is as follows:

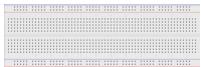


【11】 is a code trash, you can drag and drop the code instruction block to delete it.

Lesson 1 How to Light the Blinking LED

In this lesson, we will learn how to light the blinking LED.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	1	
LED	1	

2. The LED Light-Emitting Diode

(1)What is Diode?

1. Definition:

A diode is an electronic device made of semiconductor materials (silicon, selenium, germanium, etc.). It has unidirectional conductivity. In the circuit, current (voltage) is allowed to flow in a single direction, and it will prevent current and voltage from passing in the opposite direction. Therefore, the diode has positive and negative poles (anode and cathode). When a positive voltage is applied to the anode and cathode of the diode, the diode conducts. When a reverse voltage is applied to the anode and cathode, the diode is turned off. Therefore, the turning on and off of the diode is equivalent to the turning on and off of the switch.

2. Structure of a diode:

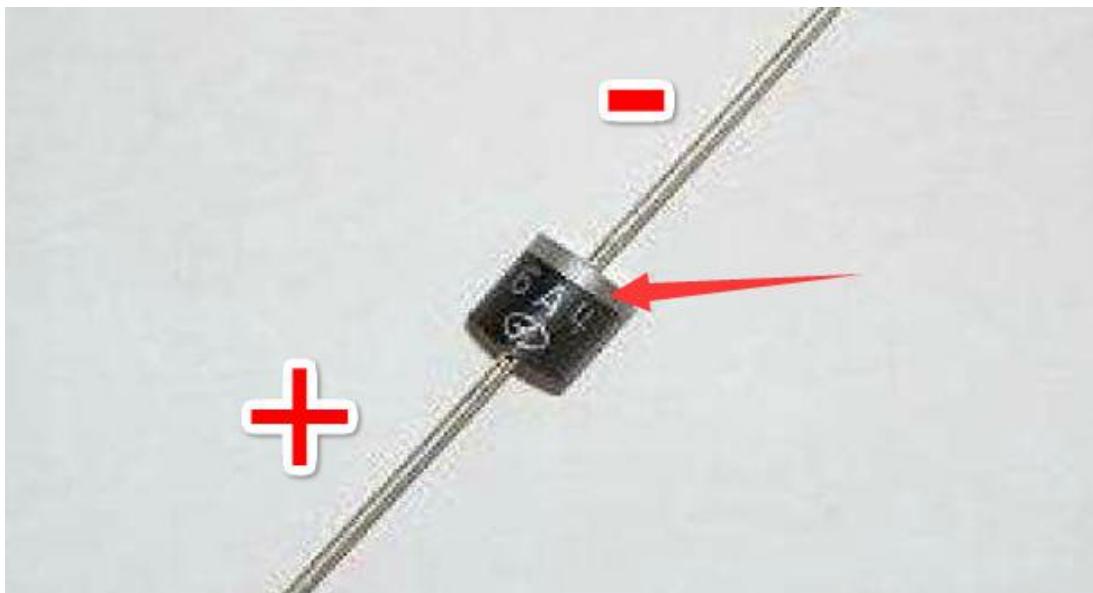
A diode is made up of a PN junction plus corresponding electrode leads and package.

PN junction: P-type semiconductor and N-type semiconductor are made on the same semiconductor (usually silicon or germanium) substrate, and a space charge region formed at their interface is called PN junction. The electrode drawn from the P area is called the anode, and the electrode drawn from the N area is called the cathode. Because of the unidirectional conductivity of the PN junction, the direction of the current when the diode is turned on is from the anode to the cathode through the inside of the tube. The following picture is a common diode:



(2) Anode (Positive Electrode) and Cathode (Negative Electrode) of Common Diodes:

1. The positive and negative electrodes of ordinary diodes are shown in the picture:



2. The positive and negative poles of the light-emitting diode are shown in the picture: the long pin is the positive pole, and the short pin is the negative pole.

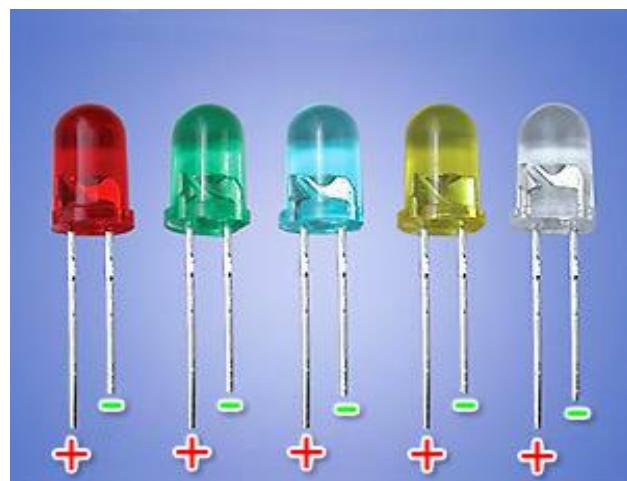
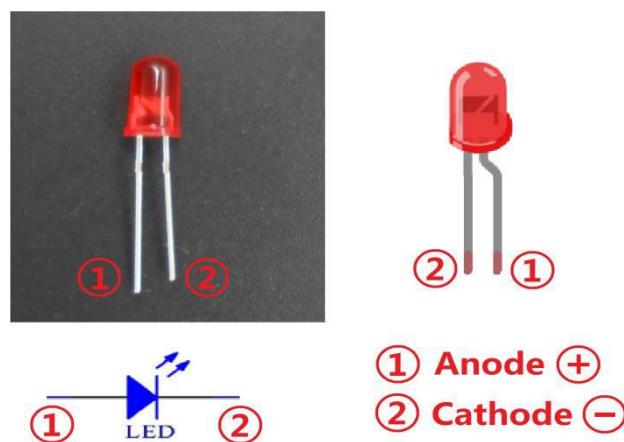


figure1-3



(3)What is Light-Emitting Diode?

1. Definition

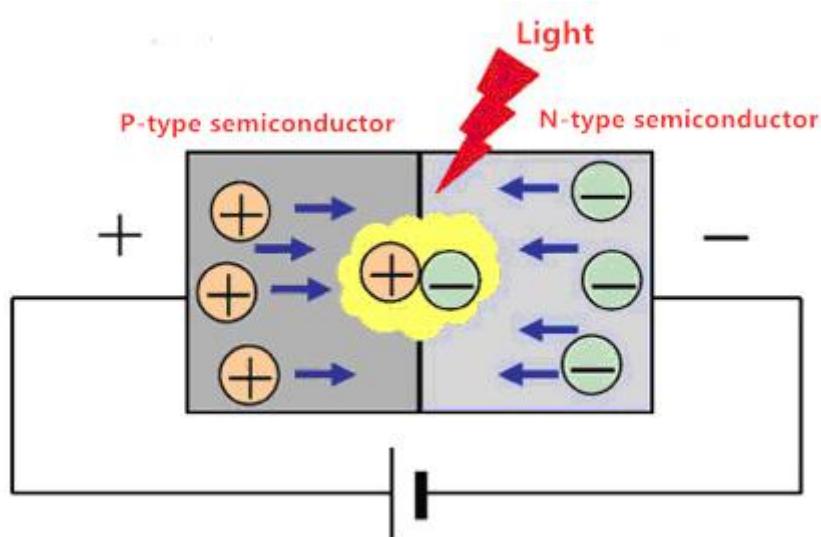
Light-emitting diode is short as LED. The light-emitting diode is a type of diode, composed of a PN junction. With the characteristics of a diode, it has unidirectional conductivity like a diode. In the circuit, current can only flow in from the anode of the diode and out of the cathode. The difference between a light emitting diode and a diode is that it can emit light, red light, green light, blue light, yellow light, etc.

2. Why do LEDs emit light?

(1) The reason why the light-emitting diode emits light is that its core light-emitting part is a chip. The chip in the light-emitting diode is a compound gallium nitride, which has a property: it emits light when low current passes, mainly to convert electrical energy into light energy.

(2) The principle of light emitting diode:

The principle of light emission is mainly a combination of N (-: negative) semiconductors with many electrons (negatively charged) and P (+: positive) semiconductors with many holes (positively charged). When the semiconductor is applied with a forward voltage, electrons and holes will move and combine again at the junction. It is during the junction that a lot of energy is generated, and this energy is released in the form of light. As shown below:



Light-emitting diodes have a wide range of uses in modern society, such as lighting, displays, medical devices, circuits and instruments as indicator lights.

(3) Classification of light-emitting diodes

Light-emitting diodes can also be divided into ordinary monochrome light-emitting diodes, high-brightness light-emitting diodes, ultra-high-brightness light-emitting diodes, color-changing light-emitting diodes, flashing light-emitting diodes, voltage-controlled light-emitting diodes, infrared light-emitting diodes, and negative resistance light-emitting diodes.

(4) How to wire (connect) the LED in the circuit

1. The LED has positive and negative poles. When connecting to the circuit, we need to connect the positive pole of the LED to the positive pole of the power supply, and the negative pole to the negative pole of the power supply. The light emitting diode cannot be directly connected to the power supply, which can damage the components. In the circuit using LED light-emitting diodes, a resistor with a certain resistance value must be connected in series.

2. Calculation formula of LED current limiting resistor:

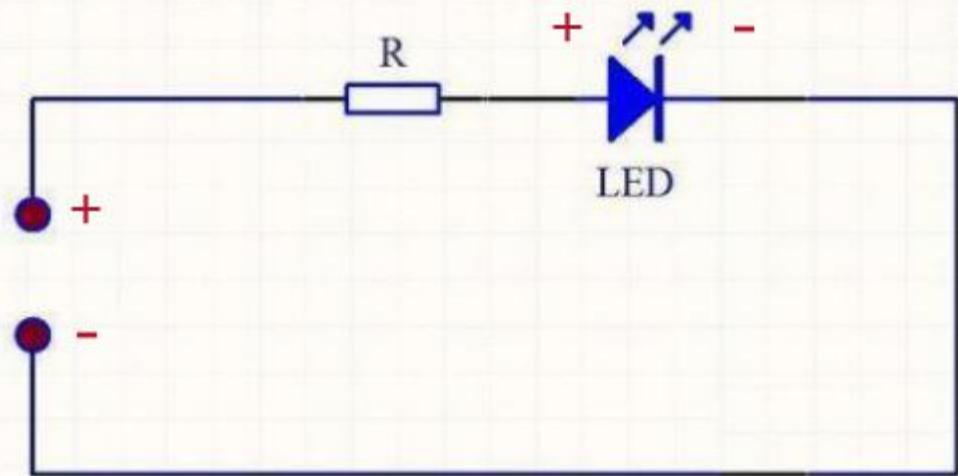
$$\text{Limit Resistance (R)} = \frac{\text{Limit Voltage (U)}}{\text{Limit Current (I)}}$$

The limit current I of the LED light-emitting diode in our course is 5-20mA, and the limit voltage U is 3.3V, so our limit resistance R is:

$$R = \frac{U}{I} = \frac{3.3V}{(5 \sim 20mA)} = 165\Omega \sim 660\Omega$$

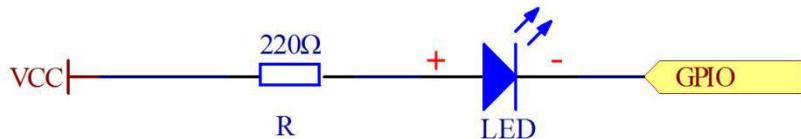
In the experiment, when we choose the connecting resistance, we can only choose between $165\Omega \sim 660\Omega$.

3. The circuit diagram of the LED light-emitting diode is as follows:

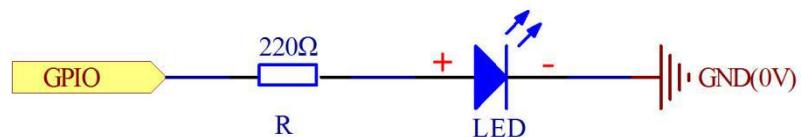


4.Two ways to connect LED and GPIO

The first method is as shown in the picture below: The positive pole of the LED is connected to the positive pole of VCC (+ 3.3V), and the negative pole of the LED is connected to the Raspberry Pi GPIO. When GPIO outputs a low level, the LED lights up because of a potential difference between VCC and GPIO; when GPIO outputs a high level, because the potential difference between VCC and GPIO does not form, the LED turns off.



The second method is as follows: the positive pole of the LED is connected to GPIO, and the negative pole of the LED is connected to GND (0V). When the GPIO outputs a high level, the LED lights up because of the potential difference between GPIO and GND; when the GPIO outputs a low level, because the potential difference between the GPIO and GND does not form, the LED lights are off.



(5) Main parameters and precautions of Light-emitting Diodes

(1) Allowable power consumption (P_m): Maximum value of the product of the forward DC voltage applied to both ends of the LED and the current flowing through it. If this value is exceeded, the LED will become hot or damaged.

(2) Maximum forward DC current (I_{Fm}): Maximum forward DC current allowed to be added. Exceeding this value can damage the diode.

(3) Maximum reverse voltage (V_{Rm}): Maximum reverse voltage allowed to be applied. Above this value, the light emitting diode may be damaged by breakdown.

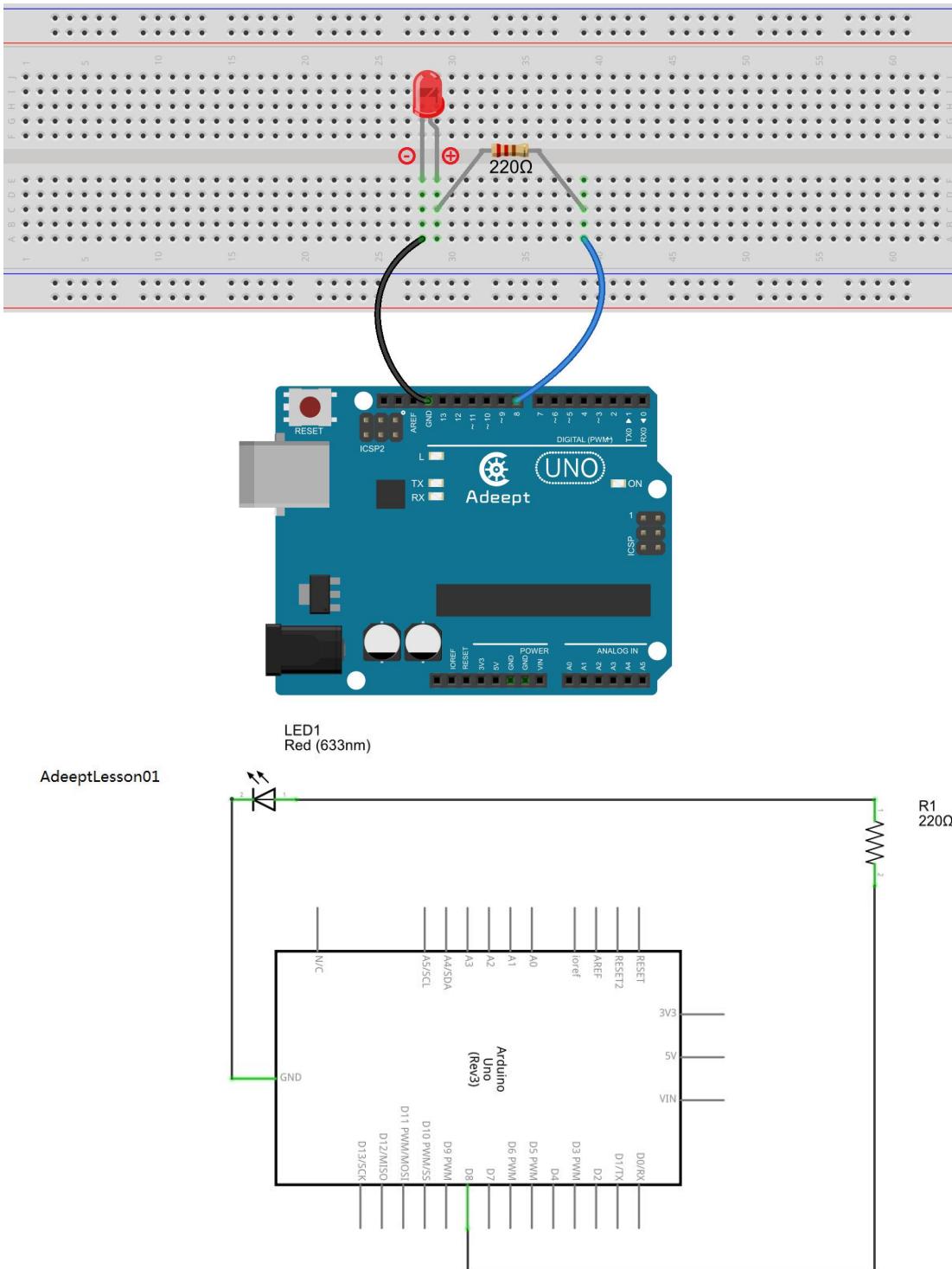
(4) Working environment (t_{opm}): Ambient temperature range where the LED can work normally. Below or above this temperature range, the LED will not work properly and the efficiency will be greatly reduced.

【Remarks】

1. LED cannot be directly connected to the power supply, which can damage the components. In the circuit using LED light-emitting diodes, a resistor with a certain resistance value must be connected in series.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:



4. How to light the blinking LED

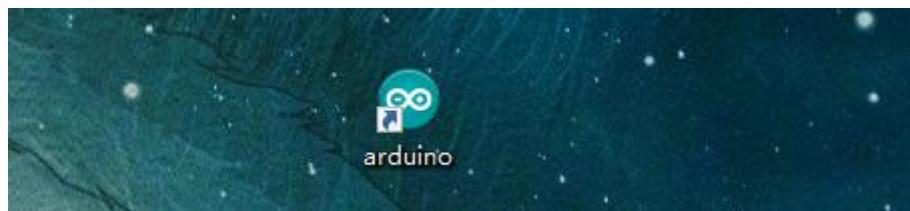
We provide two different methods to light the blinking LED. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code

block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

1. Using C language to program to light the blinking LED on Arduino UNO

(1) Compile and run the code program of this course

1. Open the Arduino IDE software, as shown below:

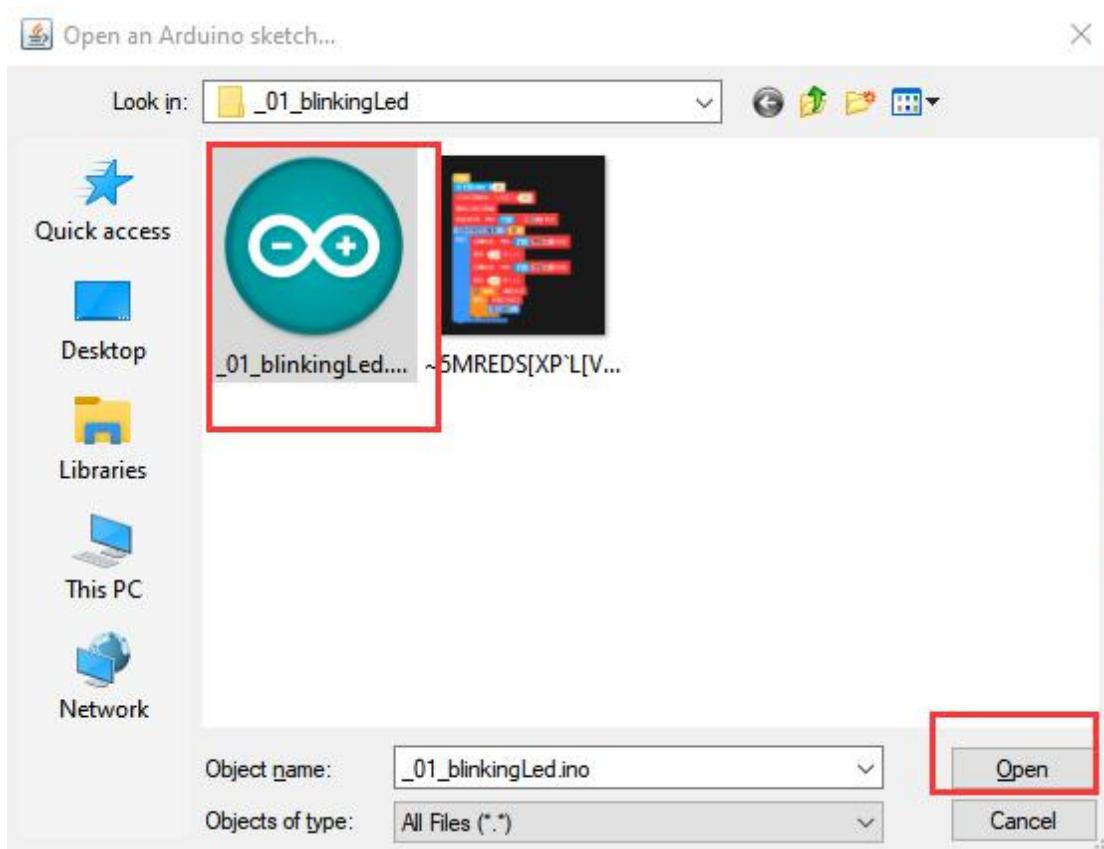


2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\01_blinkingLed directory.

Select_01_blinkingLed.ino. This file is the code program we need in this course. Then click Open.



4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

www.adeept.com

_01_blinkingLed | Arduino 1.8.12

File Edit Sketch Tools Help



_01_blinkingLed

```

/*
File name: 01_blinkingLed.ino
Description: Lit LED, let LED blinks.
Website: www.adeept.com
E-mail: support@adeept.com
Author: Tom
Date: 2015/05/02
*/
int ledPin=8; //definition digital 8 pins as pin to control the LED
void setup()
{
    pinMode(ledPin,OUTPUT); //Set the digital 8 port mode, OUTPUT: Output mode
}
void loop()
{
    digitalWrite(ledPin,HIGH); //HIGH is set to about 5V PIN8
    delay(1000); //Set the delay time, 1000 = 1S
    digitalWrite(ledPin,LOW); //LOW is set to about 5V PIN8
    delay(1000); //Set the delay time, 1000 = 1S
}

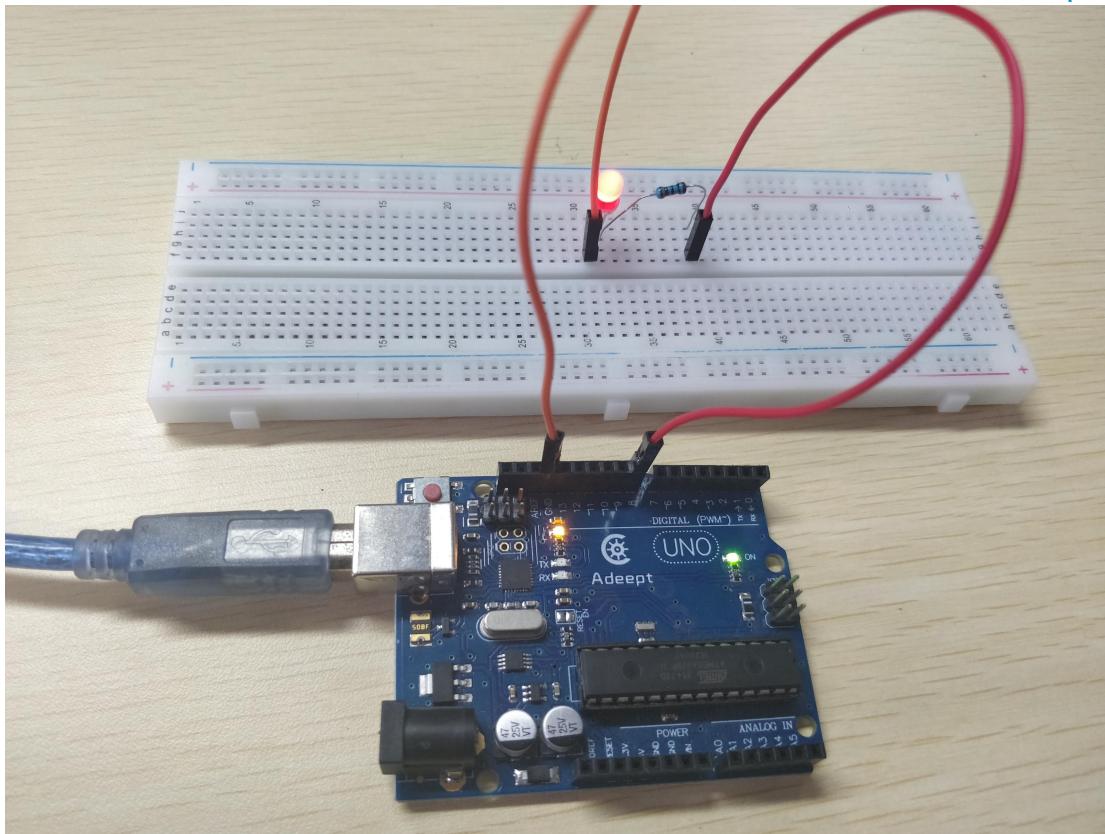
```

Done uploading.

Sketch uses 936 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2

Arduino Uno on COM4

5. At this time, when we observe the LED, it will be on then off, and keep cycling, indicating that our experimental test is successful. The physical connection diagram of the experiment is as follows:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to light the LED on Arduino UNO. We will introduce how our core code can be achieved:

Define the pin of led as pin 8 by int ledPin=8; in the setup() method, set ledPin as the output mode with pinMode(ledPin, OUTPUT); in the loop() method, set ledPin to high level with digitalWrite(ledPin, HIGH), and the LED will be on. Control the LED to stay on for 1s with delay (1000); set ledPin to low level with digitalWrite (ledPin, LOW), then the LED is off. Control the LED to be off continuously for 1s with delay (1000). When the code in loop() is executed cyclically, the LED will be on and off repeatedly.

```

int ledPin=8; //definition digital 8 pins as pin to control the LED
void setup()
{
    pinMode(ledPin,OUTPUT);    //Set the digital 8 port mode, OUTPUT: Output mode
}
void loop()
{
    digitalWrite(ledPin,HIGH); //HIGH is set to about 5V PIN8
    delay(1000);             //Set the delay time, 1000 = 1S
    digitalWrite(ledPin,LOW); //LOW is set to about 5V PIN8
    delay(1000);             //Set the delay time, 1000 = 1S
}

```

2.Using graphical code blocks to program to light the LED on Arduino UNO

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

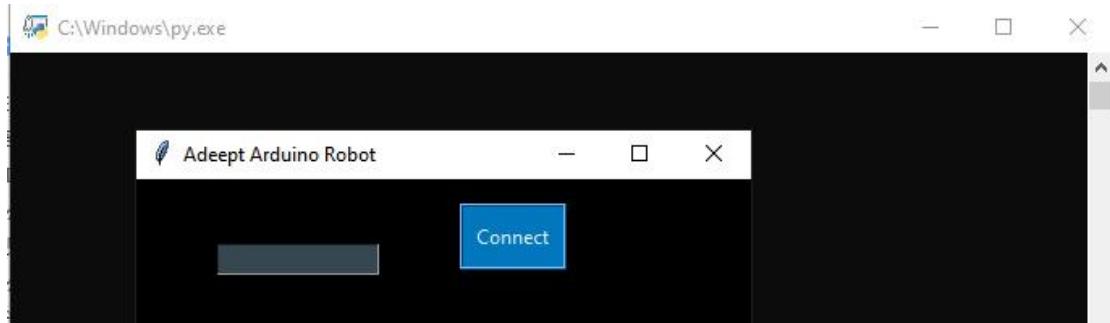
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

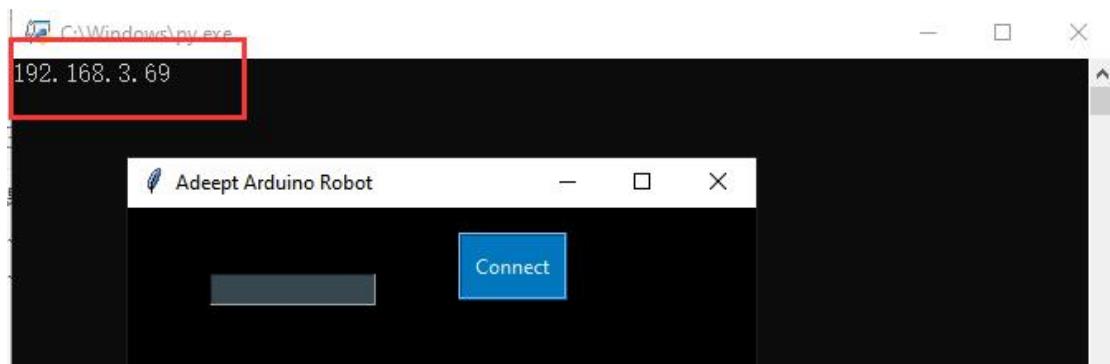
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum i:

1 Arduino Uno on COM4

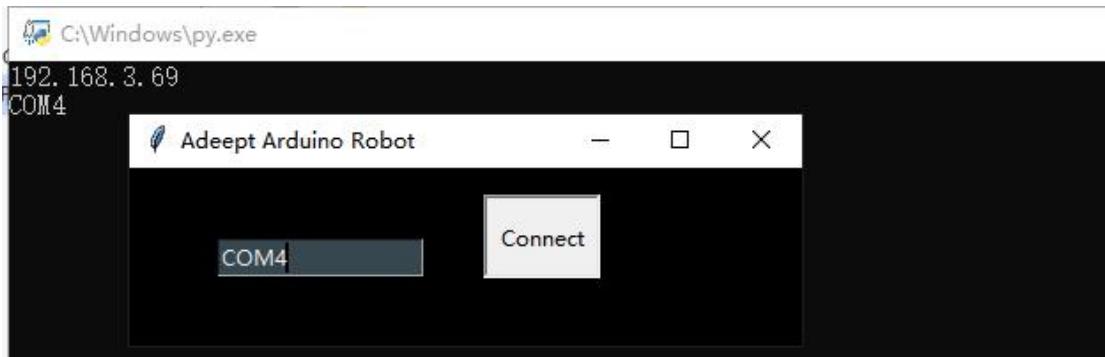
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



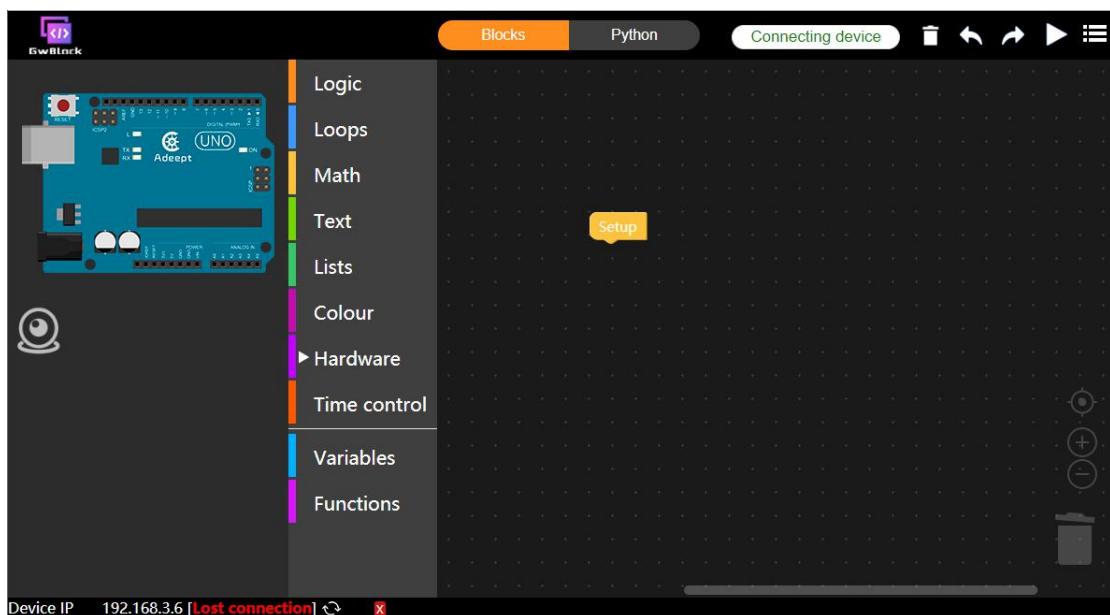
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



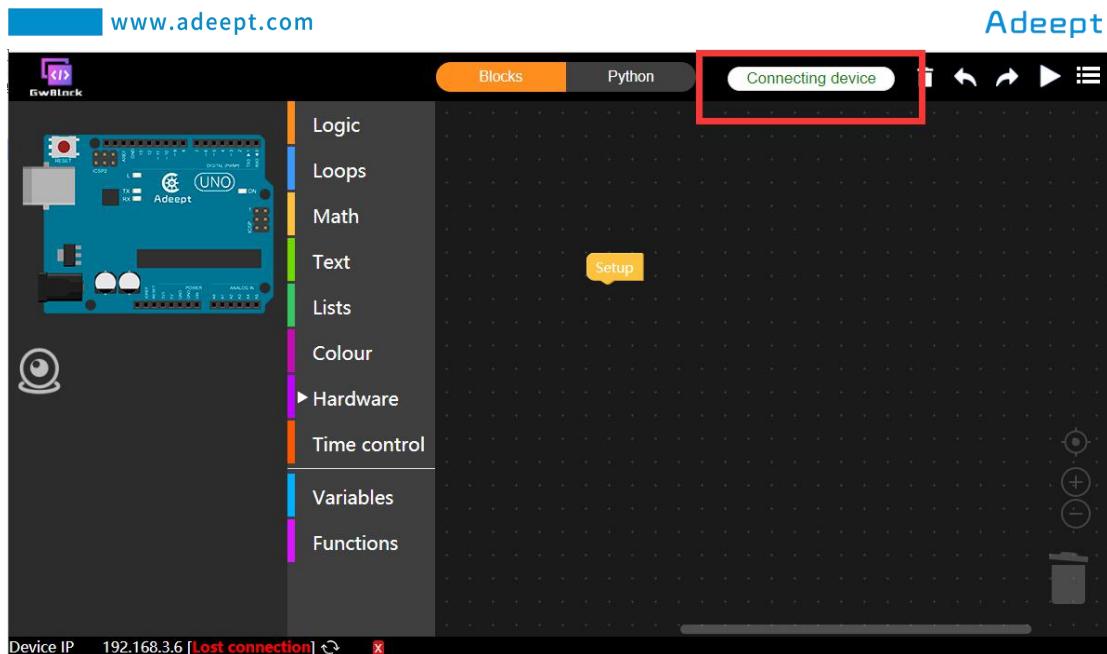
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

After successfully entering the website, the interface is as follows:



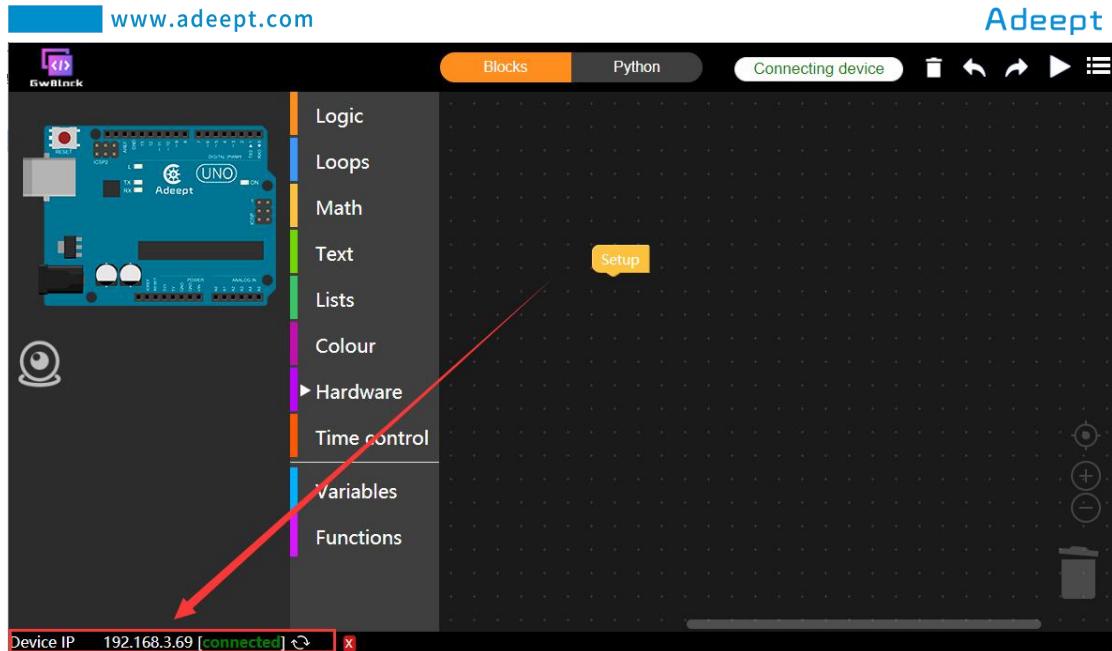
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



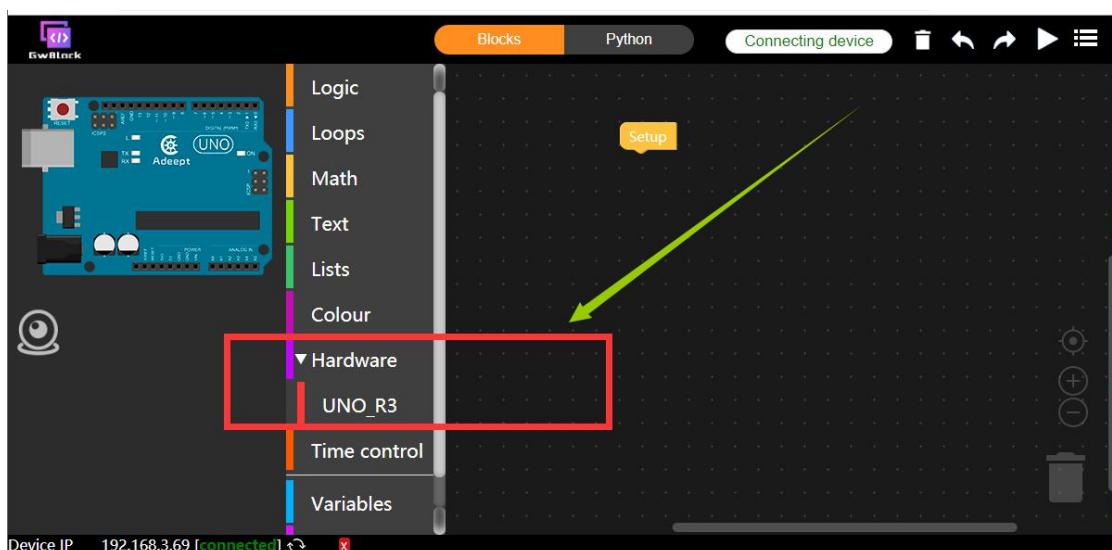
8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:



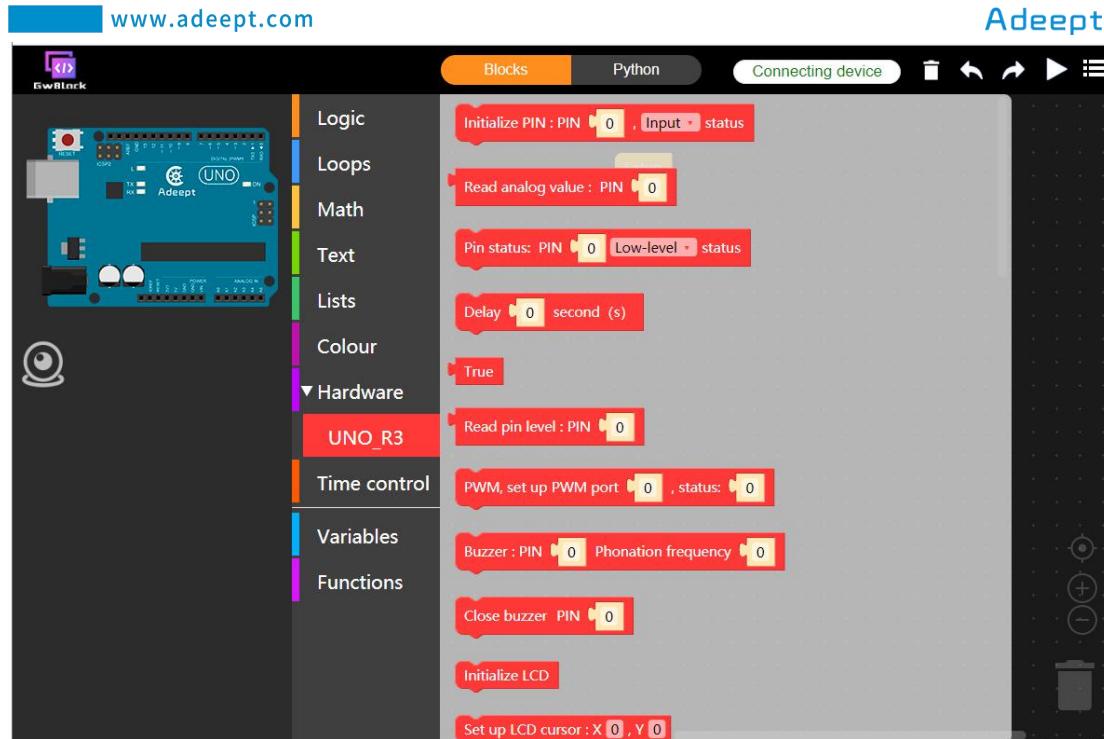
(2)Using graphical programming to light up the LED

Next let's learn how to use the graphical code block to program the LED on the Arduino UNO development board.

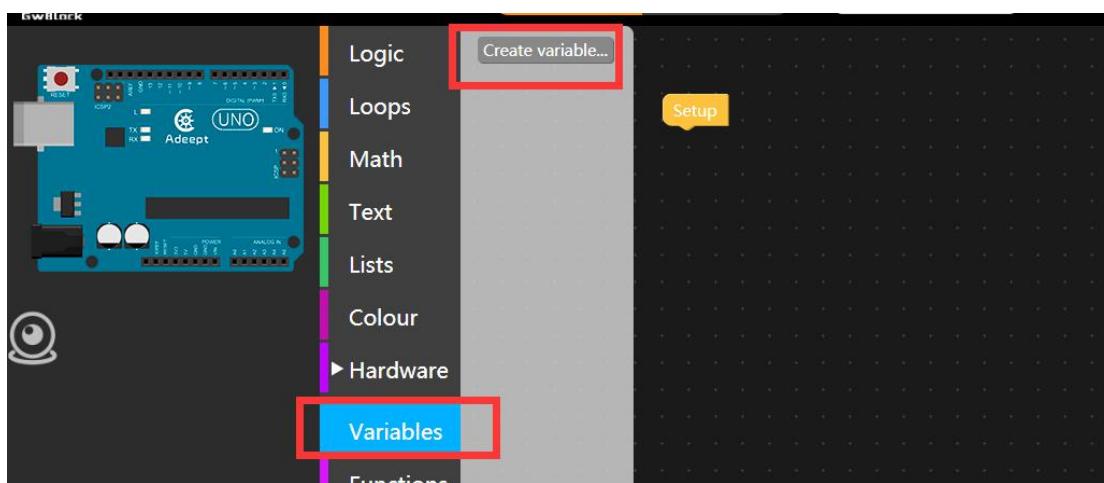
1. After successfully connecting to the GwBlock IDE, we find "Hardware" in the toolbar of the code instruction module in the middle, where the UNO_R3 code instruction module is a collection of graphical code blocks that control the Arduino UNO.



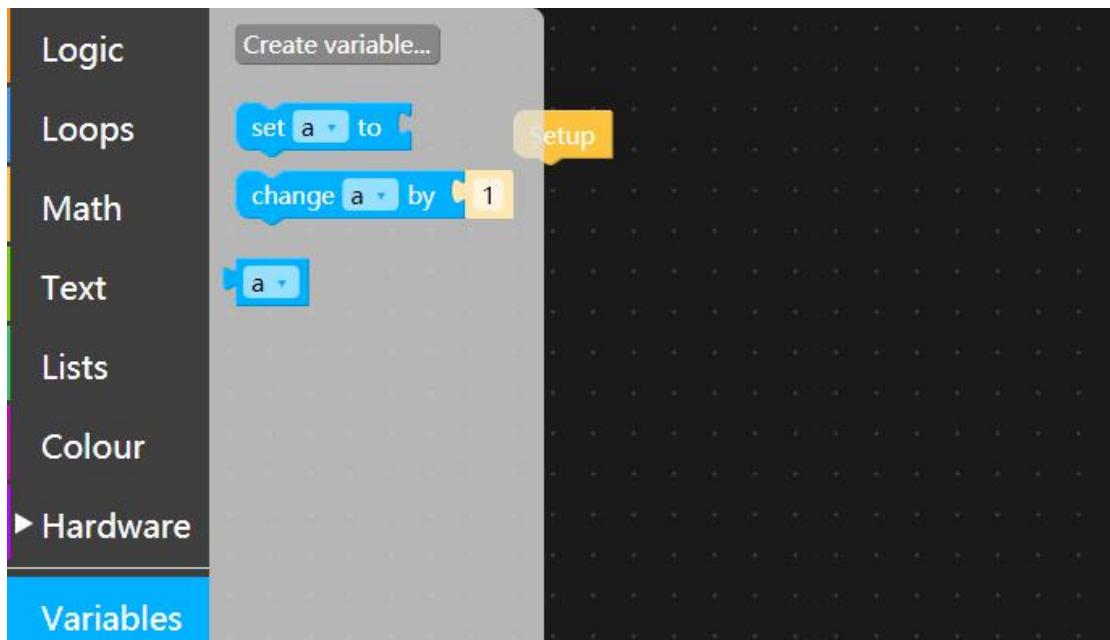
2. The opened UNO_R3 code instruction module is as shown in the figure below, and each graphical code block has different functions.



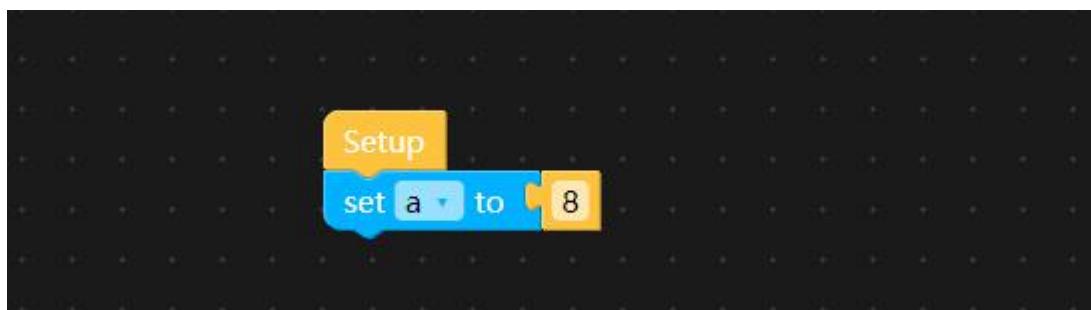
3. Click Create variable under Variables in the toolbar of the code instruction block to create variables.



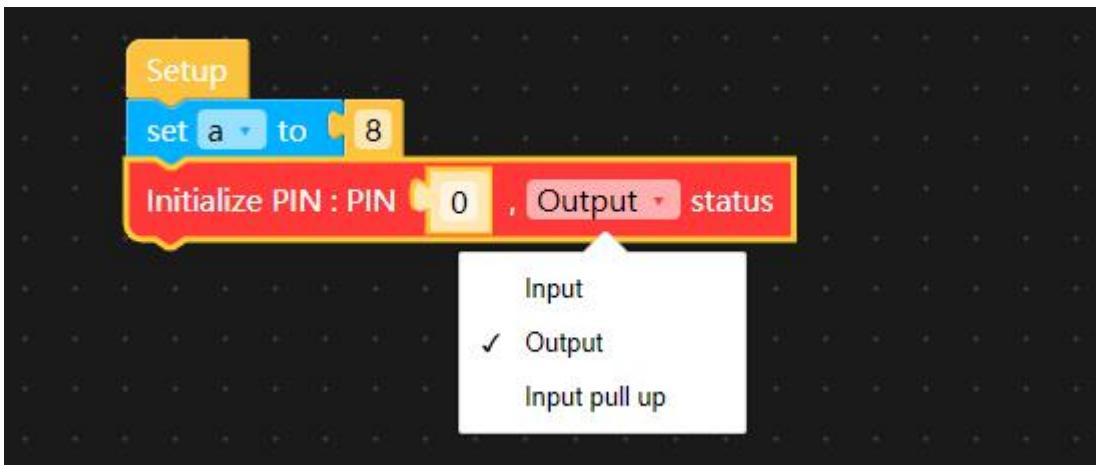
4. Create a variable a, after successful creation is as follows:



5. Drag **set a to** to the right below **Setup**. Find the instruction **+0** under the instruction module **Math**. Change the number "0" to 8. As shown below:



6. Then we need to initialize the LED pin to the output mode. Find **Initialize PIN : PIN 0 , Input status** under the code instruction module **UNO_R3**. Drag it to the right below the **set a to**. Change **Input** to Output. As shown below:



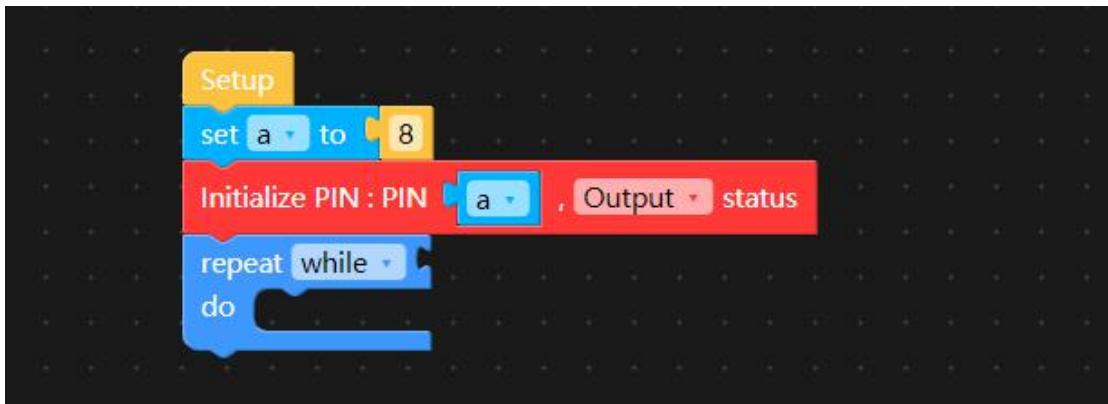
7. The variable a represents the pin number of the LED. You need to change the 0 of **PIN 0** to the variable a. Drag **a** and drop it into **0** under the toolbar of the code instruction block **Variables**, as shown below:



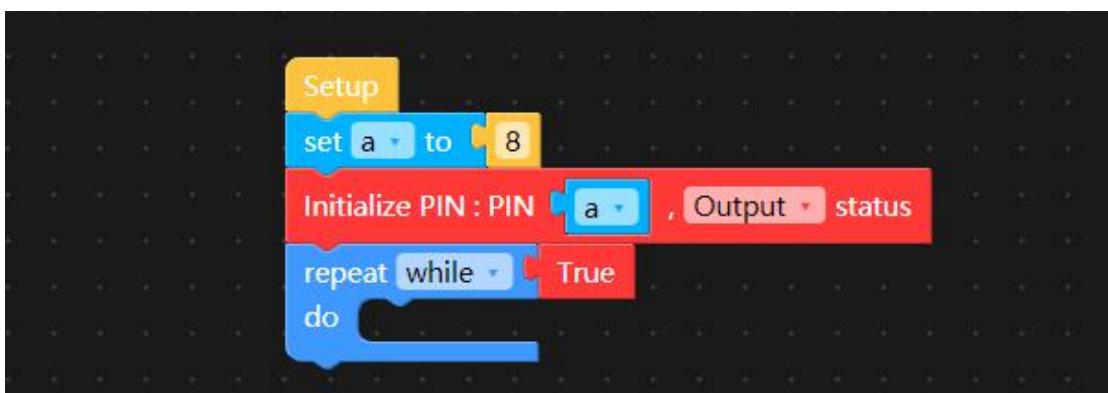
8. Find the loop judgment instruction **repeat [while] do** under the Loops code instruction module, as shown below:



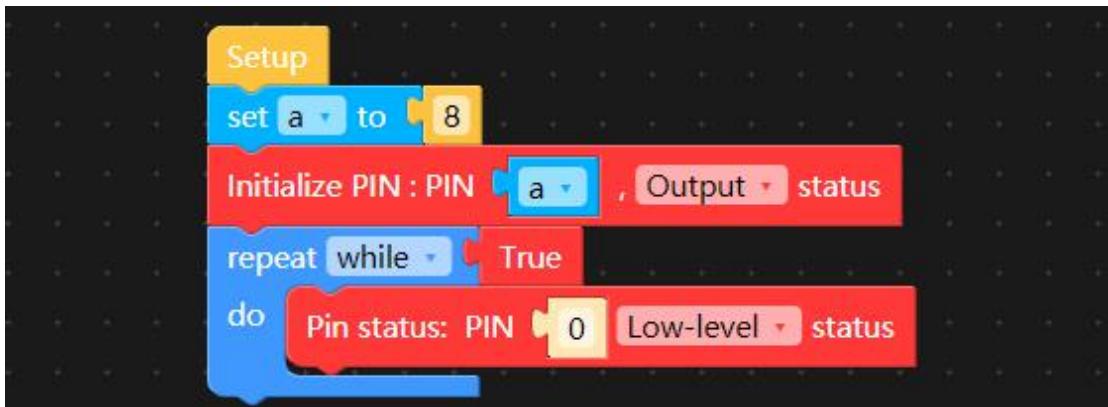
9. Drag **repeat [while] do** to the position as shown below:



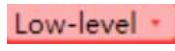
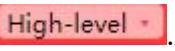
10. We need to determine whether the condition is TRUE, find  under the code instruction module , and drag it to the following location:

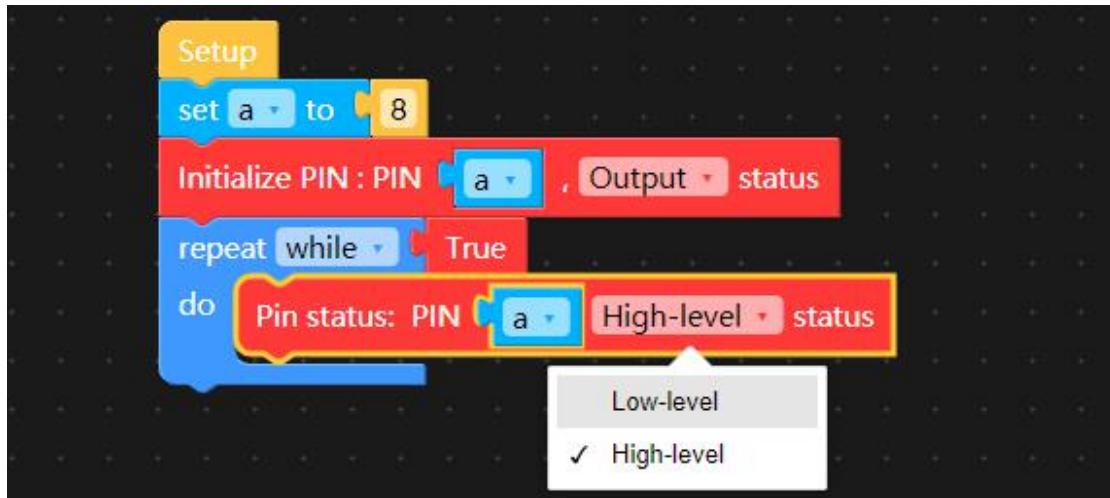


11. When the condition is True, control the LED on and off by setting the high and low levels of the LED pin. Find  under the code instruction module . This instruction block can set high and low level. Place it in the position of the following figure:

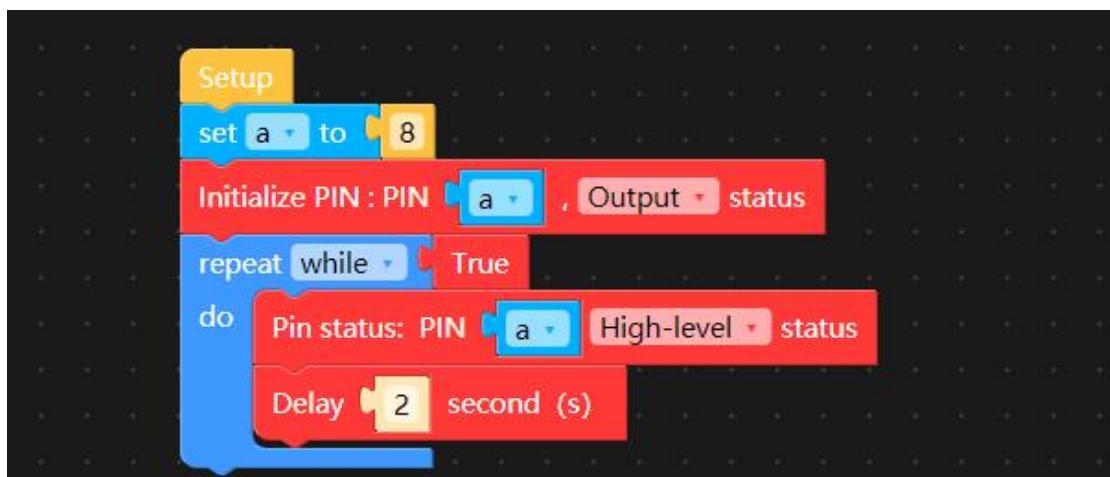


12. First set the LED pin to high level, and the LED will light up. Change  to

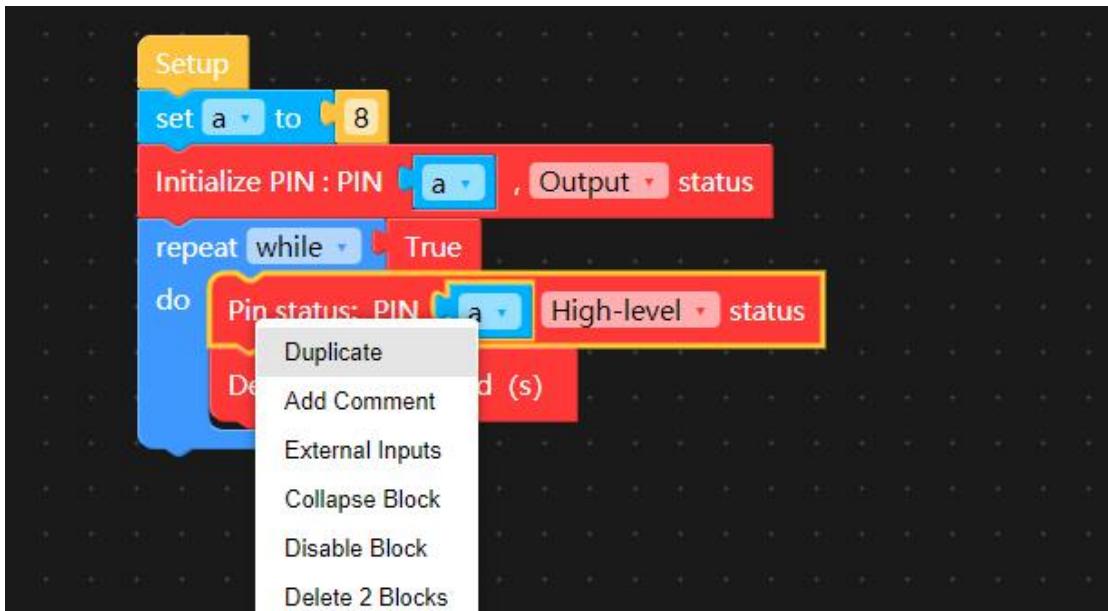
the variable a:  .Change  to  .As shown below:



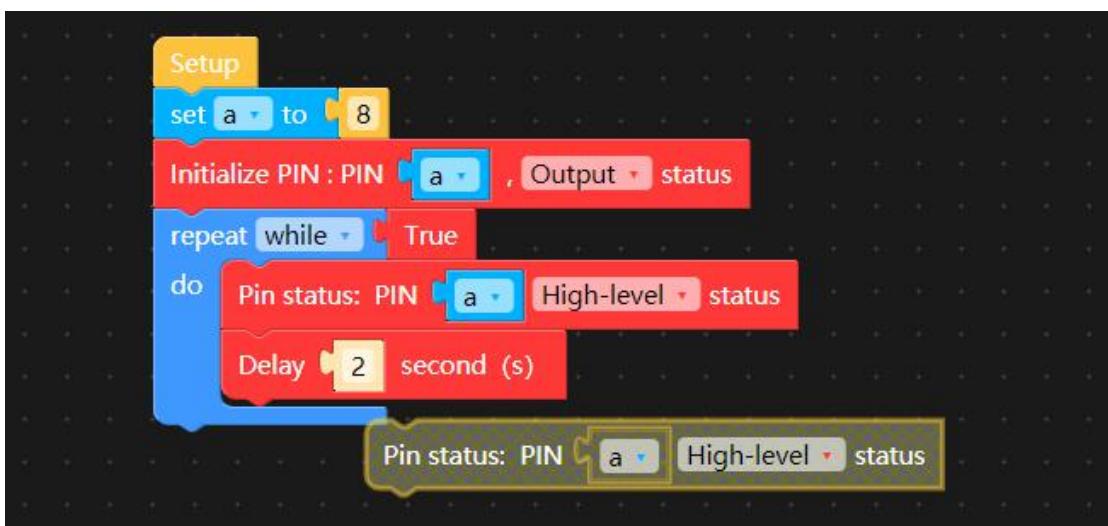
13. We need to use a delay command to control the LED to continue to light for a period of time. Find  under the code command module  .And then you can enter the number of time to delay in  . We enter 2 to control the LED to keep on for 2 seconds. As shown below:



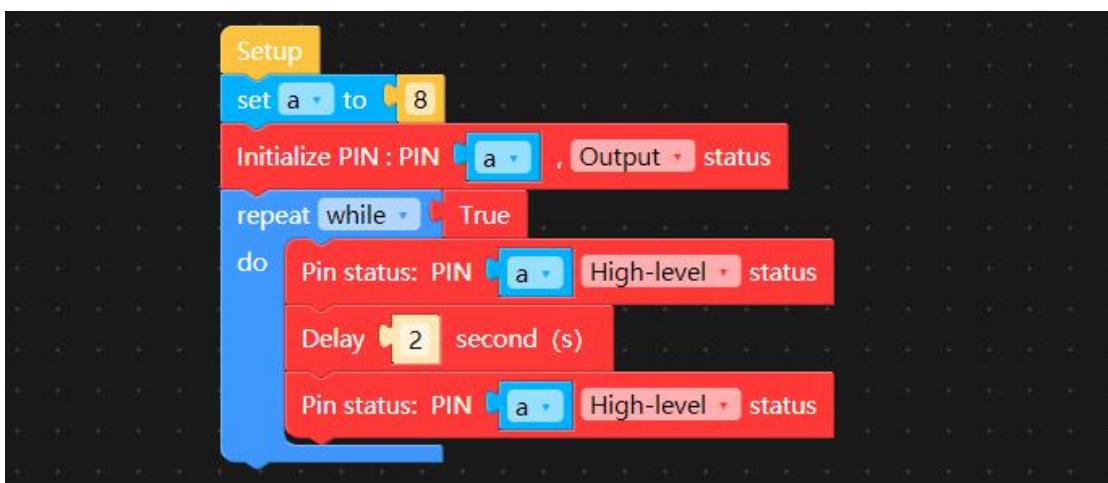
14. When we set the LED pin to low level, the LED will be off. Use the mouse to move to the location  .Click the right button.Select "Duplicate".As shown below:



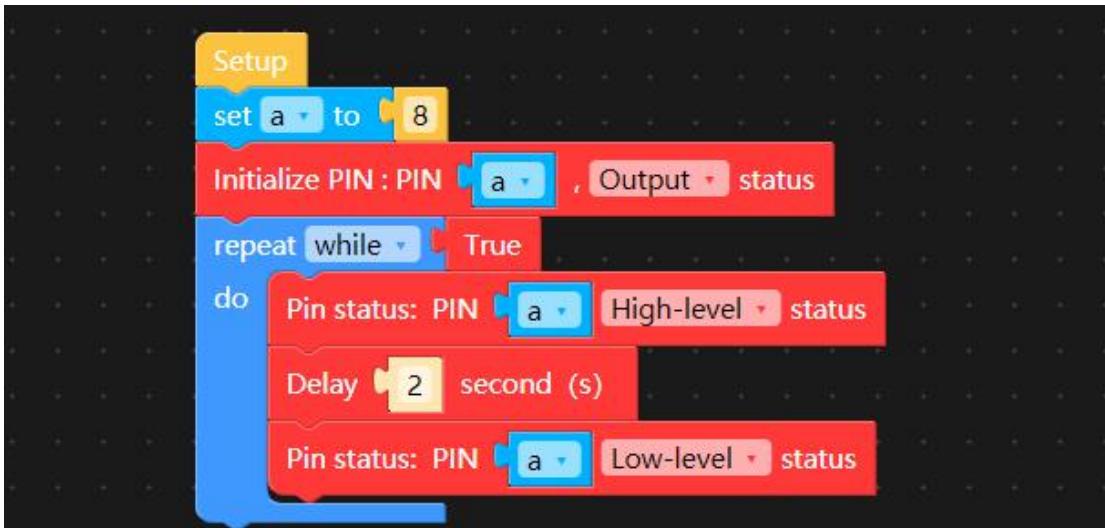
It will copy and generate an identical code instruction block, as shown below:



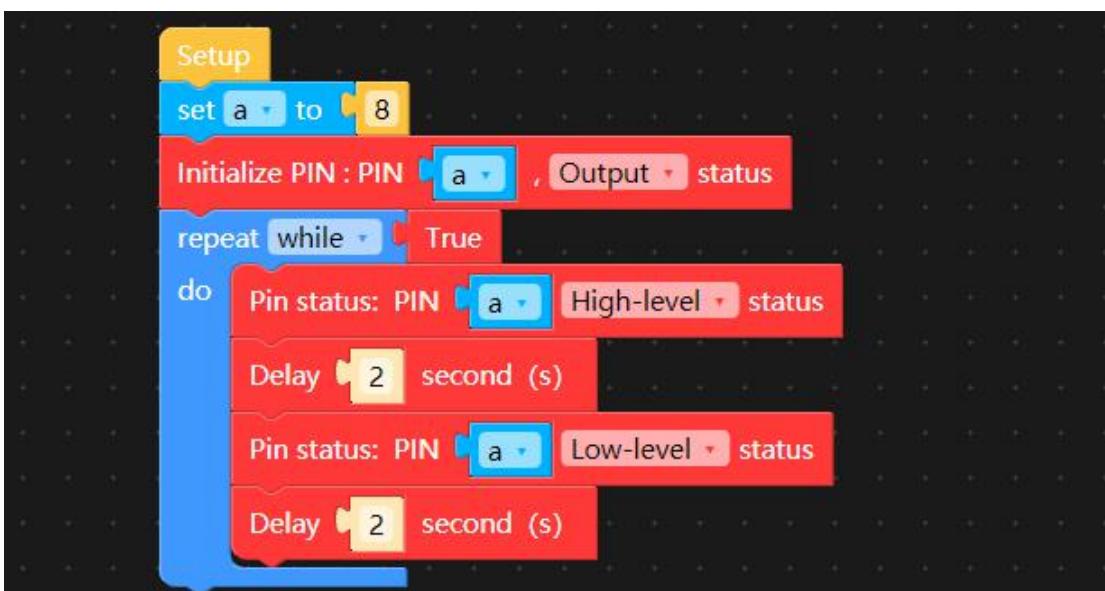
15. Place the copied code instruction block as shown below:



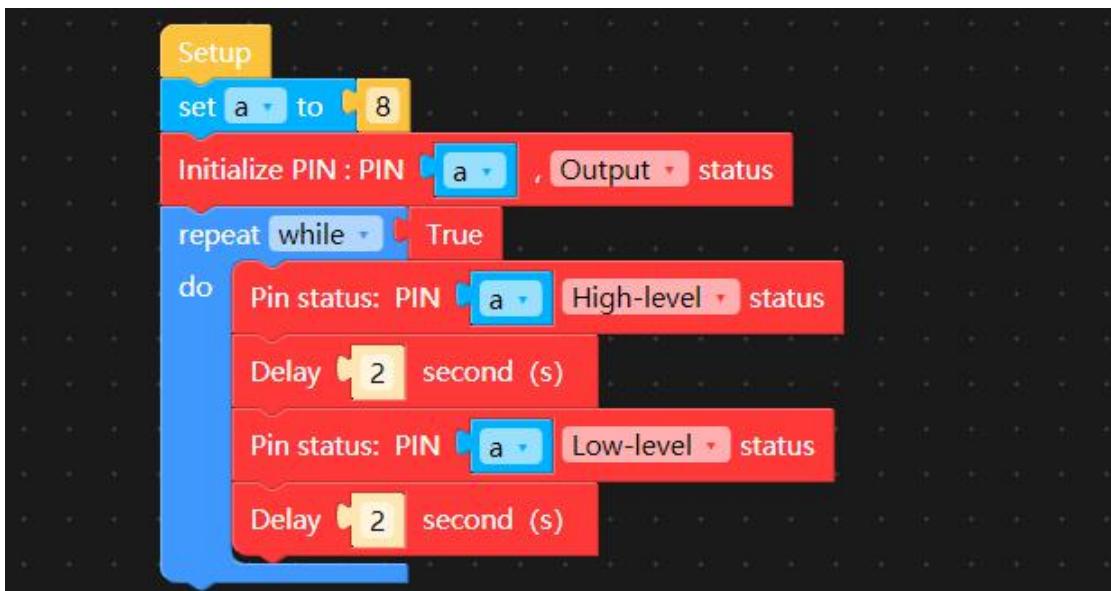
16. To control the LED to be off, we only need to change the **High-level** to **Low-level**. As shown below:



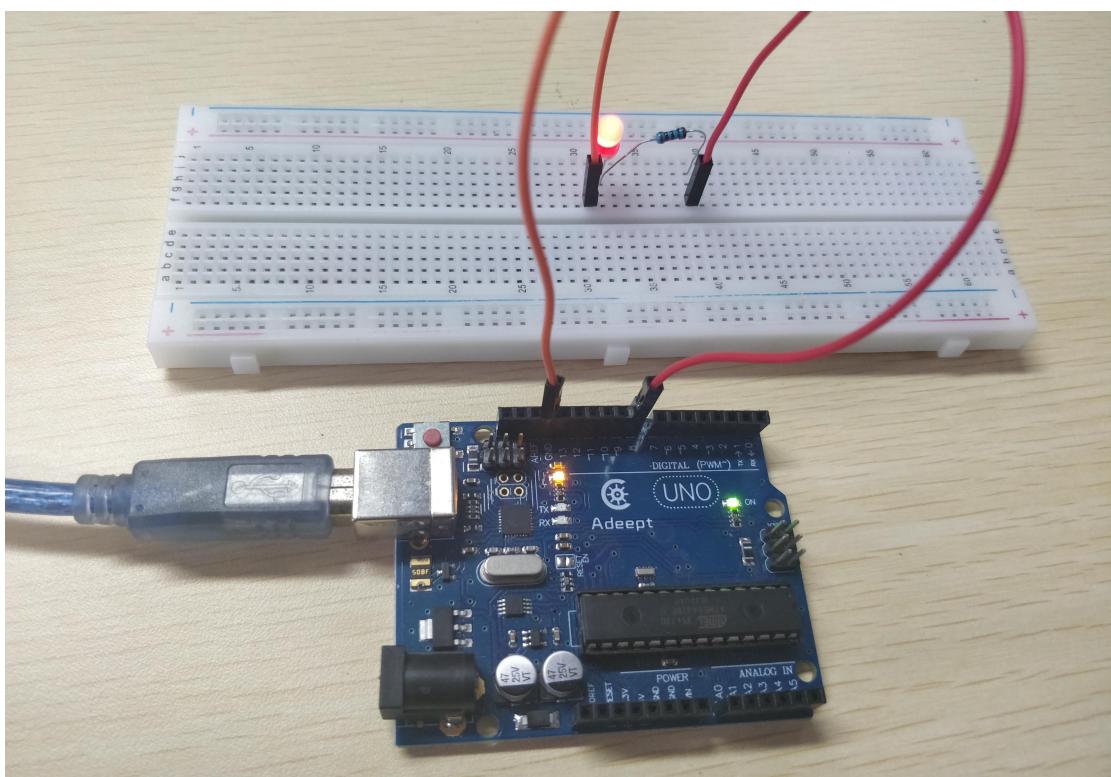
17. We need to use another delay command to control the LED to be off for a period of time. Find **Delay [0] second (s)** under the code instruction module **UNO_R3**. And then you can enter the number of the time to delay in **[0]**. We enter 2 to control the LED to be off for 2 seconds. As shown below:



18. The final program is as shown in the figure below. Click  in the upper right corner. Pay attention to observe the LED. It indicates successful if the LED is on then off.



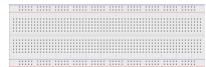
19. The physical connection of the experiment is as follows:



Lesson 2 The Application of the Active Buzzer

In this lesson, we will study the application of the Active Buzzer.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(1KΩ)	1	
Active Buzzer	1	
NPN Transistor(8050)	1	

2. The introduction of the Buzzer

(1) The Buzzer

The Buzzer is an electronic sounder with an integrated structure. It is powered by DC voltage and is widely used as a sounding device in electronic products such as computers, printers, copiers, alarms, electronic toys, automotive electronic equipment, telephones, timers, and other electronic products. . There are two types of buzzer: active buzzer and passive buzzer. As shown in the figure below, the left is the active buzzer (the two pins have different lengths), and the right is the passive buzzer (the two pins have the same length).



(2) Working principle of the Buzzer

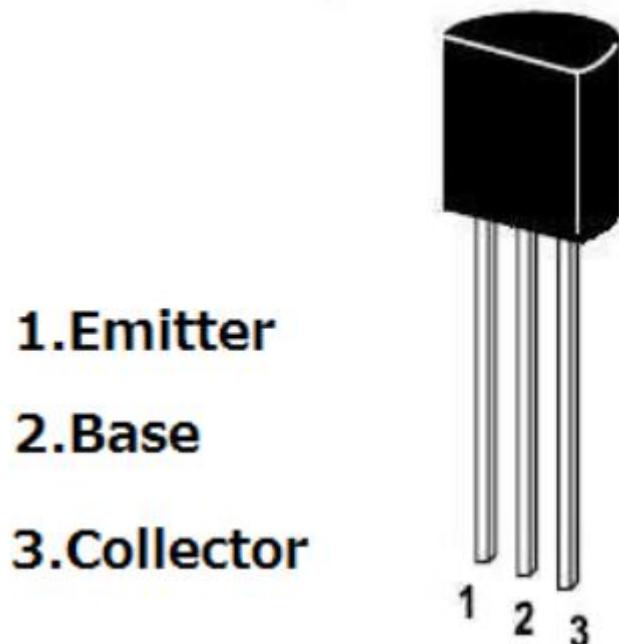
The sounding principle of buzzer is composed of vibration device and resonance device, and buzzer is divided into passive buzzer and active buzzer. The working sounding principle of passive buzzer is: square wave signal input resonant device is converted into sound signal output; the working sounding principle of active buzzer is: DC power input is generated by the amplification sampling circuit of the oscillation system under the action of the resonance device Sound signal. Our course in this section uses an active buzzer. As long as the power is on, the active buzzer will sound. We can program the Arduino UNO output high and low alternately, so that the active buzzer will sound.

(3) Two kinds of Transistors (S8050 and S8550)

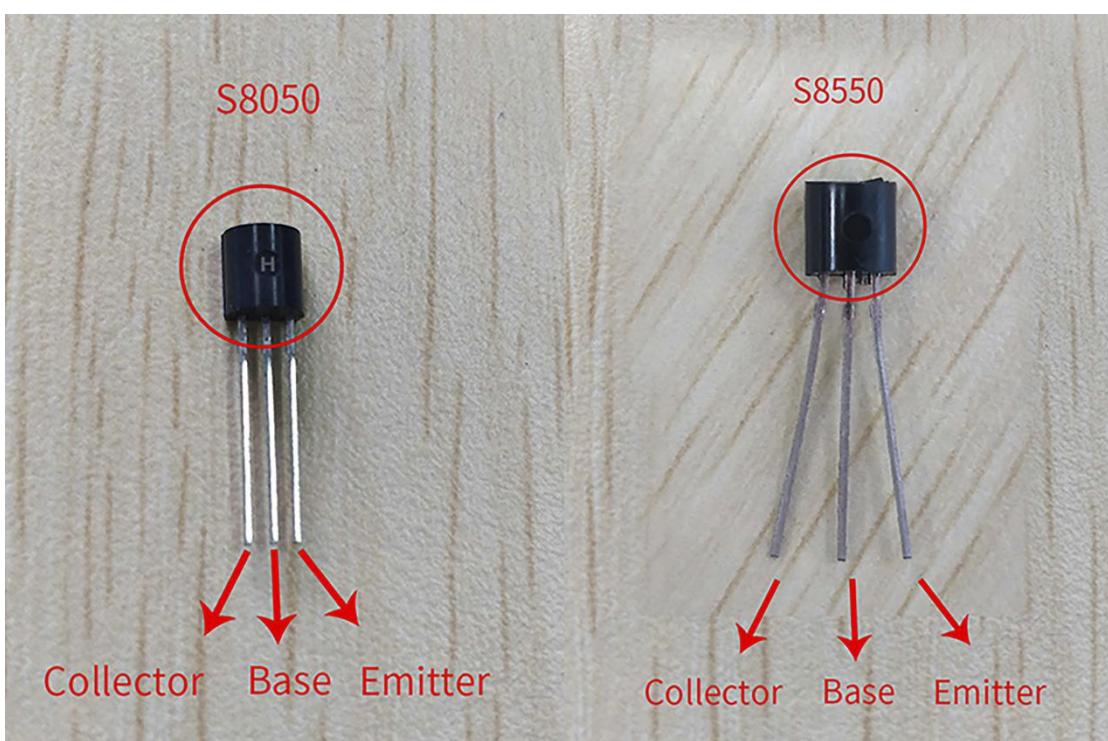
To make the active buzzer sound, a large current is required. However, the output current of Arduino UNO development board is very weak, so we need a transistor S8050 or S8550 to drive the active buzzer. The main function of the transistor S8050 (S8550) is to amplify the voltage or current, and it can also be used to control the conduction or cut-off time of the circuit.

There are two kinds of transistors, one is NPN, such as the transistor S8050 used in our course; the other is a PNP transistor, such as the other S8550 we provide. The pin structure of the two transistors we use is the same. Their pin structure is as shown

in the figure below. In the circuit, Emitter is abbreviated as e, Base is abbreviated as b, and Collector is abbreviated as c.



The S8050 and S8550 transistors provided by our course are as shown below. The letter H is S8050, and the letter H is S8550.



The transistors S8050 and S8550 and the buzzer are connected in the circuit as shown below:

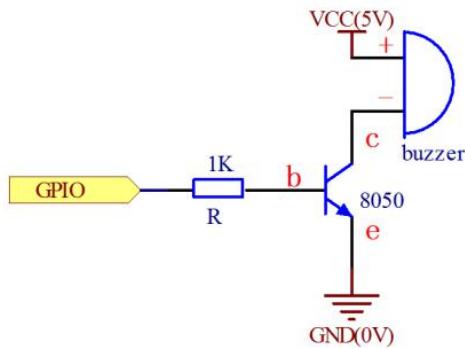


Figure1

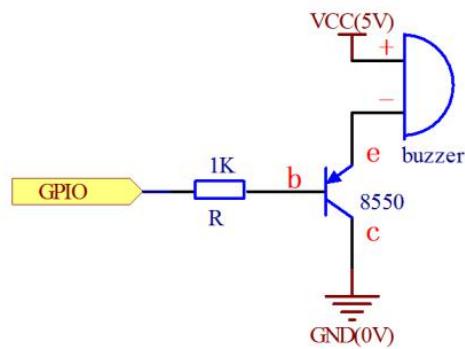


Figure2

Figure1:

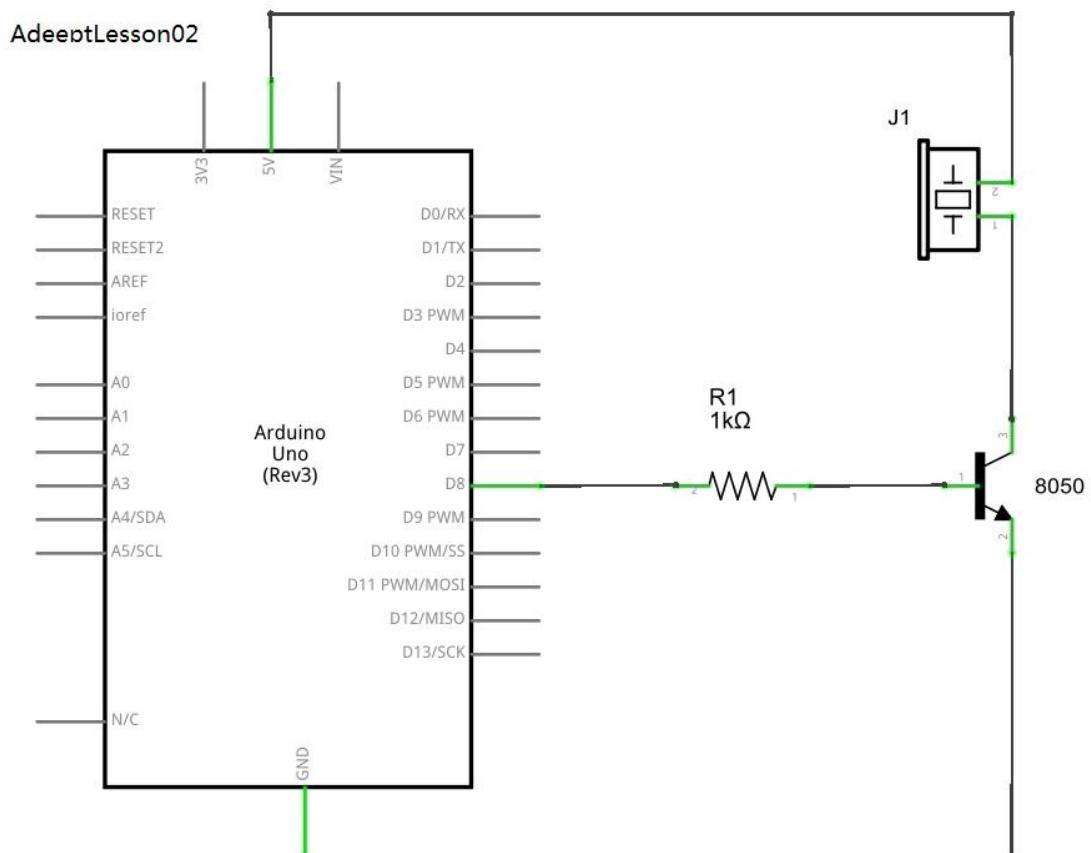
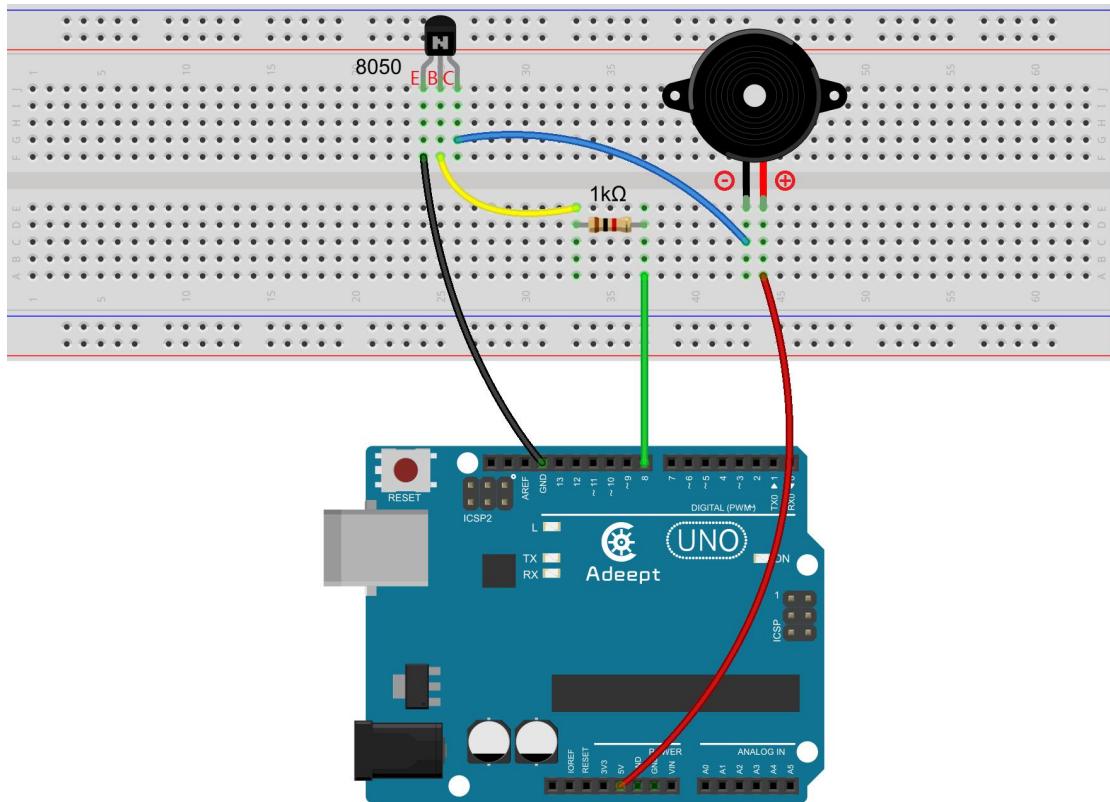
Set the Arduino UNO as a high level, the transistor S8050 will conduct, and then the buzzer will sound; set the Arduino UNO as low level, the transistor S8050 will cut off, then the buzzer will stop.

Figure2:

Set the Arduino UNO as low level, the transistor S8550 will conduct, and the buzzer will sound; set the Arduino UNO as a high level, the transistor S8550 will cut off, then the buzzer will stop.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connected them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes, as shown in the following figure:



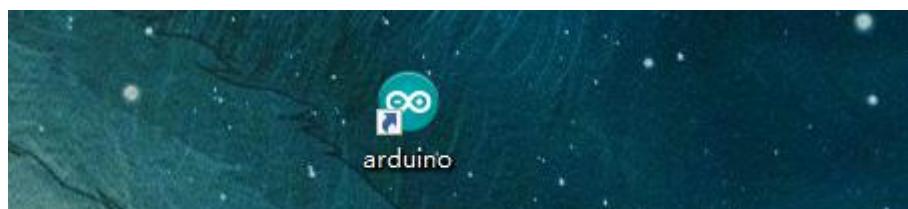
4. The application of the Active Buzzer

We provide two different methods to control the active buzzer. One is to program the active buzzer on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program the active buzzer on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

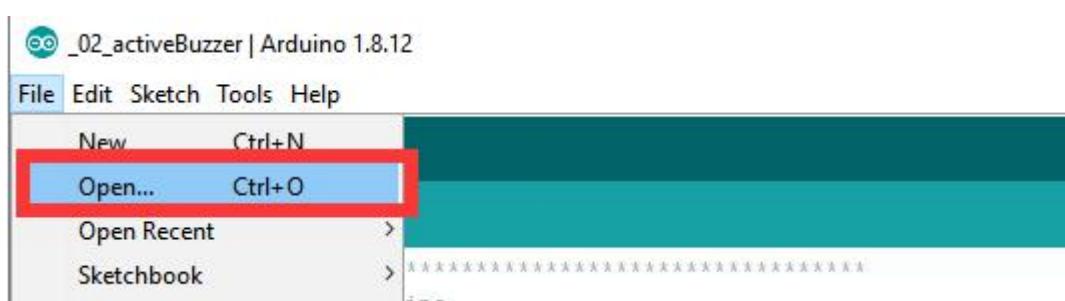
1. Programming and controlling the Active Buzzer in C language on Arduino UNO

(1) Compile and run the code program of this course

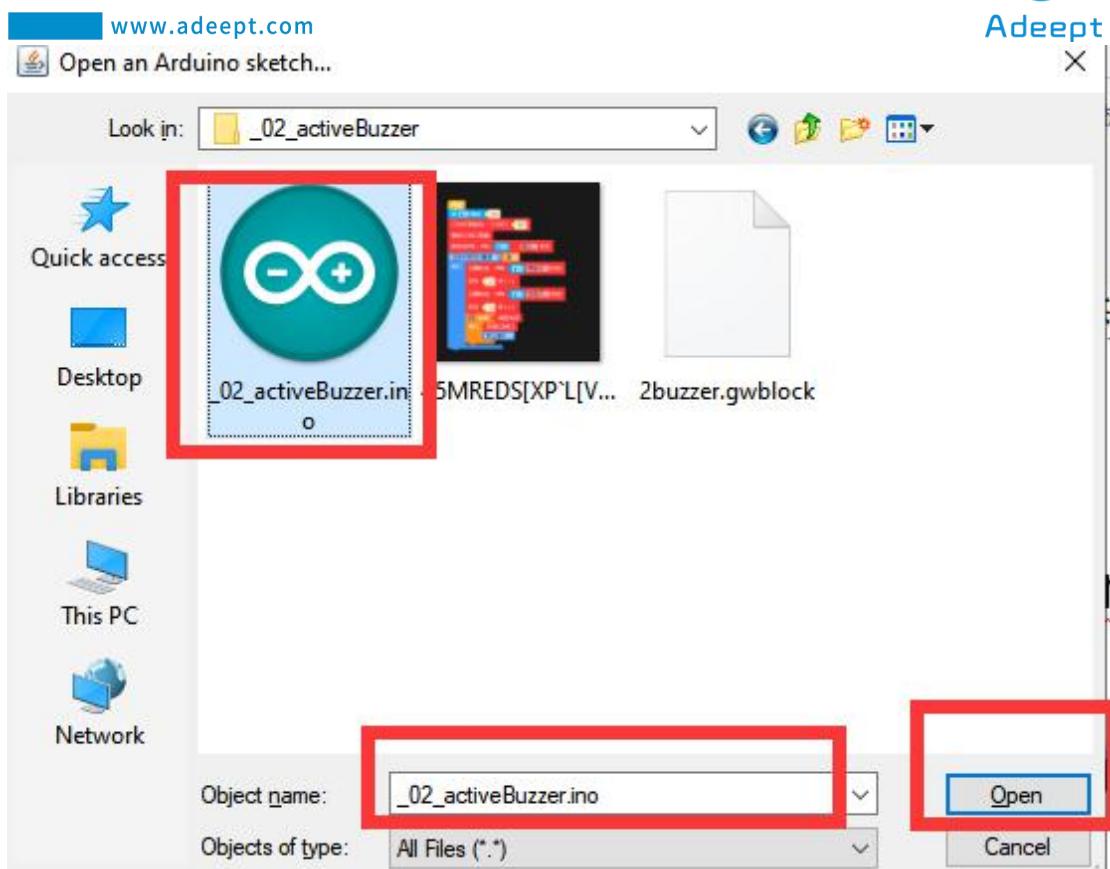
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\02_activeBuzzer directory. Select _02_activeBuzzer.ino. This file is the code program we need in this course. Then click Open.



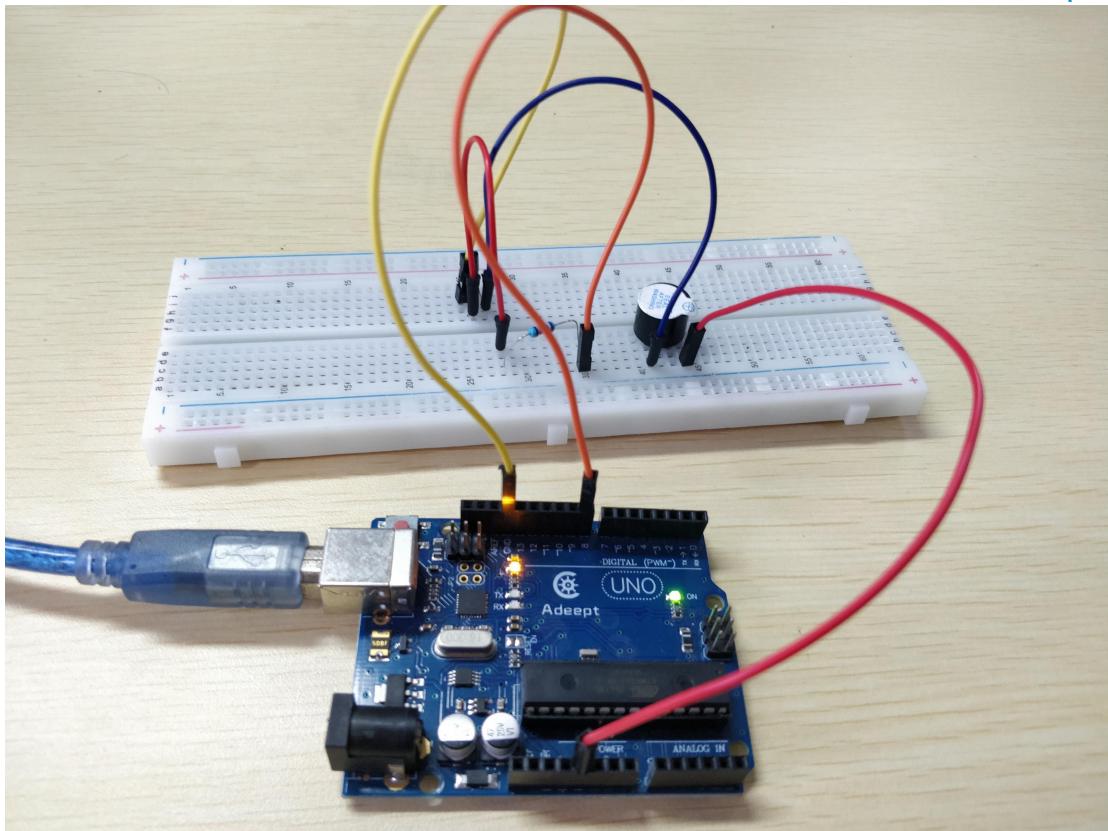
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 936 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1 Arduino Uno on COM4
```

5. If the program upload is successful, the active buzzer will sound, indicating that the experimental test was successful. The physical connection diagram of the experiment is as below:



(1) The core code program

After the above hands-on operation, you must be very interested to know how we make the Active Buzzer sounded with C language on Arduino UNO. We will introduce how our core code can be achieved:

1.Define the pin of the active buzzer by int buzzerPin=8. Then connect it to port 8 on the Arduino UNO. In the setup () method, set buzzerPin as the output mode through pinMode (buzzerPin, OUTPUT).

```
int buzzerPin=8; //definition digital 8 pins as pin to control the buzzer
void setup()
{
    pinMode(buzzerPin,OUTPUT); //Set digital 8 port mode, the OUTPUT for the output
}
```

2.In loop() method, set buzzerPin to HIGH state through digitalWrite(buzzerPin, HIGH). At this time the active buzzer sounds. Then set the active buzzer to delay 2s to sound through delay (2000). Set buzzerPin to LOW state through digitalWrite (buzzerPin, LOW) after 2s. At this time the active buzzer is off. Then set the active buzzer to delay 2s to be off through delay (2000).

```

void loop()
{
    digitalWrite(buzzerPin,HIGH); //Set PIN 8 feet as HIGH = 5 v
    delay(2000);                //Set the delay time, 2000ms
    digitalWrite(buzzerPin,LOW); //Set PIN 8 feet for LOW = 0 v
    delay(2000);                //Set the delay time, 2000ms
}

```

2.Programming and controlling the Active Buzzer on Arduino UNO with graphical code blocks

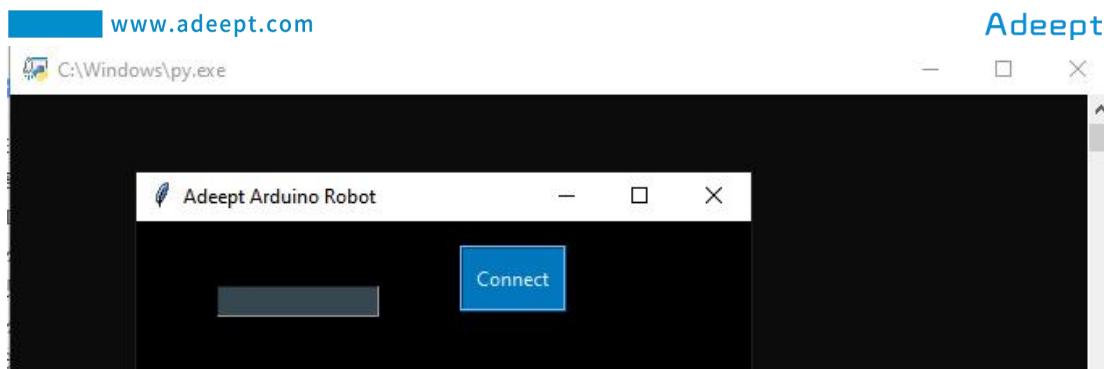
(2)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

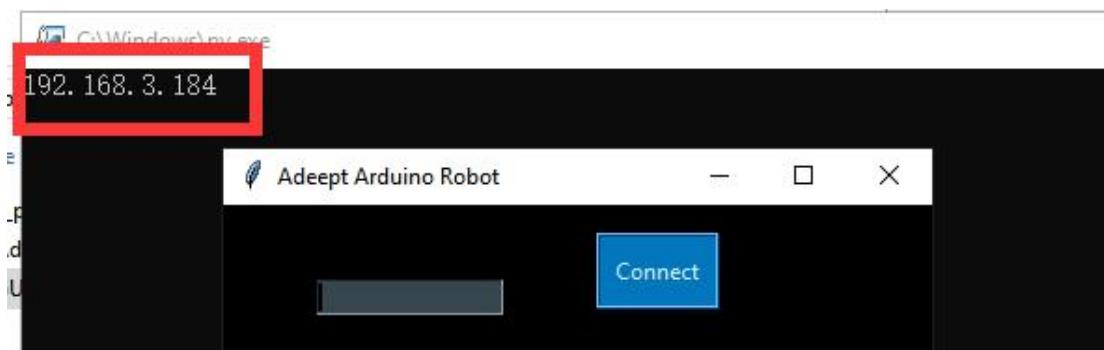
1.Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (opened with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:



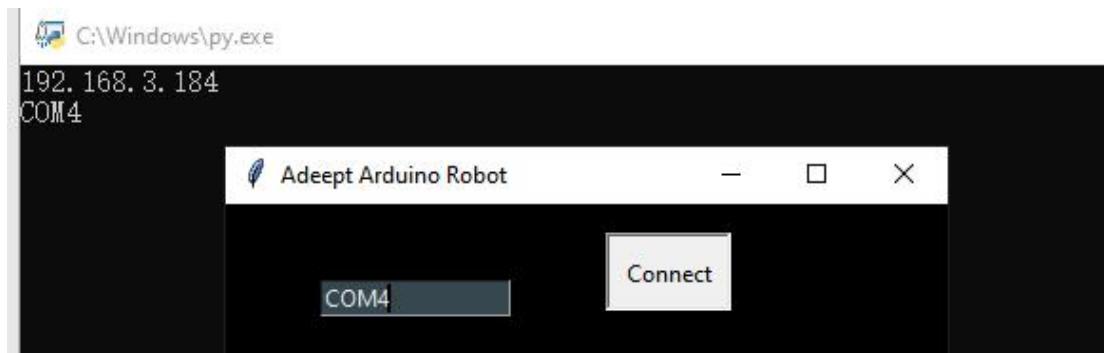
2.Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again.Then open the websocket folder. Double-click to open the GUI info v1.0.py file.



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



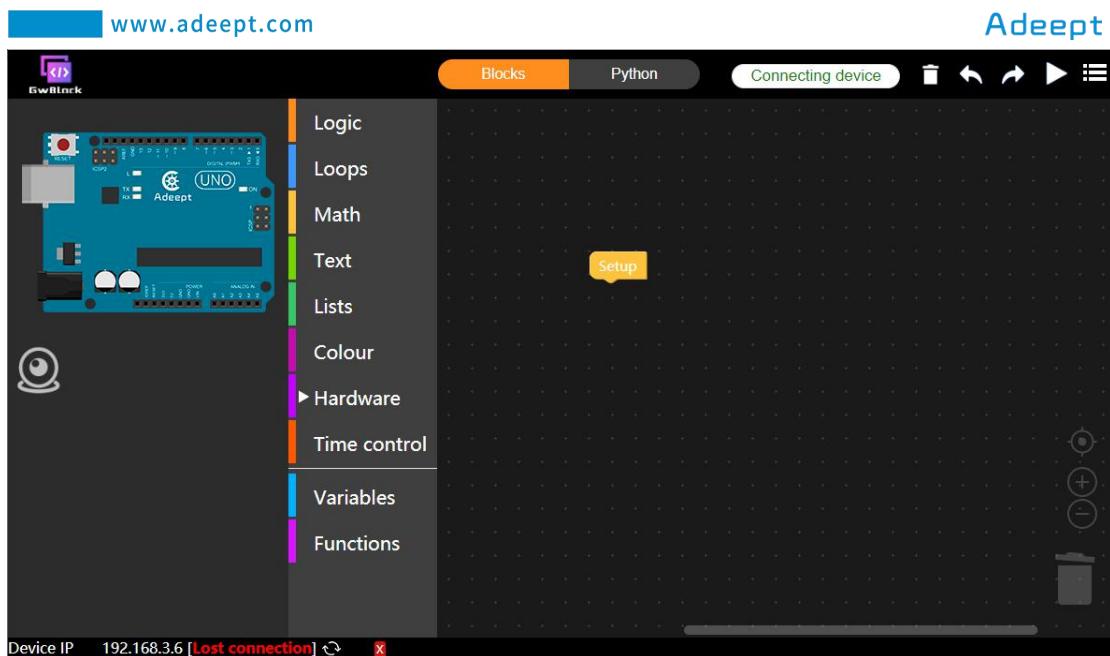
4. Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



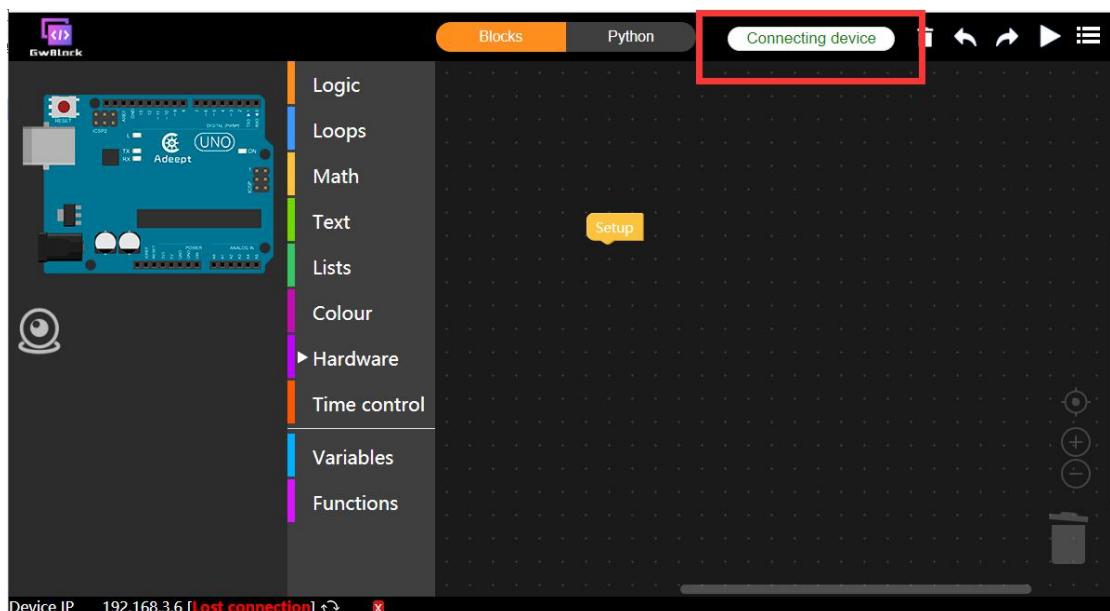
5. Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

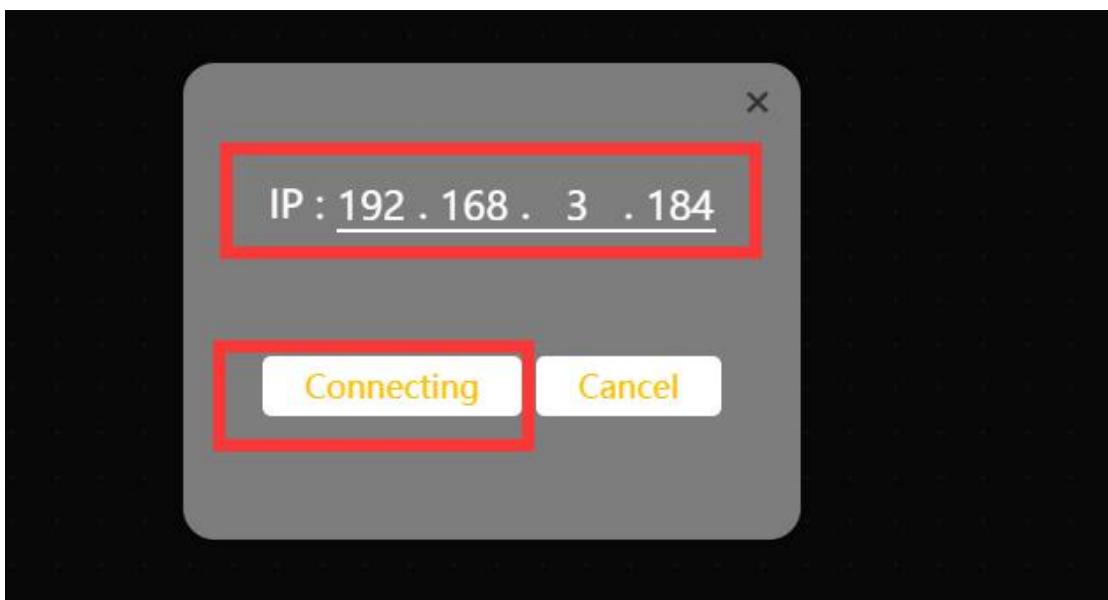
After successfully entering the website, the interface is as follows:



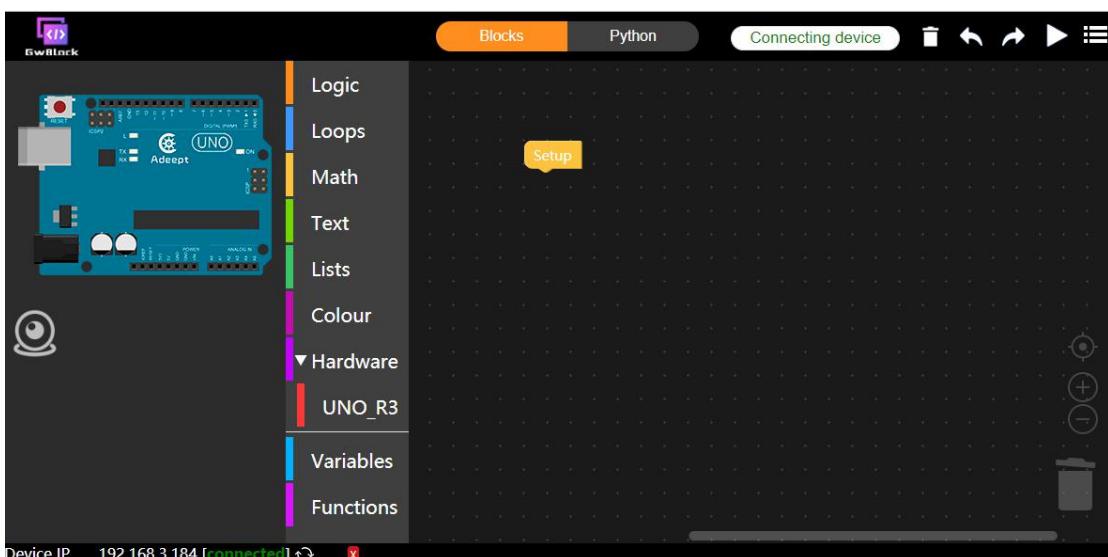
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



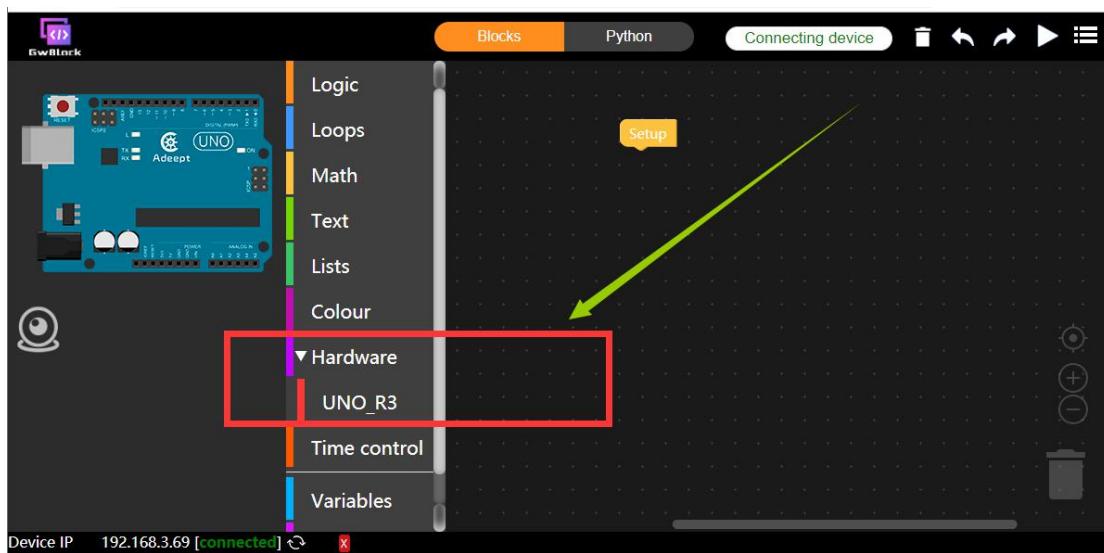
8. After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:



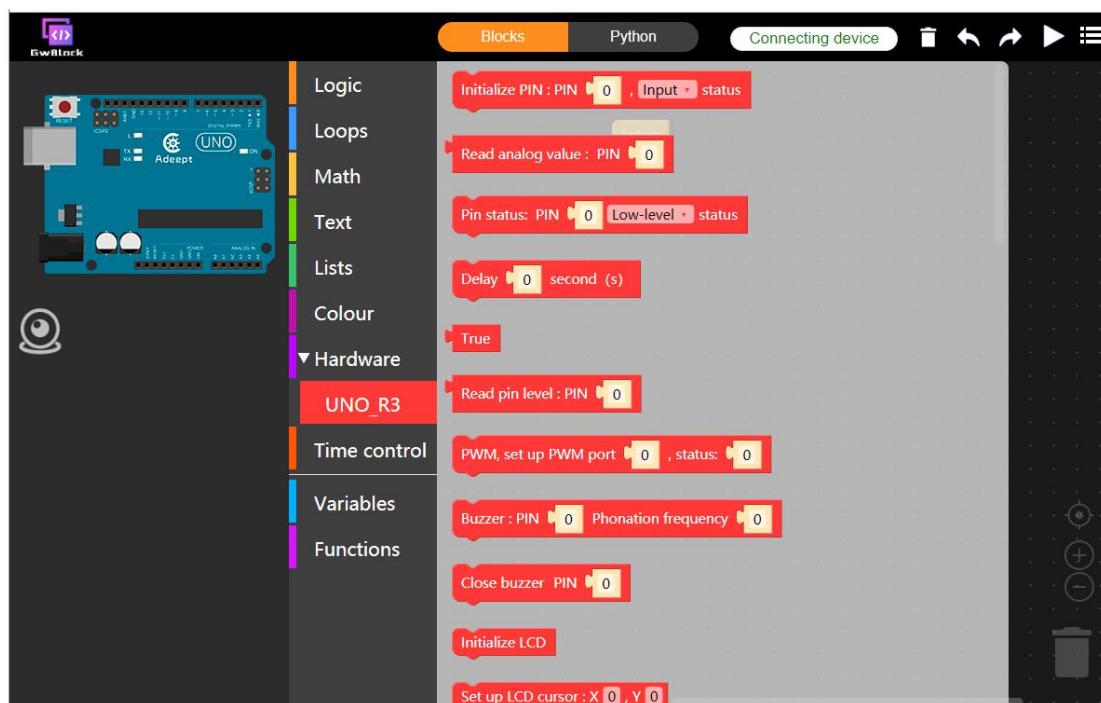
(2)Controlling the Active Buzzer

Now let us learn how to use the GwBlock graphical editor to program the active buzzer on the Arduino UNO.

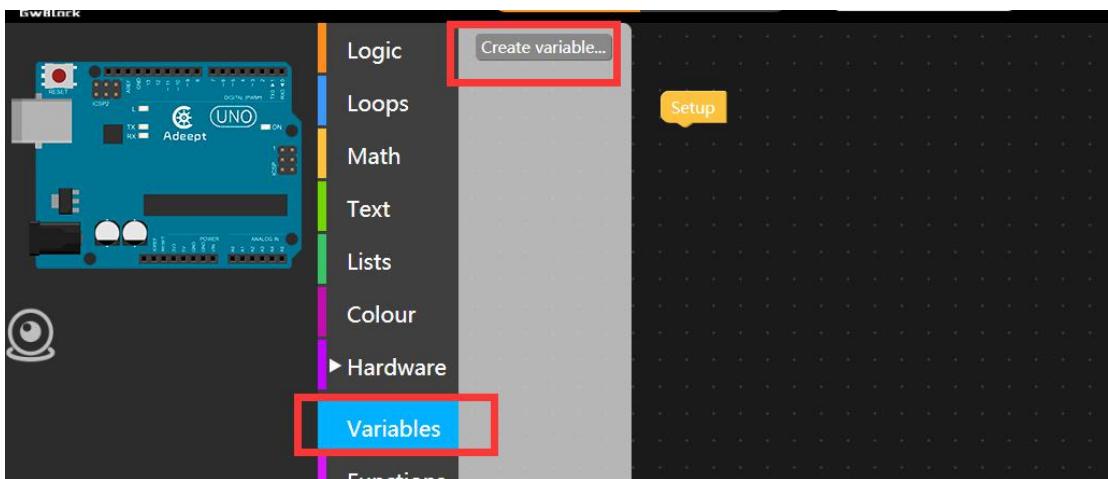
1. After successfully connecting to the GwBlock IDE, we find "Hardware" in the toolbar of the code instruction module in the middle. The UNO_R3 code instruction module is a collection of graphical code blocks that control the Arduino UNO.



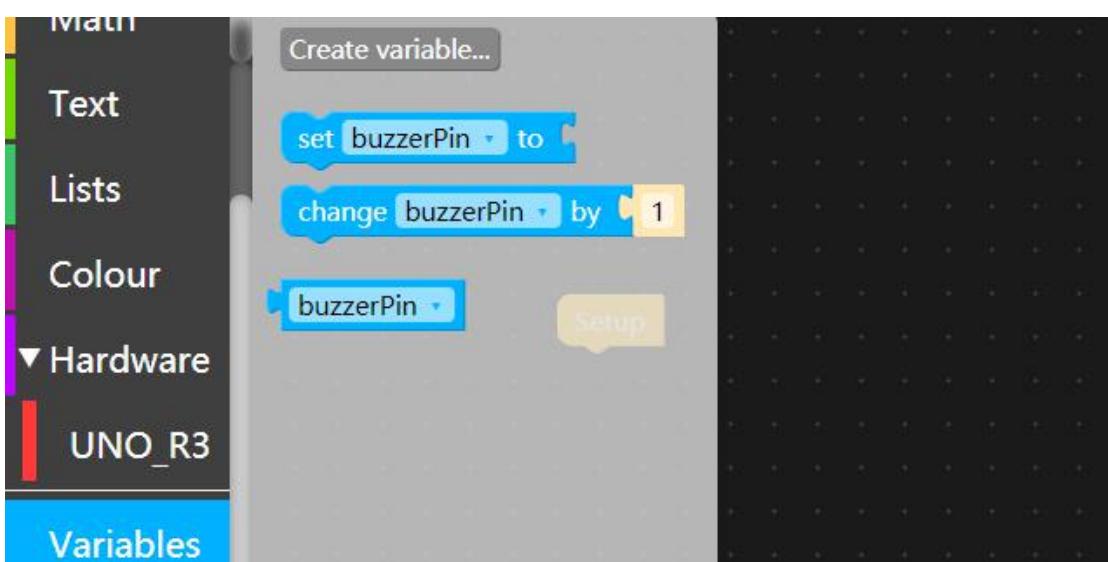
2.The code instruction module **UNO_R3** after opening is as shown in the figure below, and each graphical code block has different functions.



3.Click Create variable under Variables in the toolbar of the code instruction block to create variables.

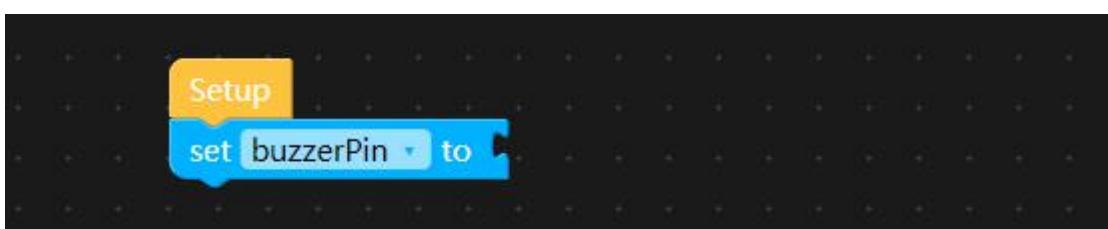


4.Create the variable `buzzerPin`. It will show as below:

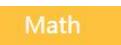


5.The variable `buzzerPin` represents the pin number of the active buzzer. Drag

`set buzzerPin to []` to the position as shown as below. It will show as below:

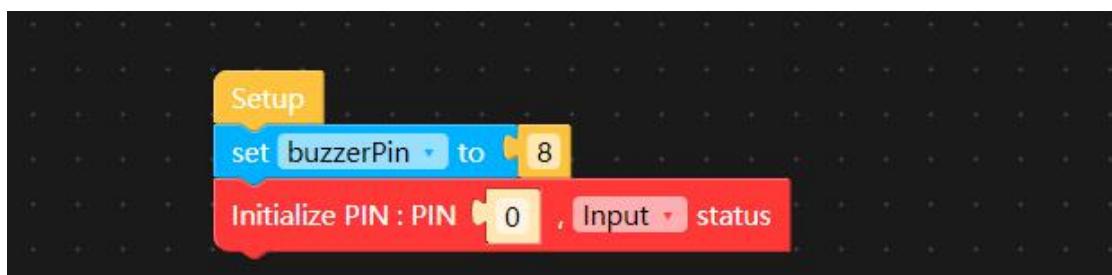


6.By knowing that the pin of the active buzzer is connected to port 8 in "3. Wiring

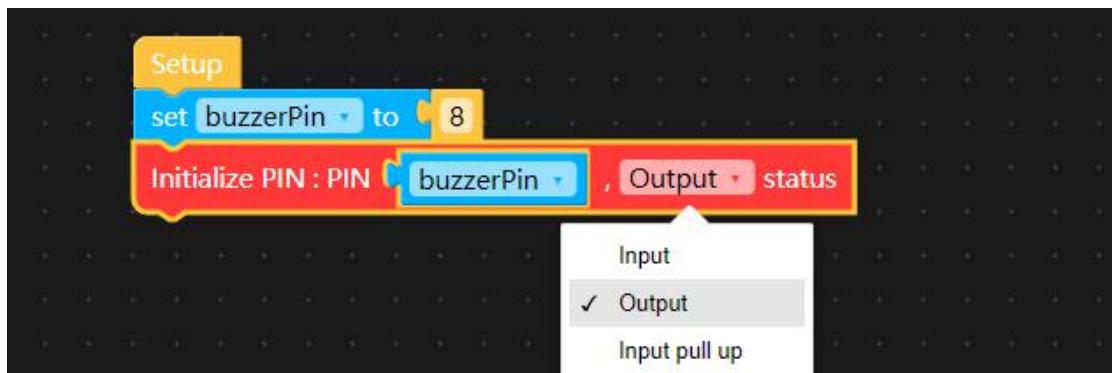
Diagram", we need to change `buzzerPin` to 8. Find the instruction `[0]` in the instruction bar  . The number "0" can be modified to 8. It will show as below:



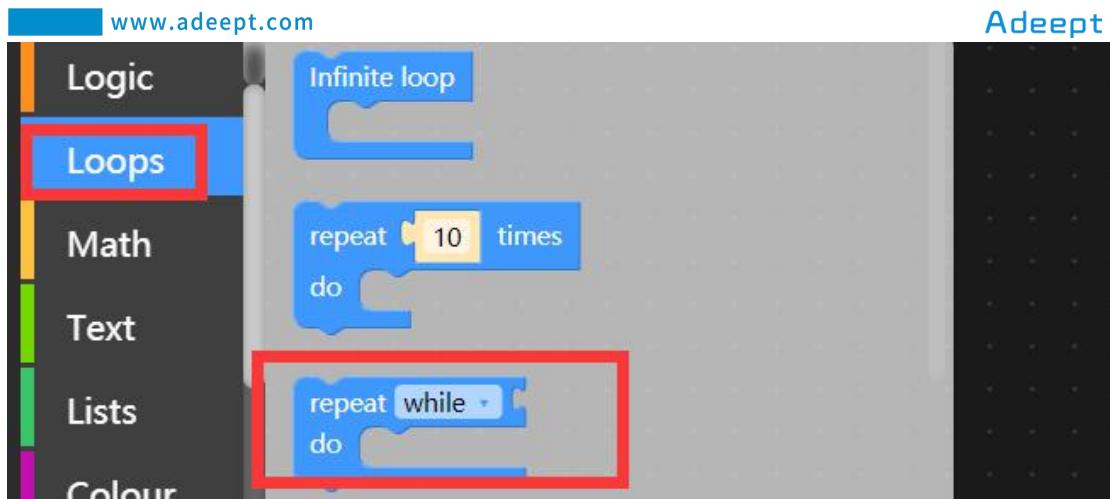
7. Find the instruction **Initialize PIN : PIN 0 , Input status** under the instruction module **► Hardware**, which can set the PIN to Input and Output modes respectively. Drag this instruction to the editing area on the right. It will show as below:



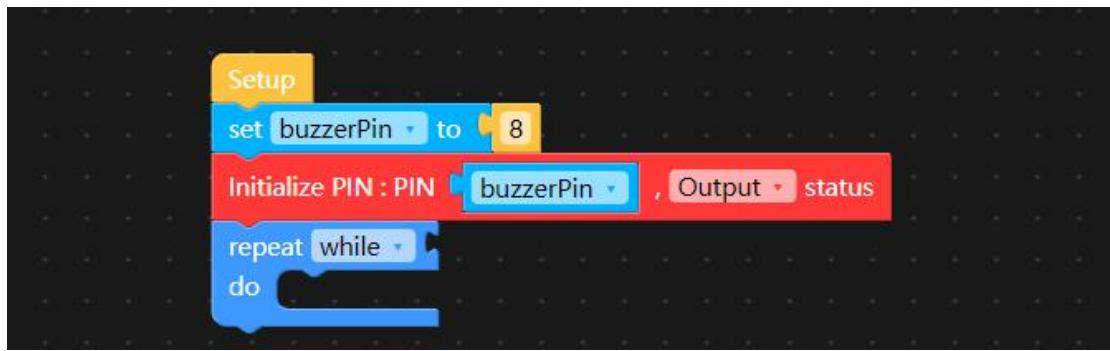
8. In the command module **Variables**, drag the variable **buzzerPin** to the right editing area **PIN 0**. You need to initialize **buzzerPin** to **Output** mode. Click **Input** to modify it to **Output** mode. It will show as below:



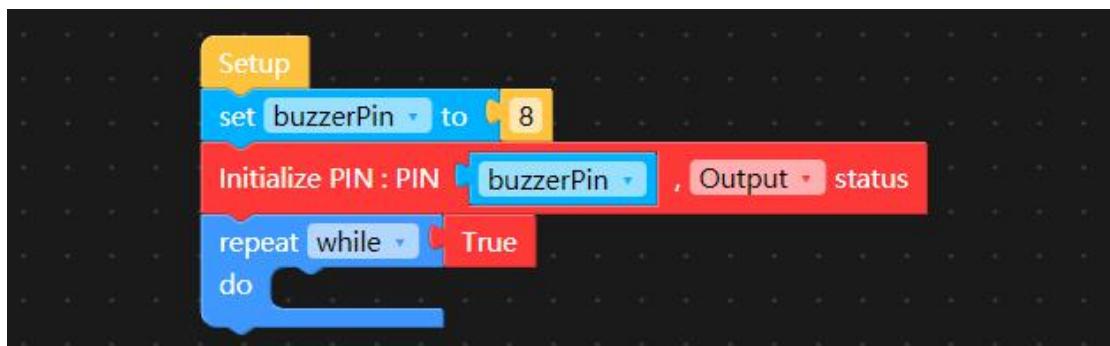
9. Find the loop judgment instruction **repeat [while] do** under the Loops code instruction module. It will show as below::



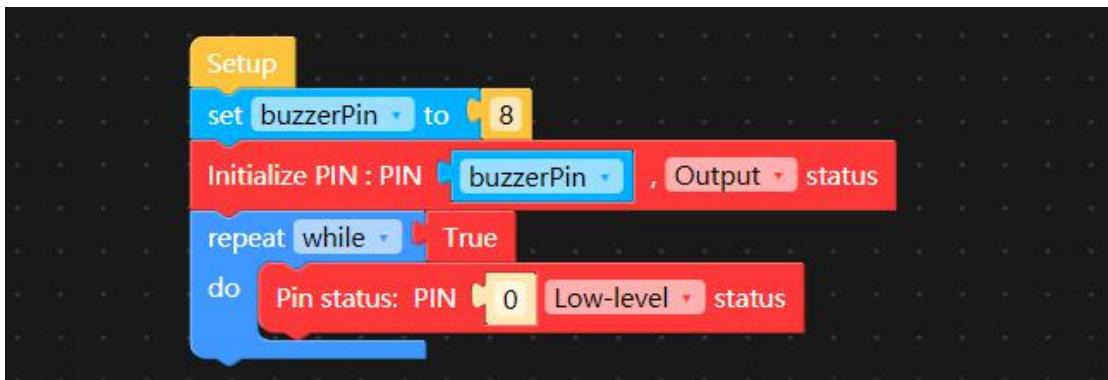
10. Drag  to the position as shown below:



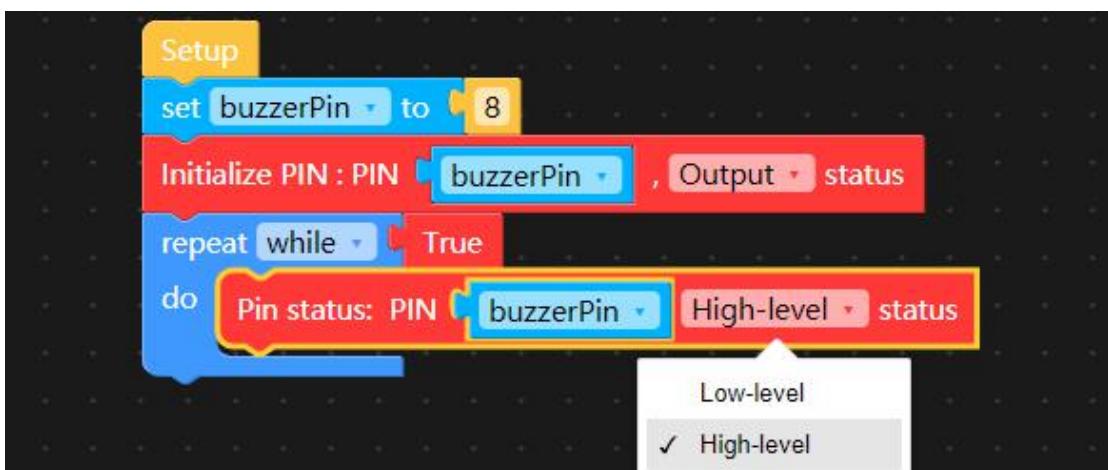
11. We need to determine whether the condition is TRUE. Find  under the code instruction module , and drag it to the following location:



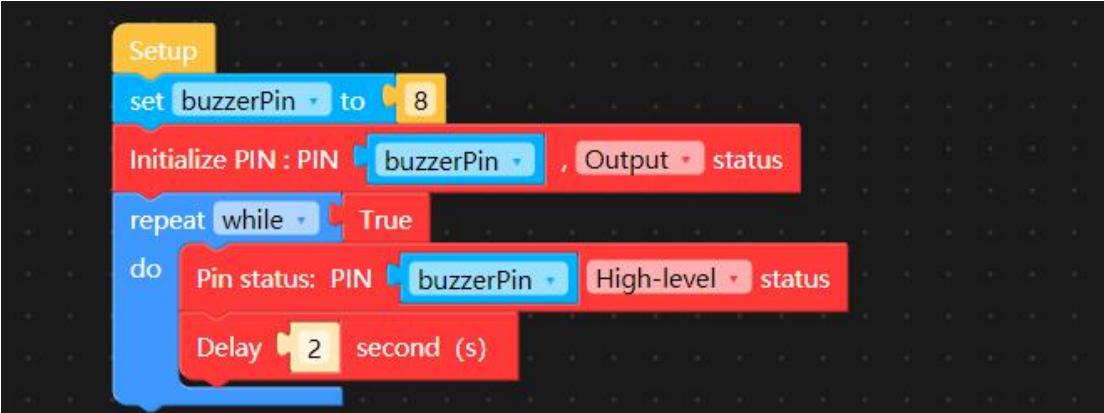
12. Set `buzzerPin` to a high level, then you can make the active buzzer sound. Find the  instruction under the code instruction module . Place it in the position as shown below:



13.In the instruction module **Variables**, drag **buzzerPin** variable into the editing area on the right. Change **Low-level** to **High-level**.It will show as below:



14.Use a delay command to control the active buzzer to continue to be on for a period of time. Find **Delay [0 second (s)]** under the code command module **UNO_R3**. And then you can enter the number of the time you need in **[0]** to control the delay. We enter 2 to control the active buzzer to continuously sound for 2 seconds. It will show as below:

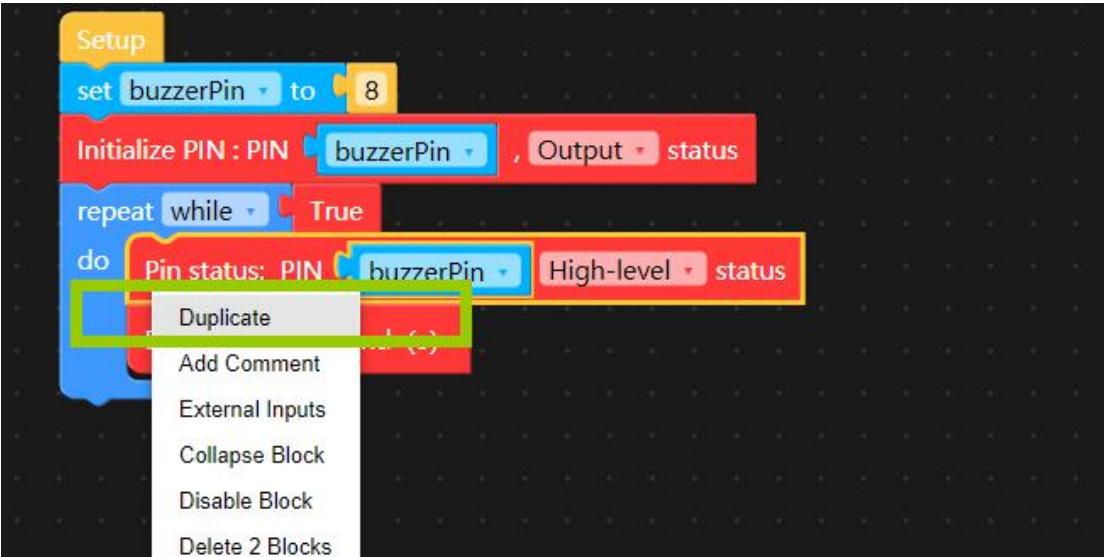


```

Setup
set buzzerPin to 8
Initialize PIN : PIN [buzzerPin v], Output [status v]
repeat (while [True v])
  do [Pin status: PIN [buzzerPin v] High-level [status v]]
    Delay [2 second (s)]
end

```

15. If the pin `buzzerPin` of the active buzzer is set to low level, the active buzzer is in the off state. Move the mouse to the position . Click the right mouse button. Select "Duplicate". It will show as below:

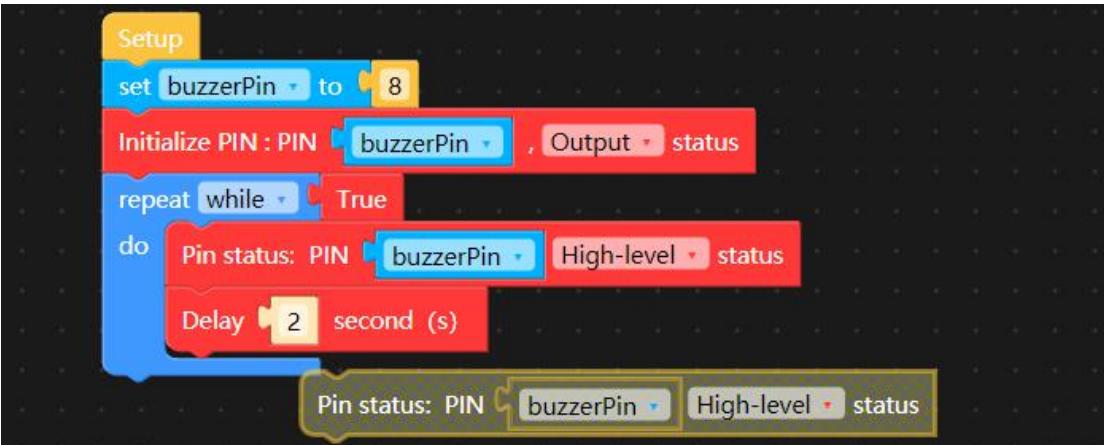


```

Setup
set buzzerPin to 8
Initialize PIN : PIN [buzzerPin v], Output [status v]
repeat (while [True v])
  do [Pin status: PIN [buzzerPin v] High-level [status v]]
    Duplicate
    Add Comment
    External Inputs
    Collapse Block
    Disable Block
    Delete 2 Blocks
  end
  Delay [2 second (s)]
end

```

At this time, the same code instruction block will be copied and generated. It will show as below:

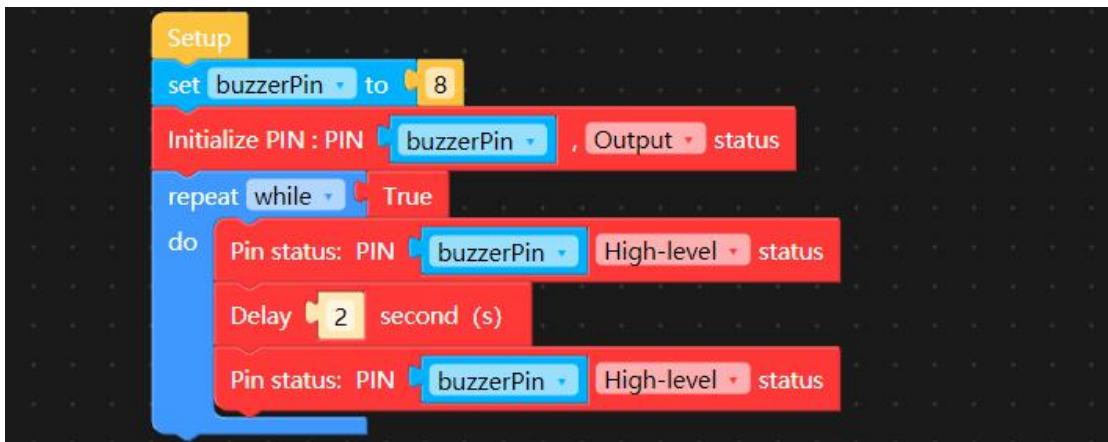


```

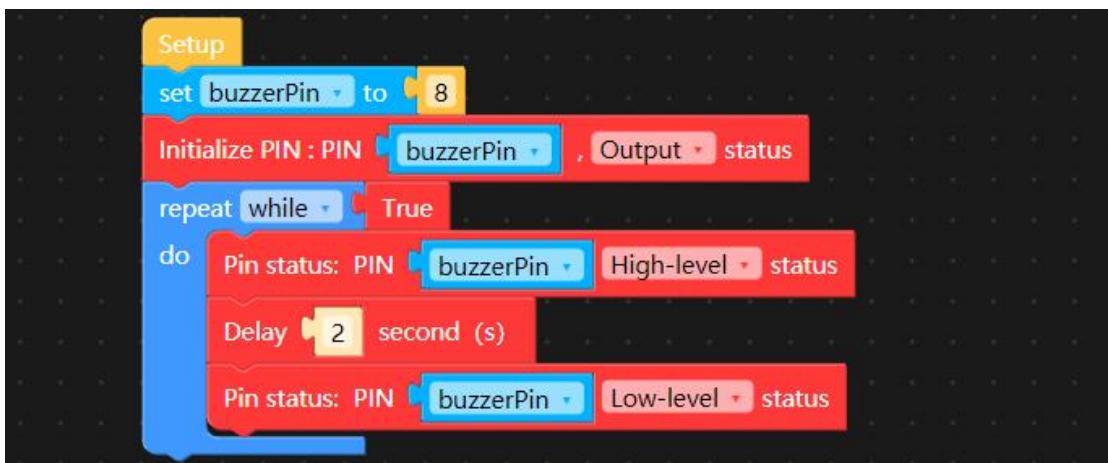
Setup
set buzzerPin to 8
Initialize PIN : PIN [buzzerPin v], Output [status v]
repeat (while [True v])
  do [Pin status: PIN [buzzerPin v] High-level [status v]]
    Pin status: PIN [buzzerPin v] High-level [status v]
  end
  Delay [2 second (s)]
end

```

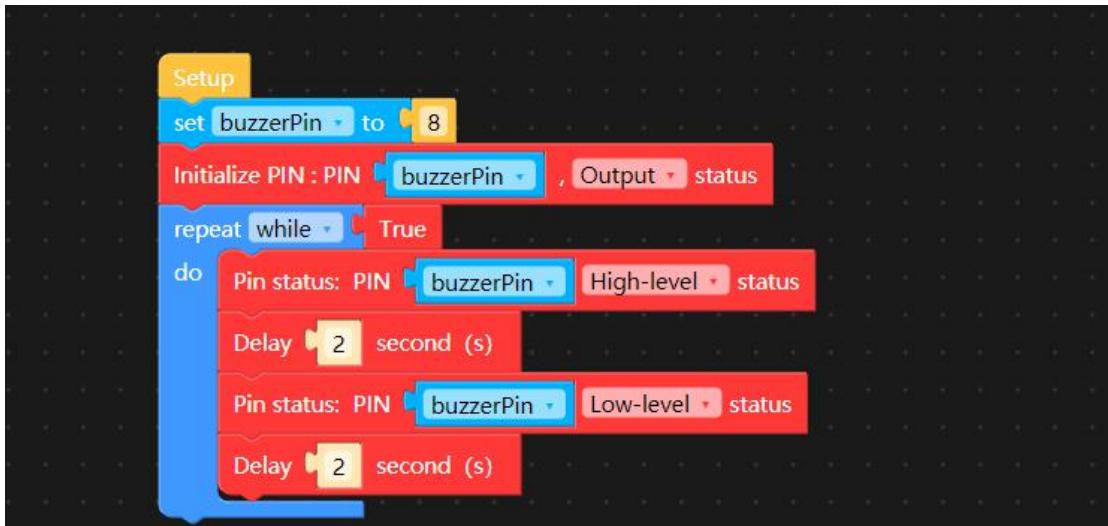
16. Place the copied code instruction block as shown below:



17. If you want to turn off the active buzzer, you only need to change **High-level** to **Low-level**. It will show as below:

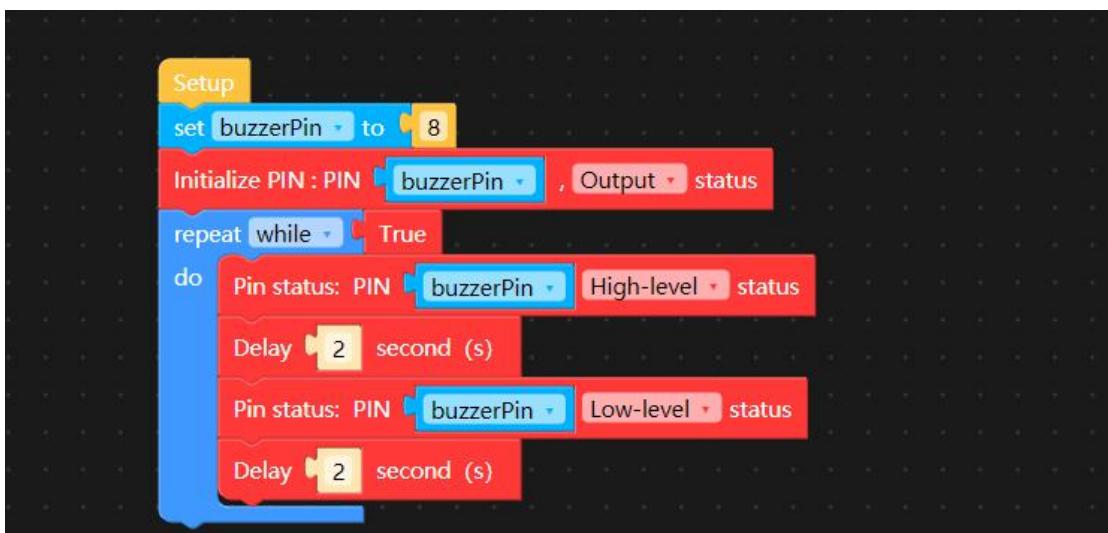


18. Use a delay command to control the active buzzer to continue to be off for a period of time. Find **Delay [0] second (s)** under the code command module **UNO_R3**. And then you can enter the number of the time you need in **[0]** to control the delay. We enter 2 to control the active buzzer to be off for 2 seconds. It will show as below:

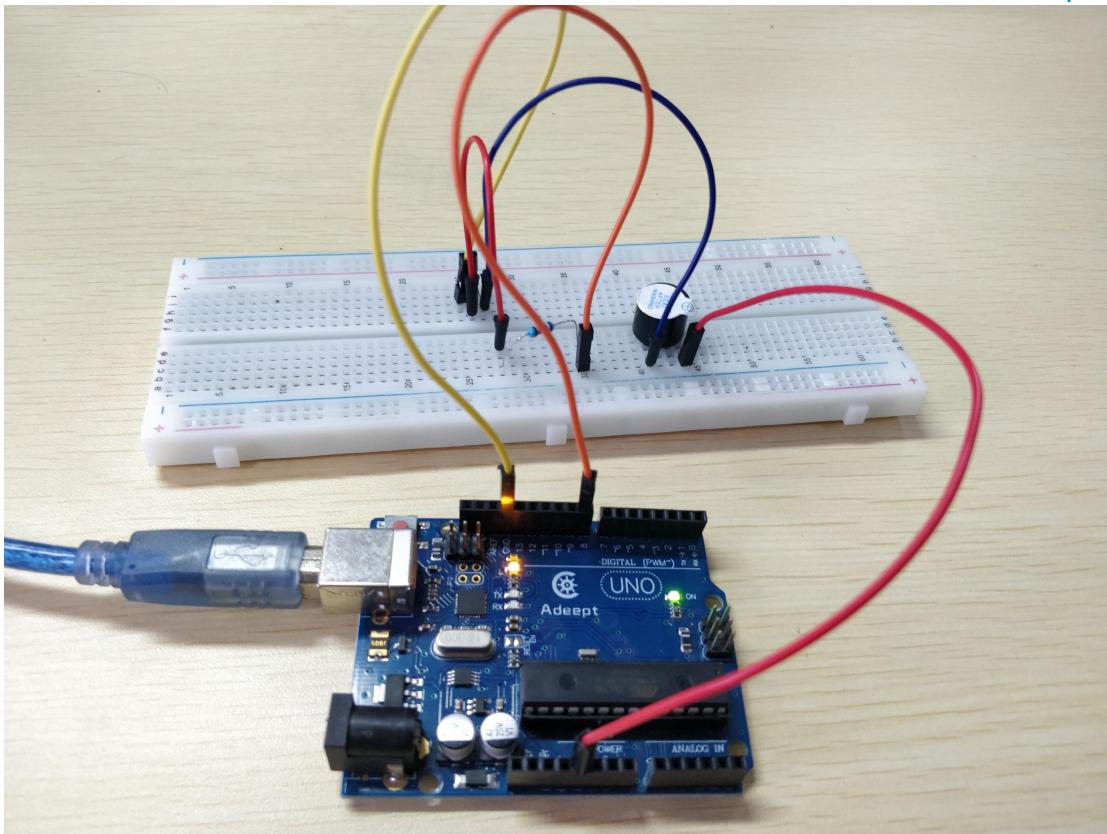


19.The final program is as shown in the figure below. Click the run program button

 in the upper right corner and pay attention to observe the active buzzer. If the active buzzer sounds and is off after a while and repeat this process, it indicates that the experiment is successful. It will show as below:



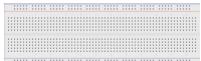
20.The physical connection of the experiment is shown below



Lesson 3 Controlling LED with Button

In this lesson, we will learn how to control LED with Button.

1. Components used in this course

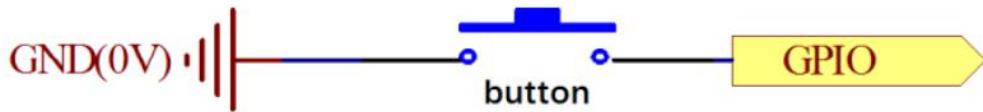
Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	1	
LED	1	
Resistor(10KΩ)	1	
Button(Large)	1	

2. Experimental principle of Button controlling to LED

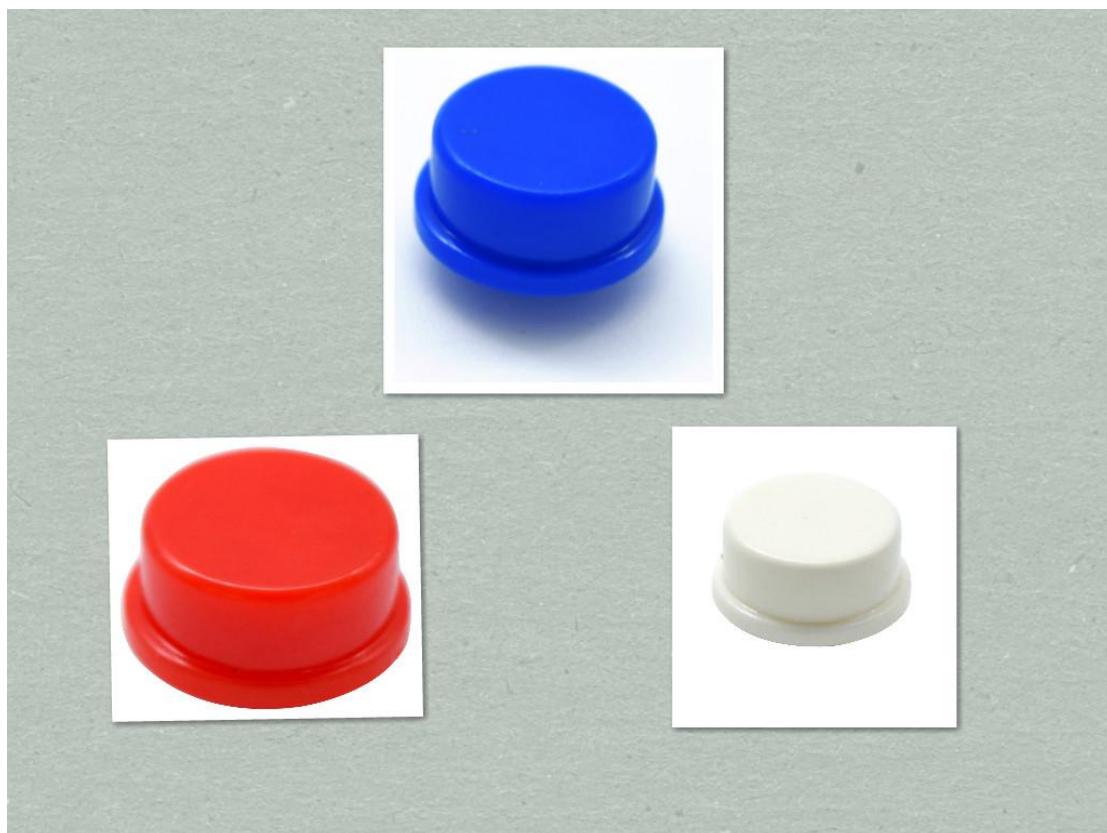
(1) What is a Button

Button is one of the most common input devices. There are two non-touching touch pieces inside a common Button. When the Button is pressed by external force, the two touch pieces are connected together and the circuit is connected. After the external force is released, it returns to the disconnected state, that is, the circuit is disconnected. Many functions can be achieved when used in conjunction with other

components. Its operation is intuitive and effective, and many operations need to be controlled by Button. Almost all electronic devices have the design of Button reserved. Let's learn how to realize simple Button operation on raspberry pie.



The Button used in this lesson is as following figure:



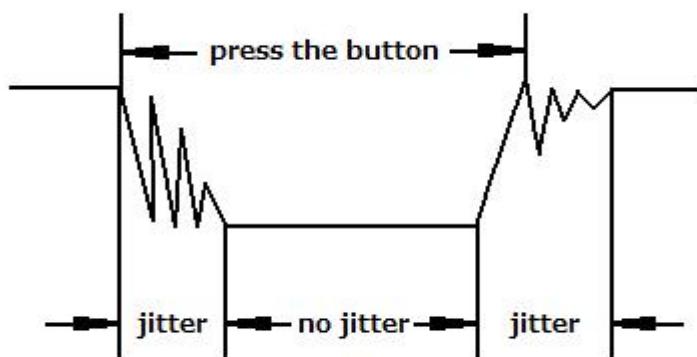
(2) Experimental principles

We control the state of the LED by judging the state of the GPIO port connected to the Button. Since the Arduino IO port can be used as output mode to light up the light, it can also be used as input mode to detect the high and low level of the IO port. Here, when we detect the Button pressed, we give the Arduino IO port a low level, indicating that the Button has been pressed, then we will light the LED. When we detect the release of the Button, we give the Arduino IO port a high level, which

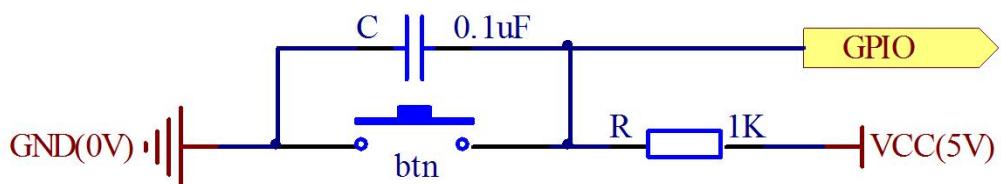
means that the Button has been released, and then we will turn off the LED.

(3) Dealing with Button jitter

The button jitter must happen in the process of using. The jitter waveform is as the flowing:

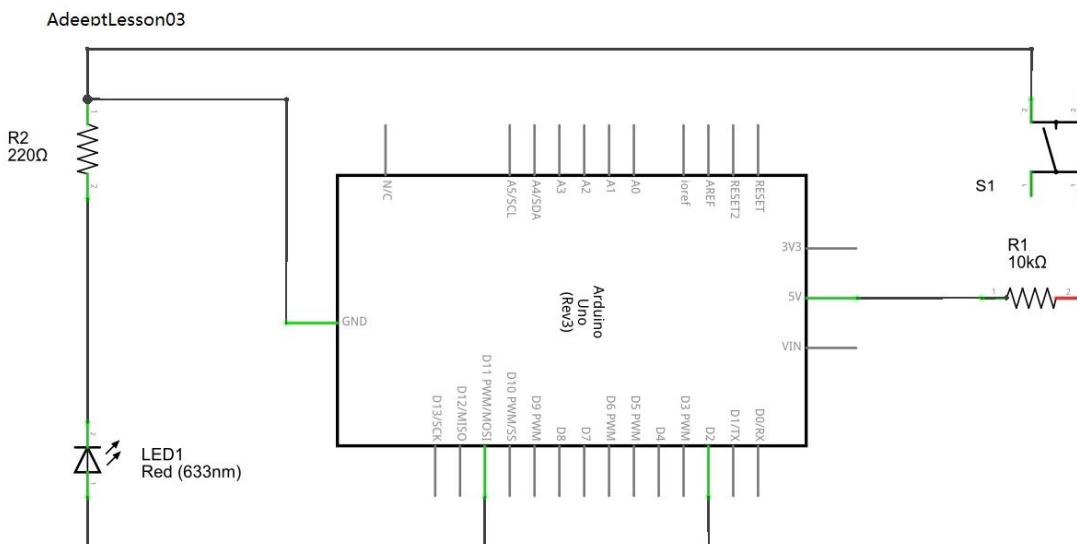
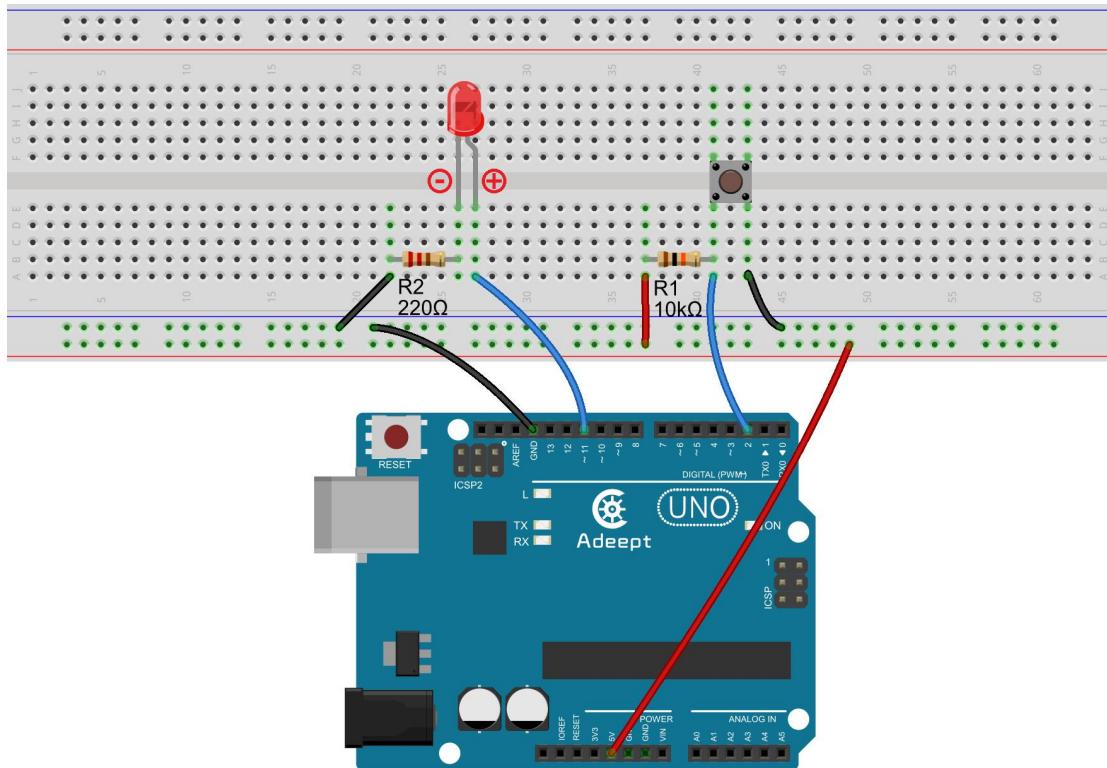


Each time you press the button, the Arduino will think you have pressed the button many times due to the jitter of the button. We have to deal with the jitter of buttons before we use the button. We can remove the jitter of buttons through the software programming, besides, we can use a capacitance to remove the jitter of buttons. Here we introduce the software method. First, we detect whether the level of button interface is low level or high level. When the level we detected is low level, 5~10 MS delay is needed, and then detect whether the level of button interface is low or high. If the signal is low, we can confirm that the button is pressed once. You can also use a 0.1uF capacitance to clean up the jitter of buttons. The schematic diagram is shown in below.



3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes, as shown in the following figure:



4. Controlling the LED through Button

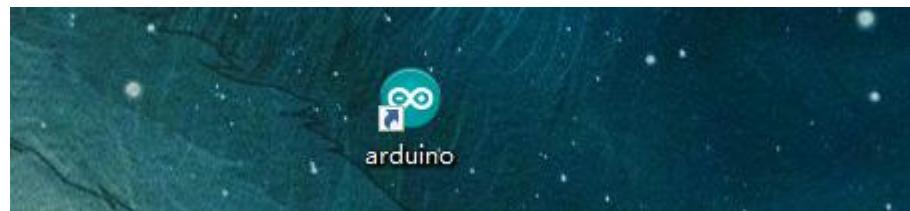
We provide two different methods to light up the LED. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C

language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

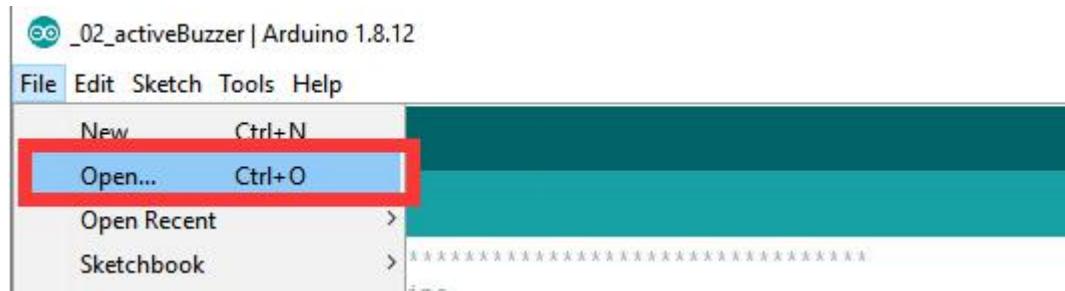
1. Using C language to program Button to control LED on Arduino UNO

(1) Compile and run the code program of this course

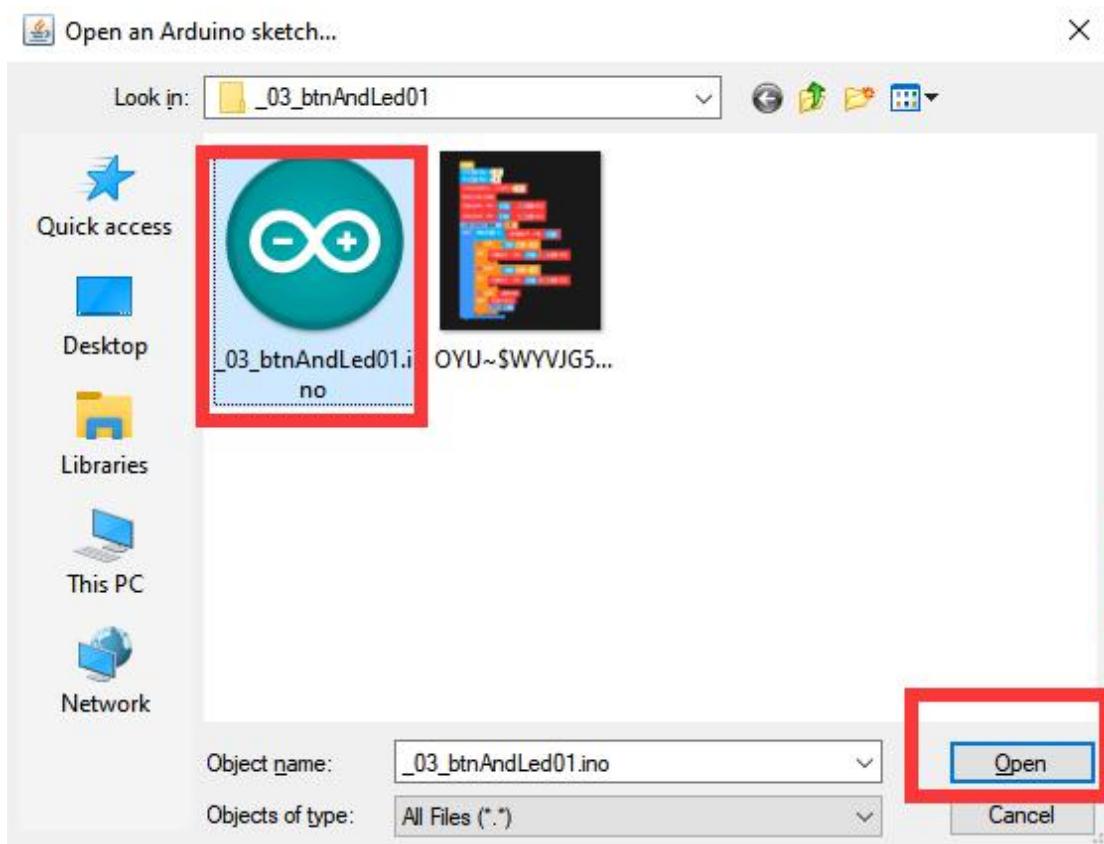
1. Open the Arduino IDE software, as shown below:



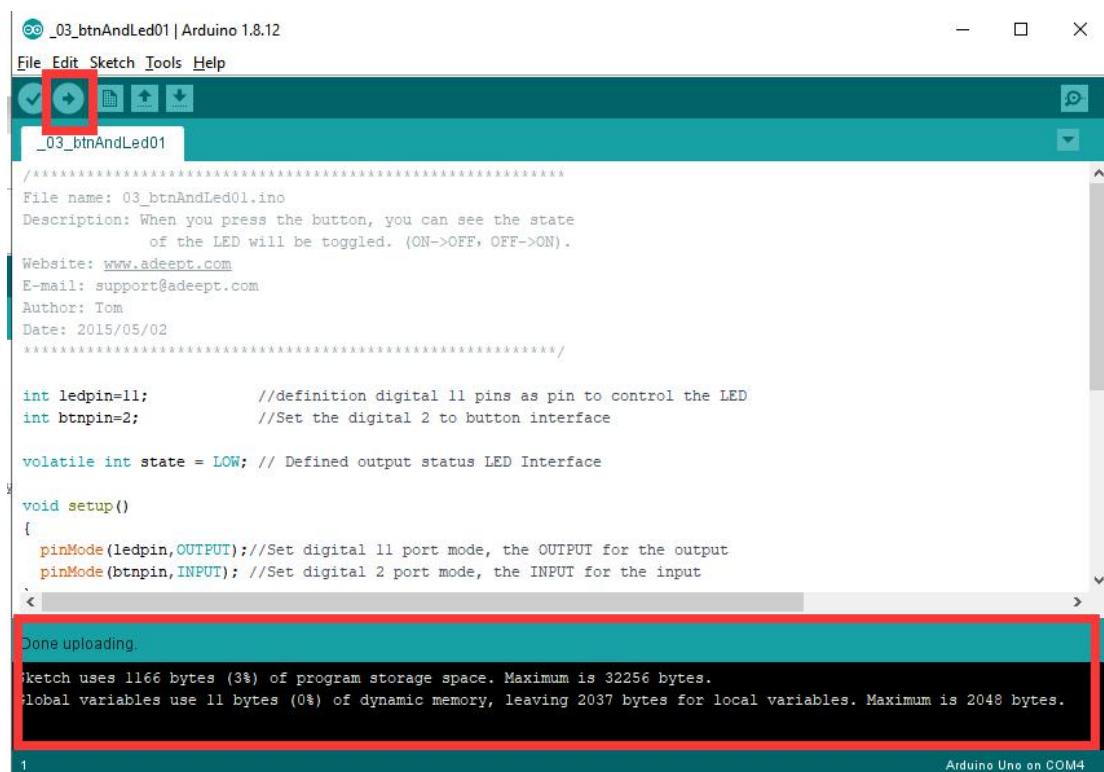
2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\03_btnAndLed01 directory. Select _03_btnAndLed01.ino. This file is the code program we need in this course. Then click Open.



4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.



The screenshot shows the Arduino IDE interface with the sketch '_03_btnAndLed01' loaded. The top menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar has several icons, with the upload icon (a blue square with a white triangle) highlighted with a red box. The code editor displays the following code:

```

//=====
File name: 03_btnAndLed01.ino
Description: When you press the button, you can see the state
            of the LED will be toggled. (ON->OFF, OFF->ON).
Website: www.adeept.com
E-mail: support@adeept.com
Author: Tom
Date: 2015/05/02
//=====

int ledpin=11;          //definition digital 11 pins as pin to control the LED
int btnpin=2;           //Set the digital 2 to button interface

volatile int state = LOW; // Defined output status LED Interface

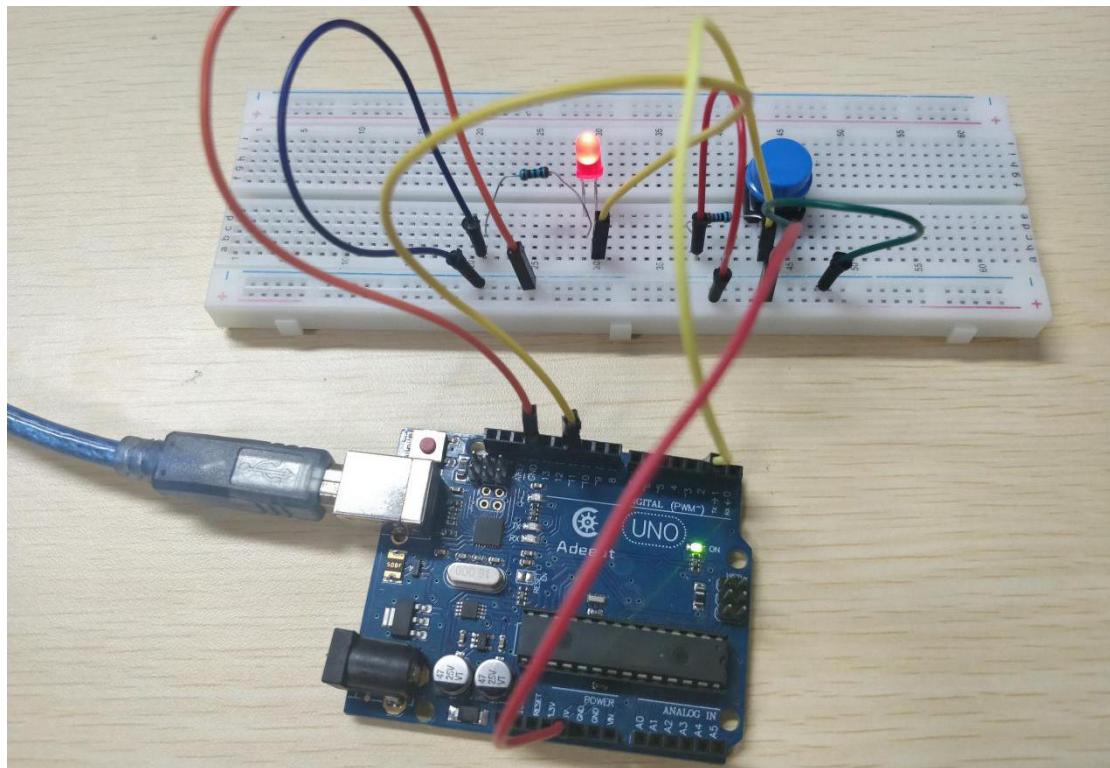
void setup()
{
  pinMode(ledpin,OUTPUT); //Set digital 11 port mode, the OUTPUT for the output
  pinMode(btnpin,INPUT); //Set digital 2 port mode, the INPUT for the input
}

void loop()
{
  if(state==LOW)
    {
      digitalWrite(ledpin,HIGH);
      state=HIGH;
    }
  else
    {
      digitalWrite(ledpin,LOW);
      state=LOW;
    }
}

```

The status bar at the bottom right shows 'Arduino Uno on COM4'. In the bottom right corner of the code editor, there is a red box highlighting the text 'Done uploading.' followed by the upload progress message: 'Sketch uses 1166 bytes (3%) of program storage space. Maximum is 32256 bytes. Global variables use 11 bytes (0%) of dynamic memory, leaving 2037 bytes for local variables. Maximum is 2048 bytes.'

5.If we press the Button, the LED will light up; if we press the Button again, the LED will turn off, indicating that the experimental test was successful. The physical connection diagram of the experiment is as below:



(2)The core code program

After the above hands-on operation, you must be very interested to know how we use C language to program Button to control LED on Arduino UNO.We will introduce how our core code can be achieved:

1.Define the pin that controls the LED as pin 11 through int ledPin=11. Define the pin that controls the Button as pin 2 through int btnpin=2. In the setup() method, set ledPin to output mode via pinMode(ledPin, OUTPUT) ledPin; set btnpin as the input mode through pinMode(btnpin, INPUT).

```

int ledpin=11;           //definition digital 11 pins as pin to c
int btnpin=2;            //Set the digital 2 to button interface

volatile int state = LOW; // Defined output status LED Interface

void setup()
{
    pinMode(ledpin,OUTPUT);//Set digital 11 port mode, the OUTPUT fo
    pinMode(btnpin,INPUT); //Set digital 2 port mode, the INPUT for
}

```

2.In the loop() method, we check whether the Button is pressed with if. when digitalRead(btnpin)==LOW, it means that the Button is pressed. The pin state of the ledpin is set to 1 (high by digitalWrite(ledpin,1) Level),and the LED will be lit up. When digitalRead(btnpin)!=LOW, it indicates that the Button is released. When the digitalWrite(ledpin,0) sets the pin status of the ledpin to 0 (low level), the LED will be off.

```
void loop()
{
    if(digitalRead(btnpin)==LOW)           //Detection button interface to low
    {
        digitalWrite(ledpin,1);          //Output control status LED, ON
    }
    if(digitalRead(btnpin)!=LOW)         //Detection button interface to high
    {
        digitalWrite(ledpin,0);          //Output control status LED, OFF
    }
}
```

2. Programming Button to control LED on Arduino UNO with graphical code blocks

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1.Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

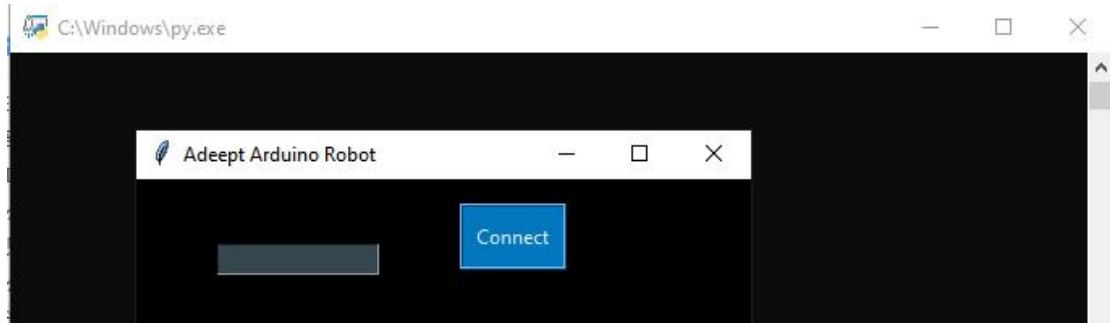
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

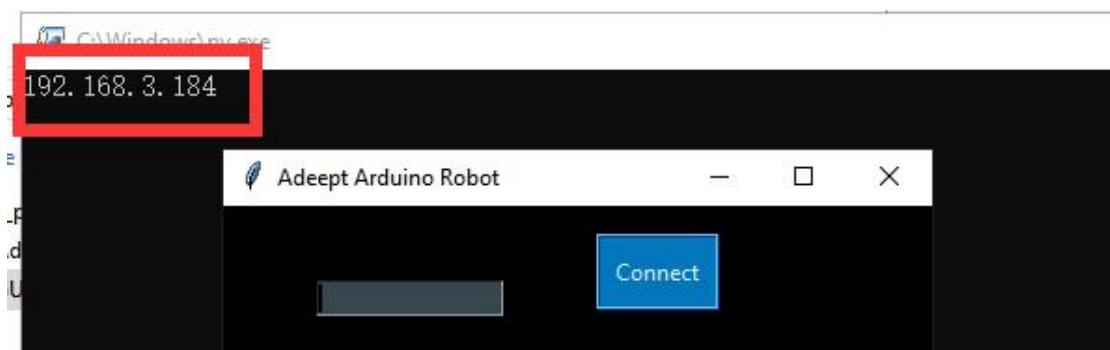
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

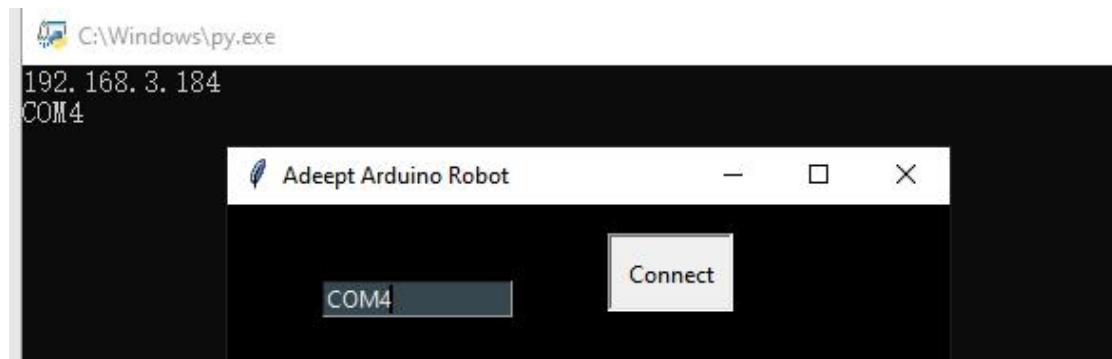
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



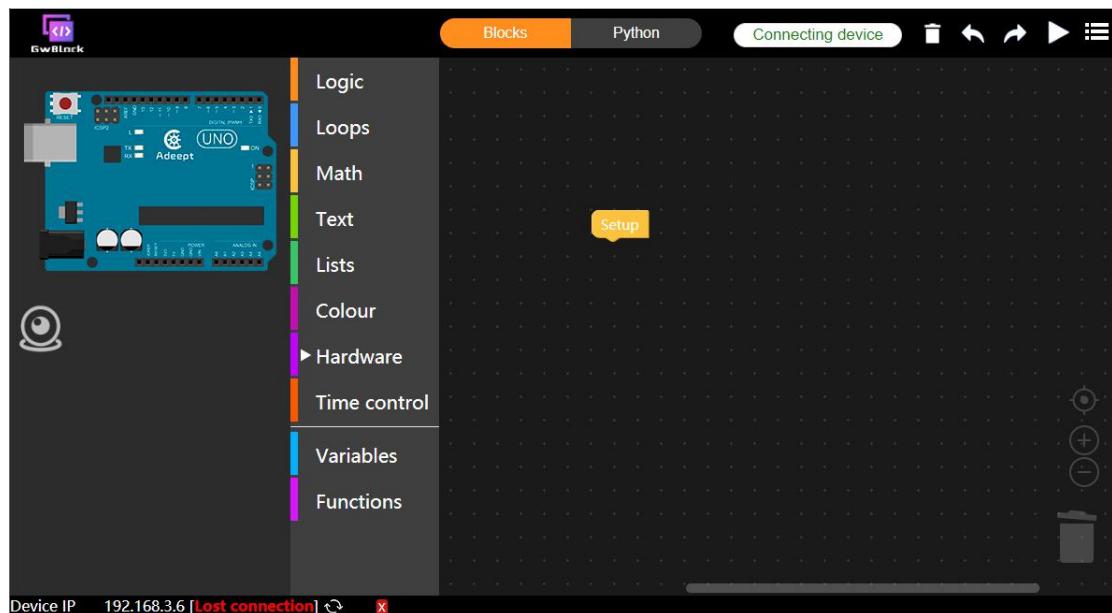
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



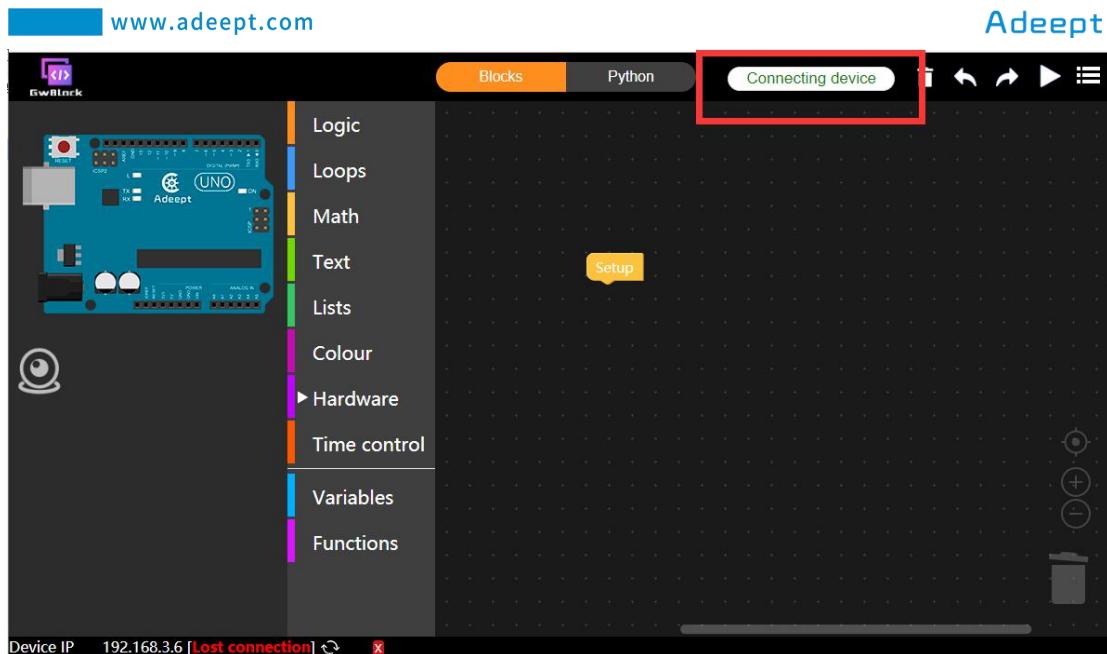
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

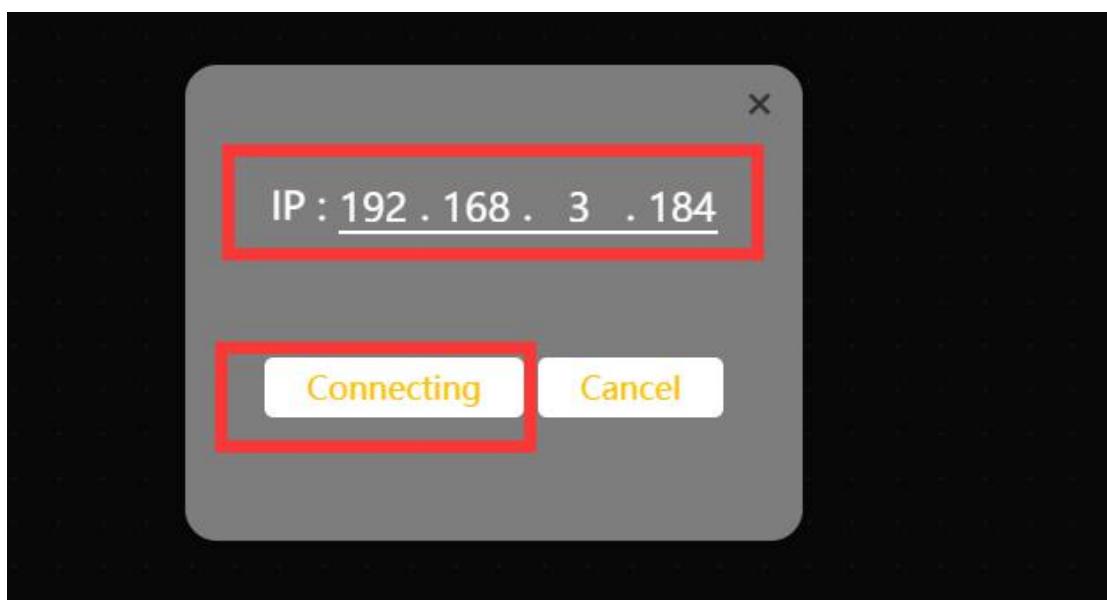
After successfully entering the website, the interface is as follows:



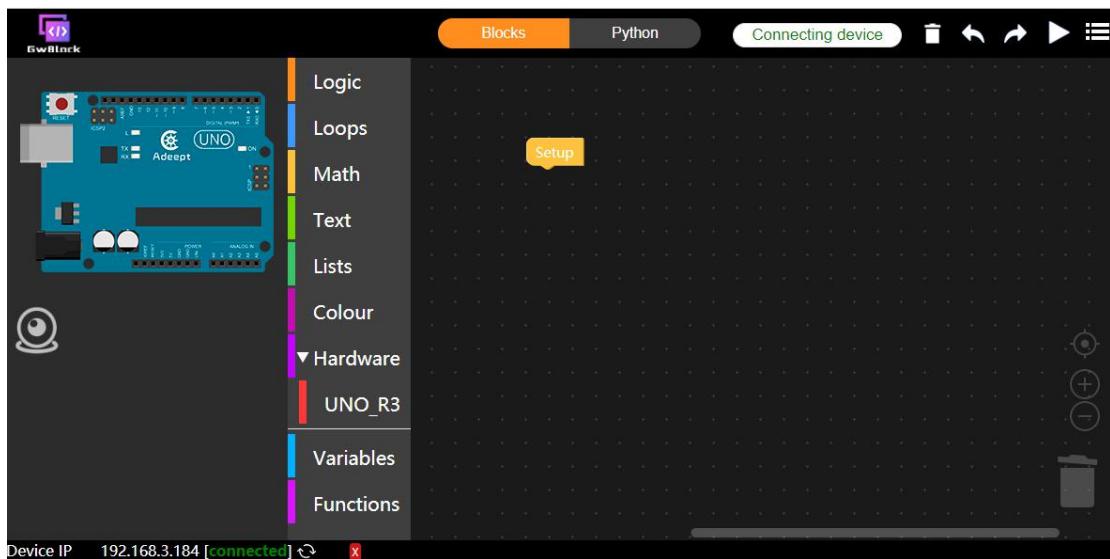
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



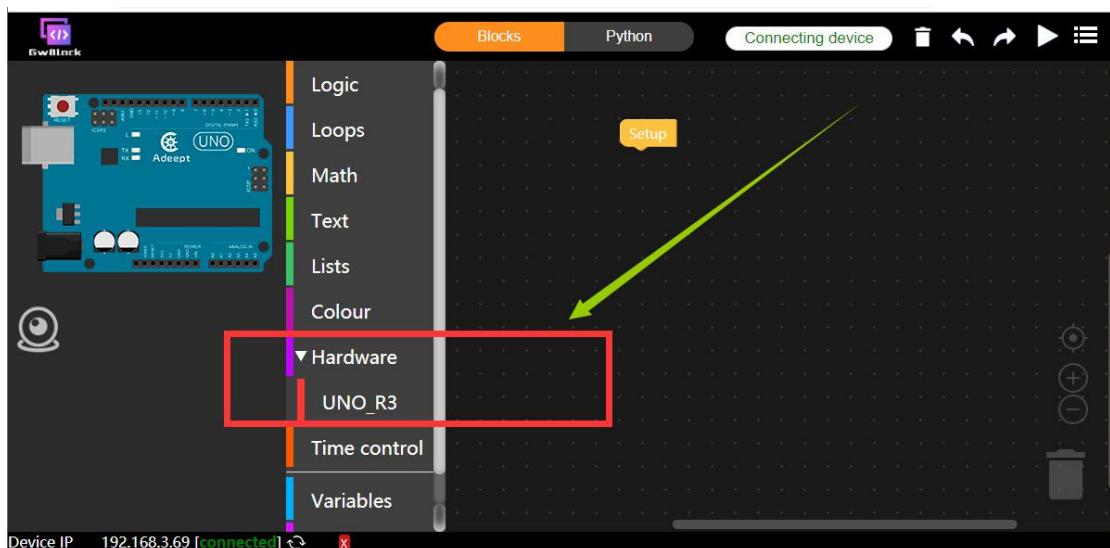
8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:



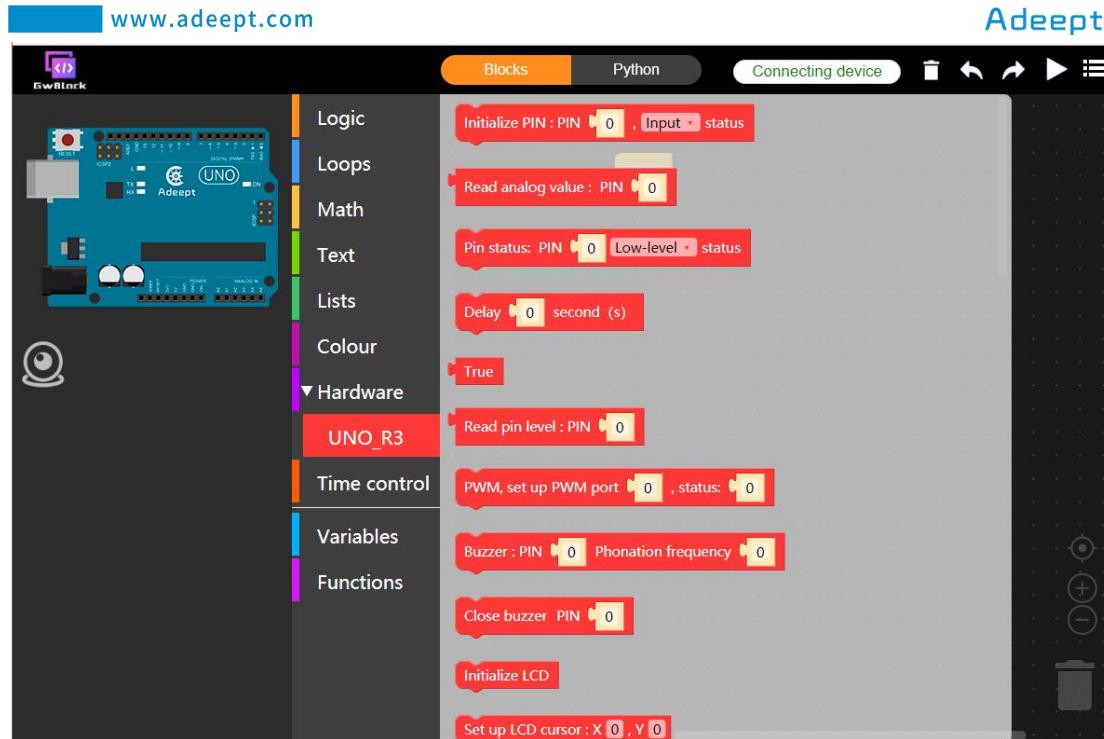
(2)Controlling LED with Button

Now let us learn how to use the GwBlock graphical editor to program to control LED with Button on the Arduino UNO.

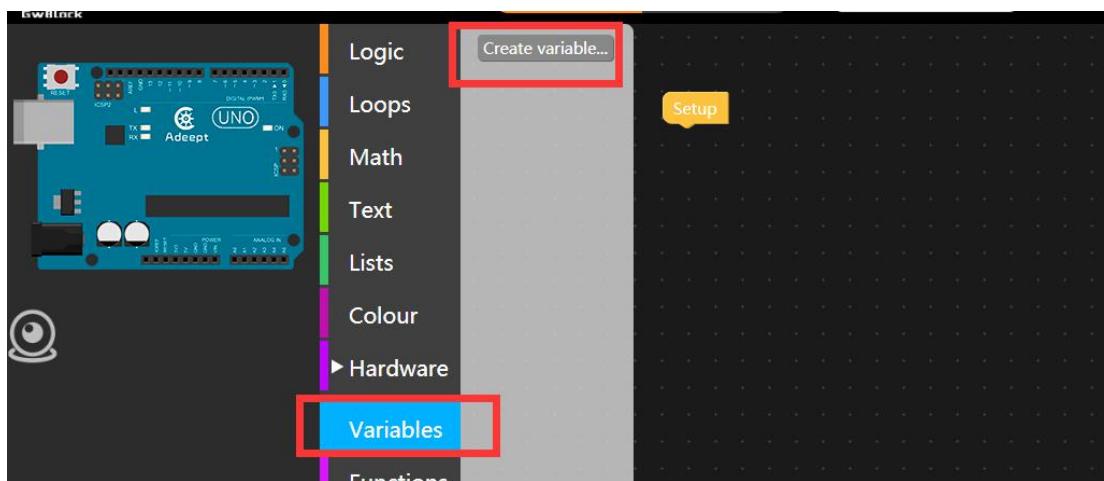
1.After successfully connecting to the GwBlock IDE, we find "Hardware" in the toolbar of the code instruction module in the middle. The UNO_R3 code instruction module is a collection of graphical code blocks that control the Arduino UNO.



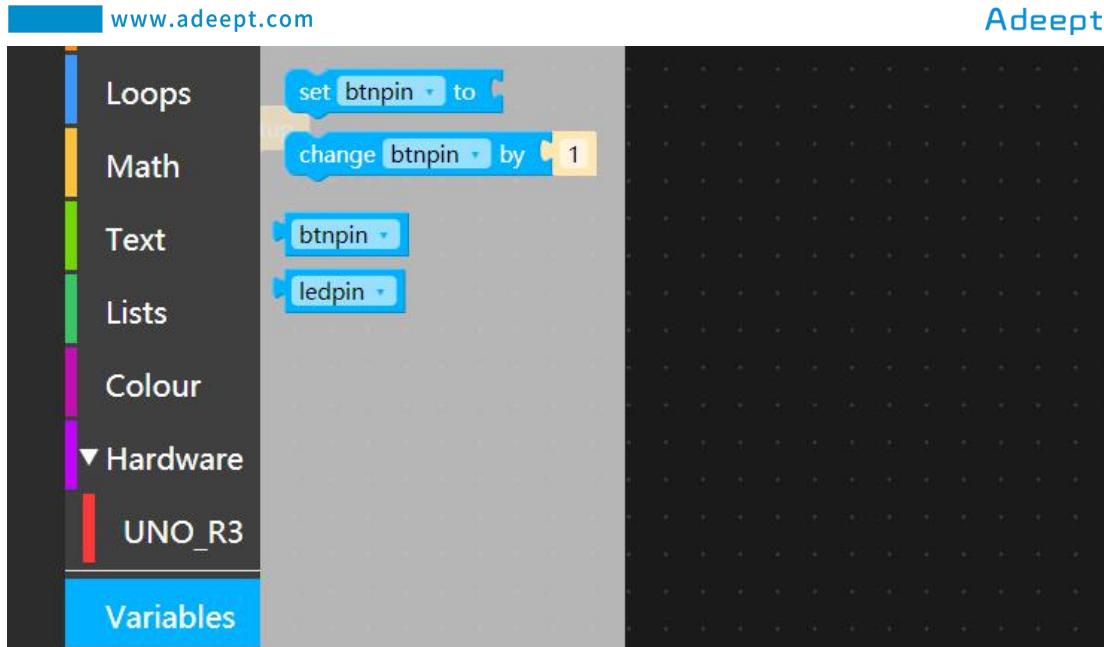
2.The code instruction module **UNO_R3** after opening is as shown in the figure below, and each graphical code block has different functions.



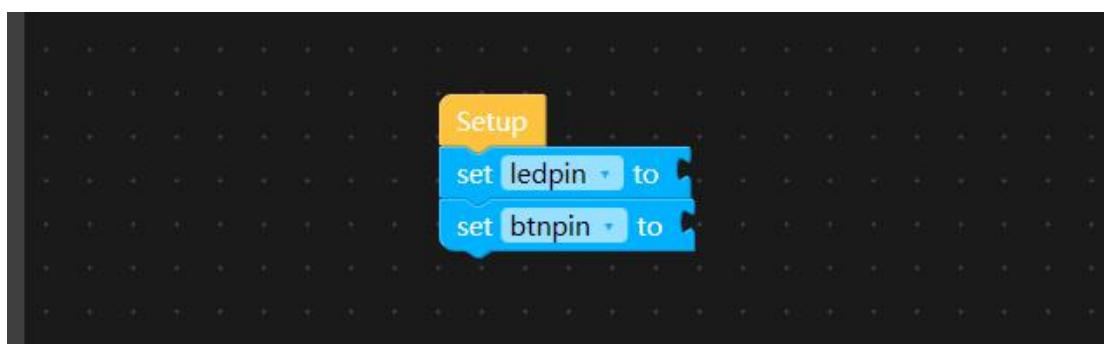
3.Click Create variable under Variables in the toolbar of the code instruction block to create variables.



4.Create the variable ledpin and bttnpin. It will show as below:

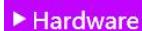


5.The variable ledpin represents the pin number of the LED.The variable btnpin represents the pin number of the Button. Drag them to the position as shown below. It will show as below:



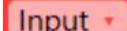
6.By knowing that the pin of the LED is connected to port 11 and the pin of the Button is connected to port 2 in "3. Wiring Diagram", we need to change ledpin to 11 and the btnpin to 2. Find the instruction **0** in the instruction bar **Math**. The number "0" can be modified to 11 and 2 respectively. It will show as below:

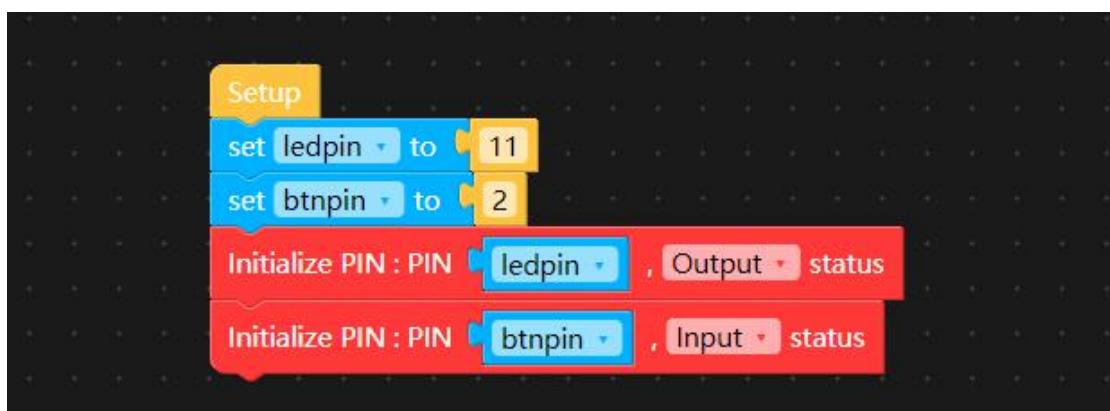


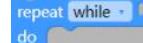
7. Find the instruction  under the instruction module  , which can set the PIN to Input and Output modes respectively.

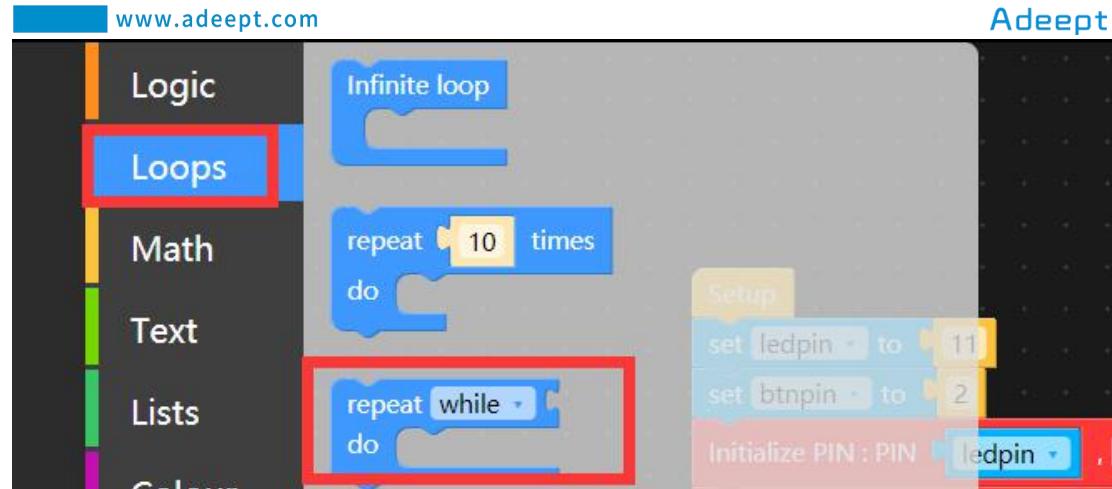
Drag this instruction to the editing area on the right. It will show as below:



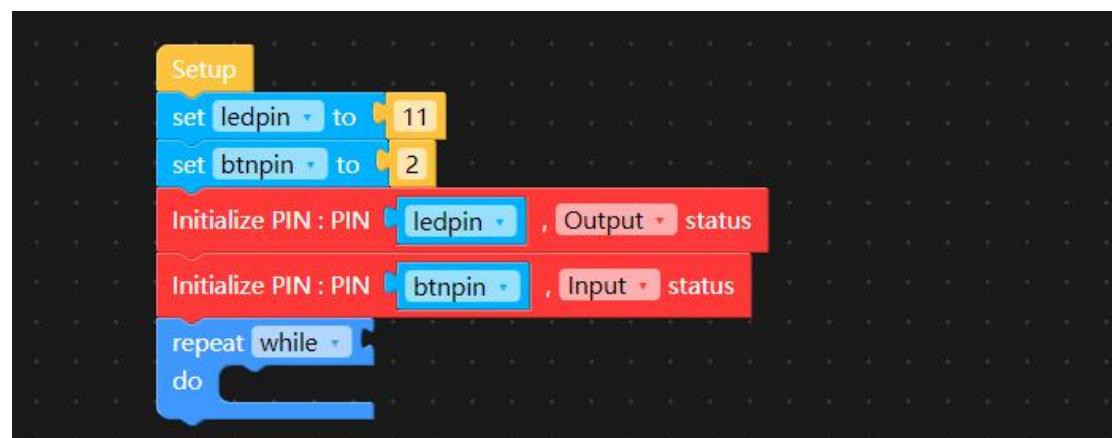
8. In the command module  , drag the variable  and  to the right editing area  . You need to initialize ledpin to Output mode and btncpin to input mode. Click  to modify it to Output mode. It will show as below:



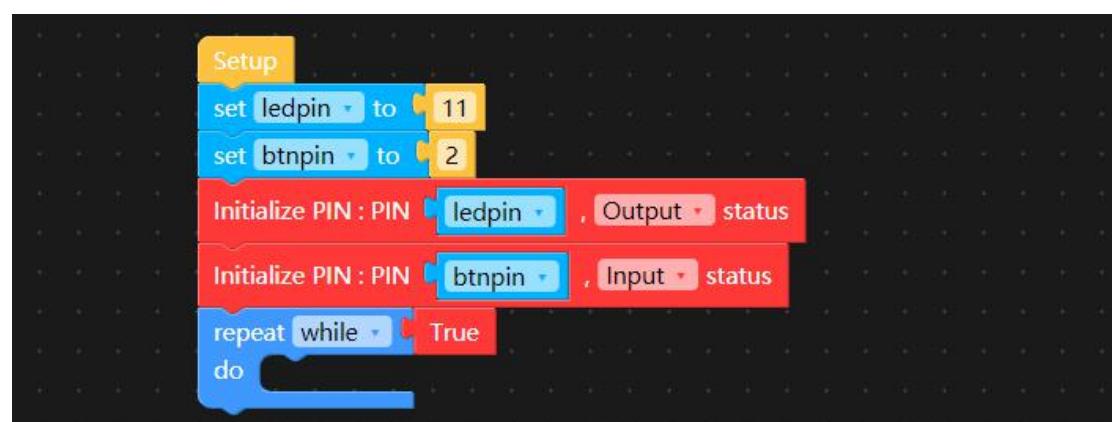
9. Find the loop judgment instruction  under the Loops code instruction module. It will show as below::



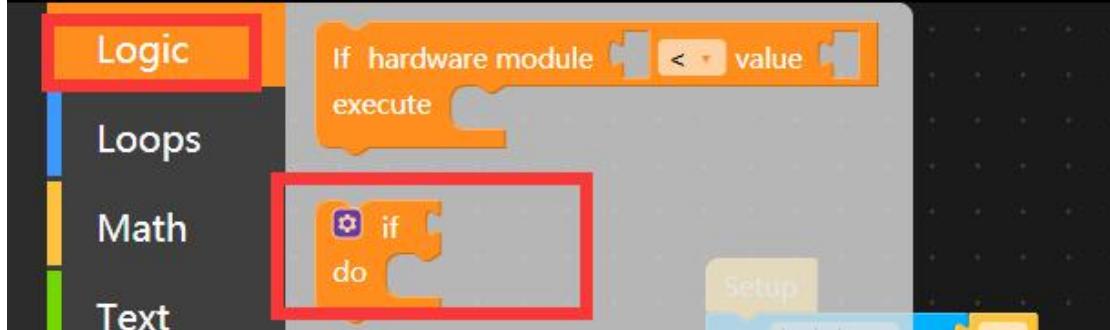
10.Drag  to the position as shown below:

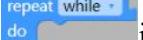


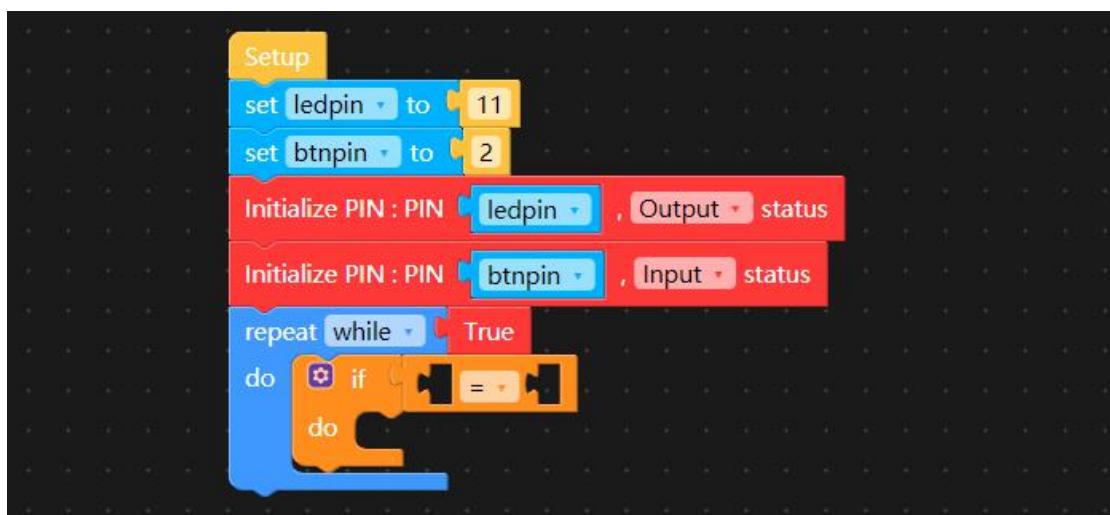
11.We need to determine whether the condition is TRUE. Find  under the code instruction module , and drag it to the following location:



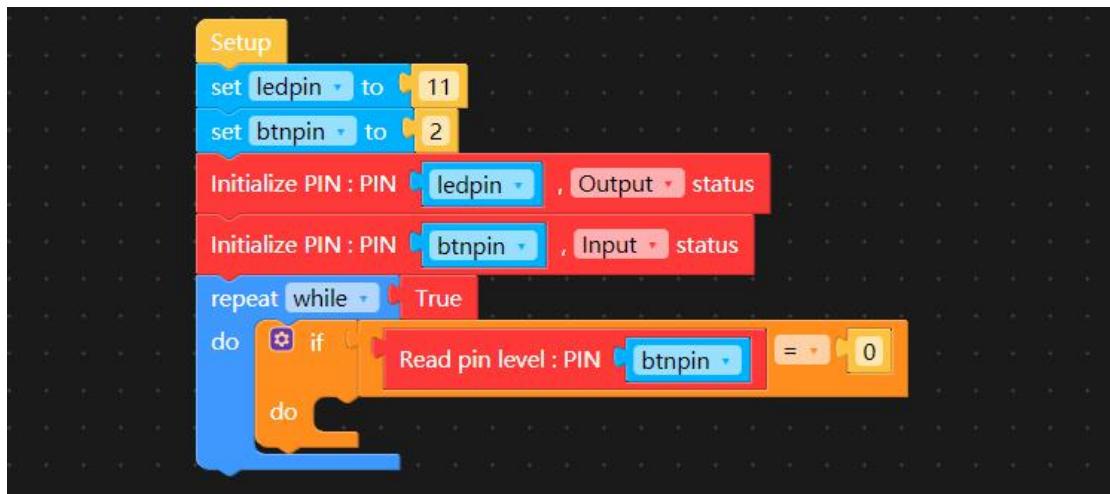
12.Use the if command block to determine whether the Button is pressed. Determine whether its high or low state is 0 or 1.Then set the LED to be on or off. Find the instruction  in the instruction module . It will show as below:



13. Drag the instruction  to the instruction  in the editing area on the right. Find  in the instruction module **Logic**. It will show as below:

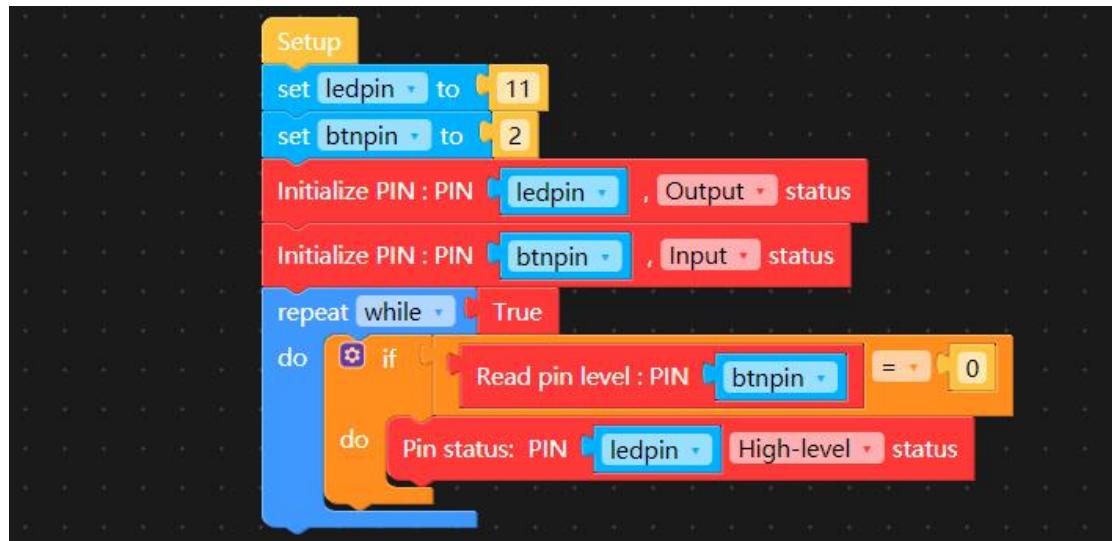


14. Determine whether the status of btncpin is low (0). Find the instruction  under the code instruction module **UNO_R3**. Find the instruction  under the instruction module **Math**. It will show as below after the modification:



15. When btncpin=0, it means that the Button is pressed and the LED is on. By setting

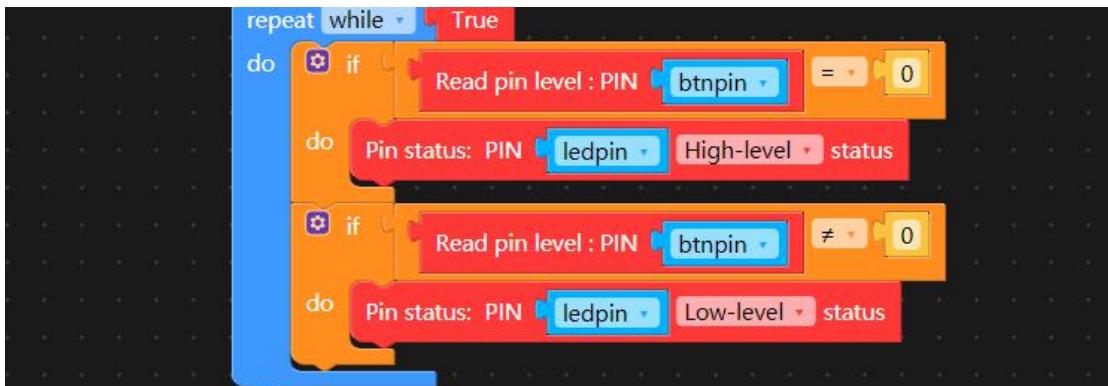
the state of the ledpin pin to high level through the command **Pin status: PIN ledpin High-level status**, the LED will be lit (turned on). It will show as below:



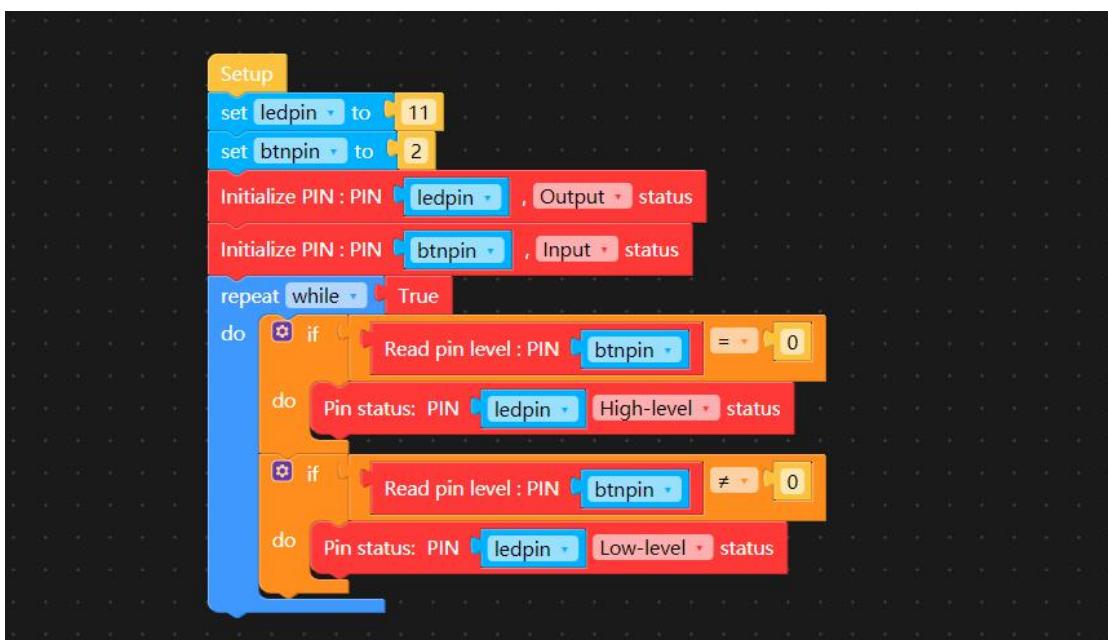
16. On the contrary, when btpnpin is not equal to 0, it means that the Button is released. At this time, the LED is off (turned off). We can know that the btpnpin is not equal to 0 through the instruction **Read pin level : PIN btpnpin ≠ 0**. It will show as below:



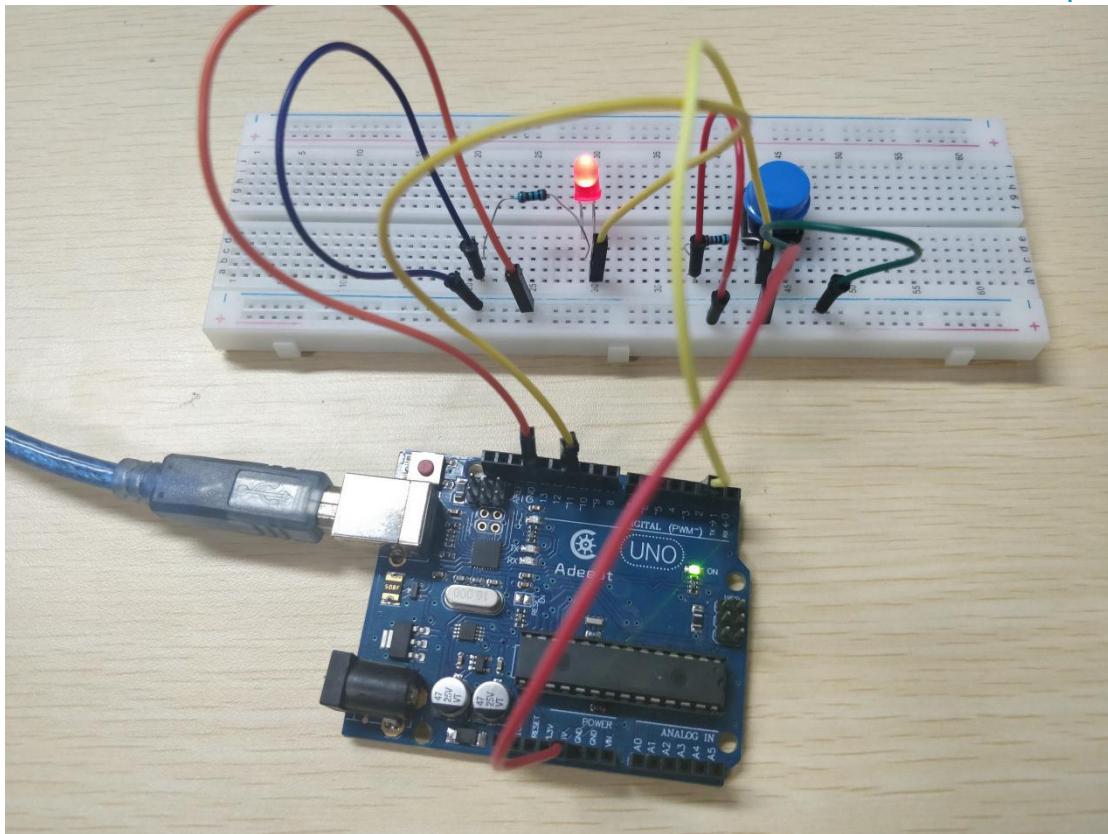
17. If the state of the ledpin pin is set to the low state through the command **Pin status: PIN ledpin Low-level status**, the LED will be off. It will show as below:



18.Then our program is done.It will show as below:



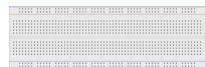
19.Click the run program button  in the upper right corner to Upload the program on the Arduino UNO. If the LED lights up when the Button is pressed, and the LED goes out when the Button is released, indicating that the experimental test was successful. It will show as below:



Lesson 4 Controlling LED with Relay Module

In this lesson, we will study how to control LED with Relay Module.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	1	
LED	1	
Relay Module	1	

2. The introduction of the Relay Module

(1) The Relay Module

Relay module is an electrical control, which is an electrical appliance that causes a predetermined step change in the controlled quantity in the electrical output circuit when the input quantity (excitation quantity) changes to the specified requirements. It has the interaction between control system (also known as the input loop) and the controlled system (also known as the output loop). Usually used in automated control circuits, it is actually a kind of “Automatic switch” with small current to control the operation of large current. Therefore, it plays a role of automatic adjustment, safety

protection and conversion circuit in the circuit.

As a control element, in all, the relay has the following functions:

- 1) Expand the control range: for example, when the control signal of the multi-contact relay module reaches a certain value, it can be changed, broken and connected to the multi-circuit according to the different forms of the contact group at the same time.
 - 2) Amplification: for example, sensitive relay, Intermediate relay, etc., with a very small amount of control, can control a large power circuit.
 - 3) Integrated signals: for example, when multiple control signals are input into multiple Coilrelays according to the prescribed form, the predetermined control effect can be achieved through comparison and integration.
 - 4) Automatic, remote control and monitoring: for example, the relay module on the automatic device can be combined with other electrical appliances to form a program control line, so as to realize automatic operation.



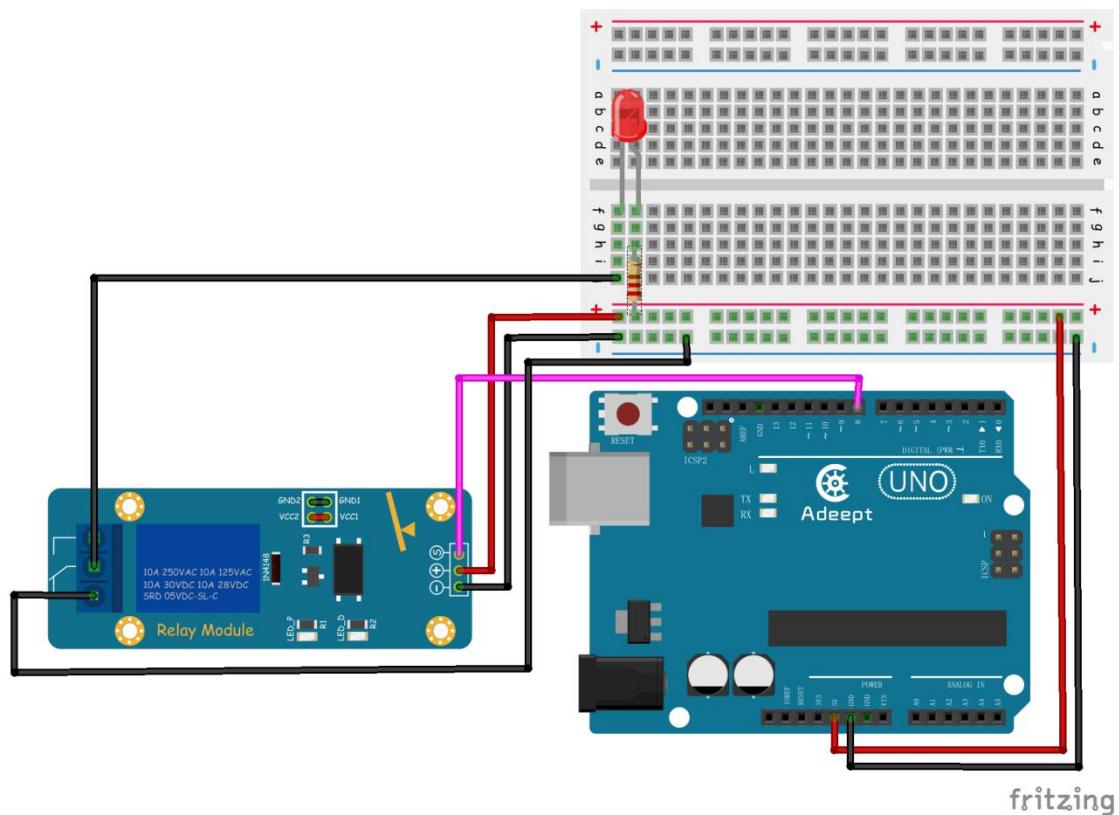
(2) Working principle of the Relay Module

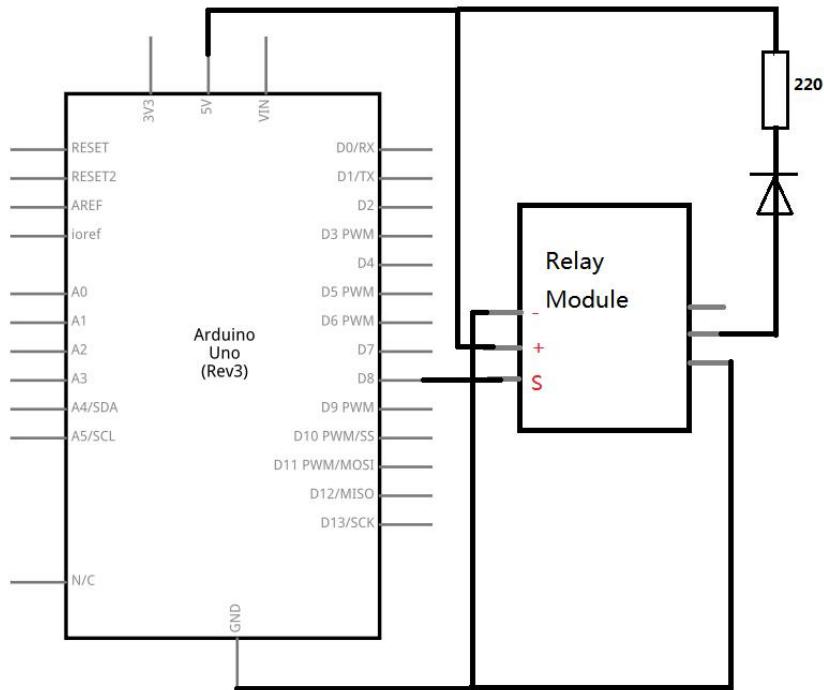
The role of the relay module is to use one circuit (usually a small current) to

control the on and off of another circuit (usually a large current), and in this control process, the two circuits are generally isolated. Its basic principle is to use the electromagnetic effect is used to control the mechanical contacts to achieve the purpose of making and breaking, and the coil with the iron core is energized-the coil current generates a magnetic field-the magnetic field absorbs the armature action to make and break the contact.

3.Wiring diagram (Circuit diagram)

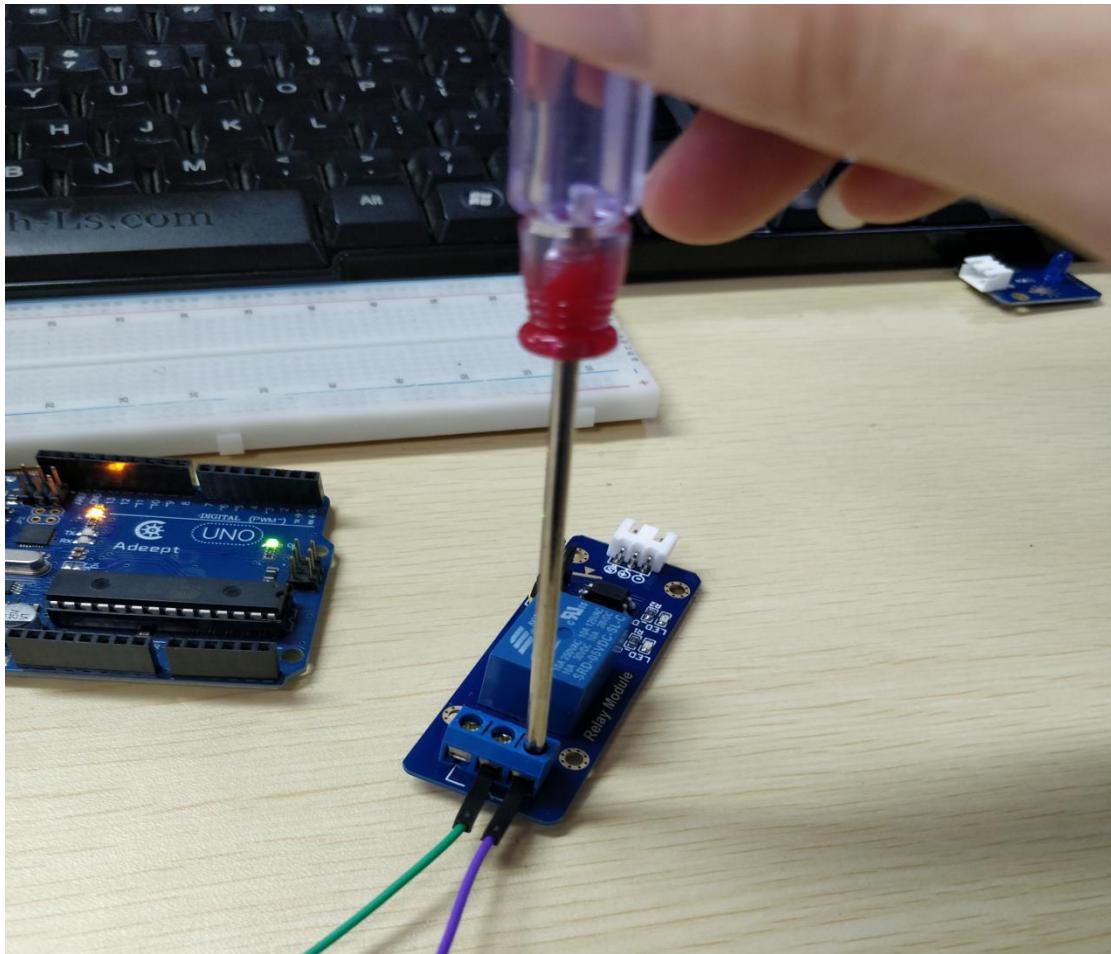
Before the experiment, we connected them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. Use 220Ω resistor. As shown in the following figure:





【pay attention】

When the relay is connected, you need to use a screwdriver to open the position of the two nuts, as shown below:



4. Controlling the Relay Module

We provide two different methods to control the active buzzer. One is to program the active buzzer on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program the active buzzer on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

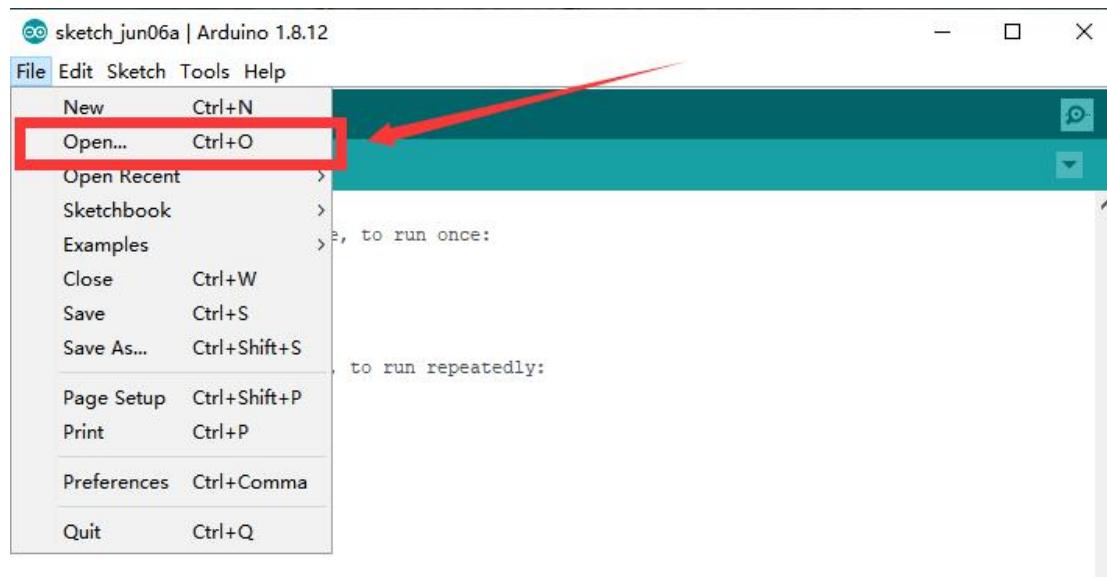
1.Using C language to program Relay Module to control LED on Arduino UNO

(1)Compile and run the code program of this course

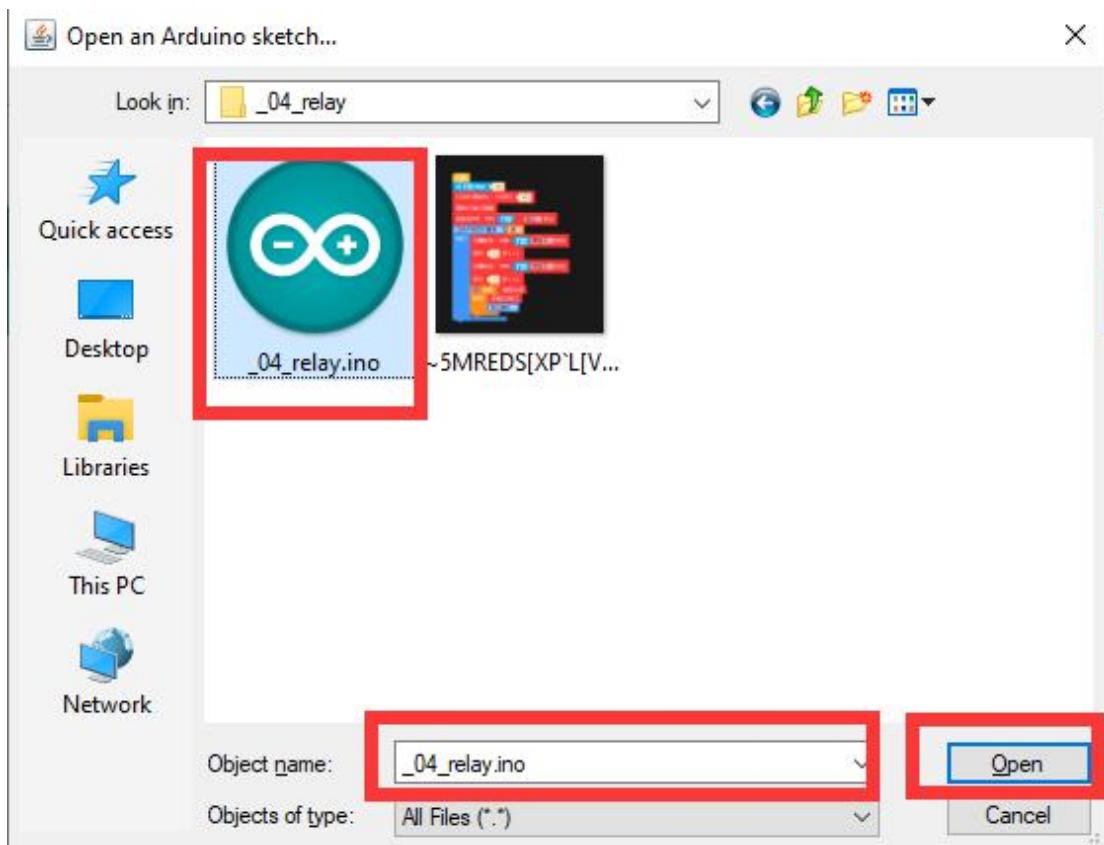
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\04_relay directory. Select _04_relay.ino. This file is the code program we need in this course. Then click Open.



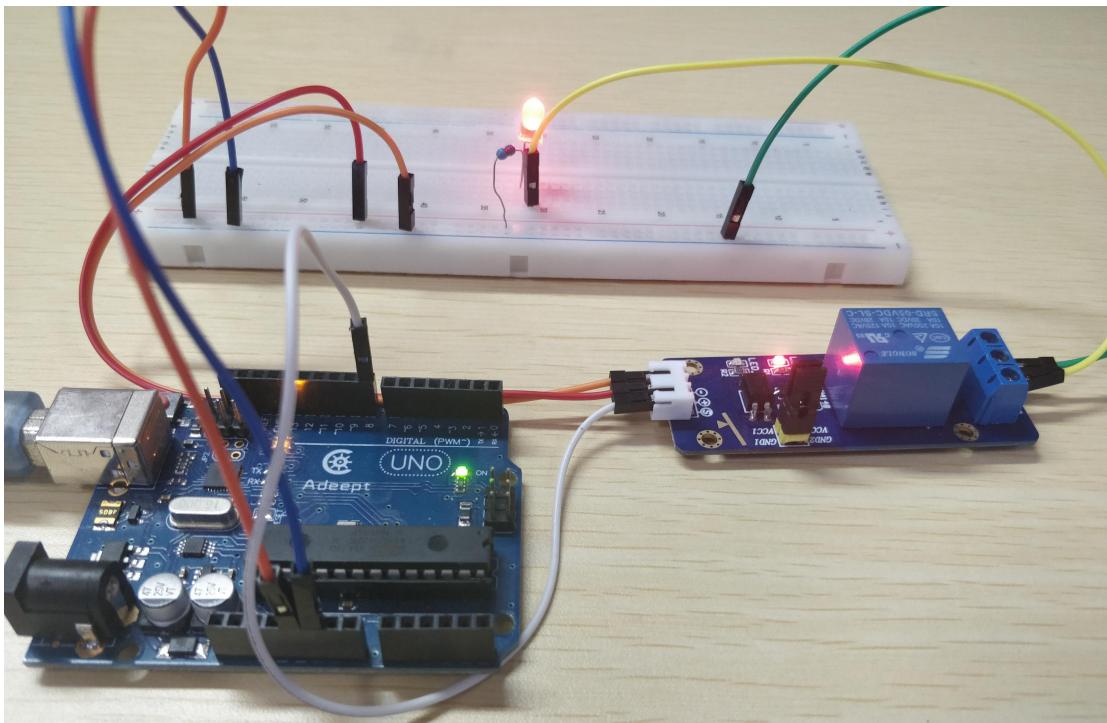
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. If Relay Moudle will make a sound during the experiment and the LED will be on and off at the same time. This shows that our experimental test is successful. The physical connection diagram of the experiment is as follows:



(2)The core code program

After the above hands-on operation, you must be very interested to know how we use C language to program Relay Module to control LED on Arduino UNO. We will introduce how our core code can be achieved:

1. Set relayPin to Output mode through pinMode(relayPin, OUTPUT) in the setup() method;
2. In the loop() method, set relayPin to HIGH through digitalWrite(relayPin, HIGH). At this time, the Relay Module is closed and powered on, and the circuit is on, and then the LED lights up. Set relayPin to LOW through digitalWrite(relayPin, LOW). At this time, the Relay Module closes the circuit and the LED goes out.

```

const int relayPin = 8; //the base of the transistor attach to

void setup()
{
    pinMode(relayPin, OUTPUT); //initialize the relayPin as an output
}

void loop()
{
    digitalWrite(relayPin, HIGH); //drive relay closure conduction
    delay(1000); //wait for a second

    digitalWrite(relayPin, LOW); //drive the relay is closed off
    delay(1000); //wait for a second
}

```

2. Controlling LED with Relay Module by programming on Arduino UNO with graphical code blocks

(2)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

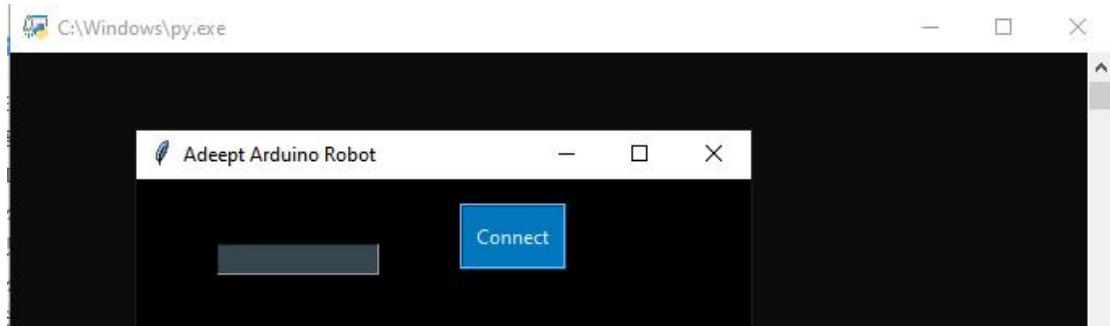
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

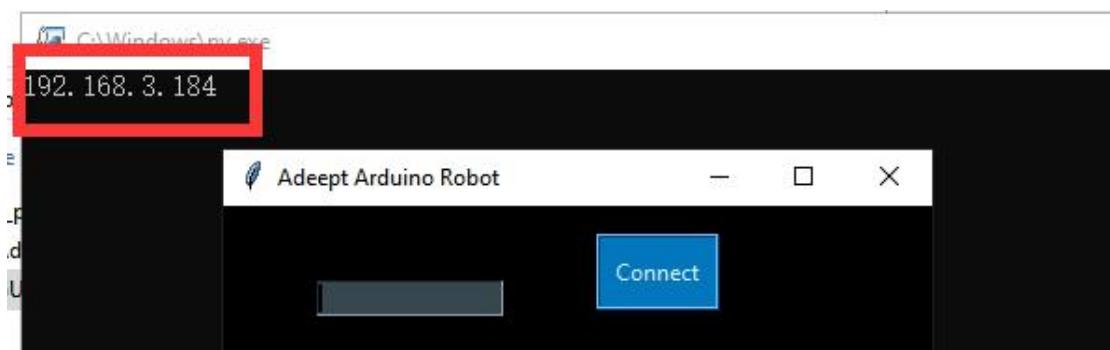
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

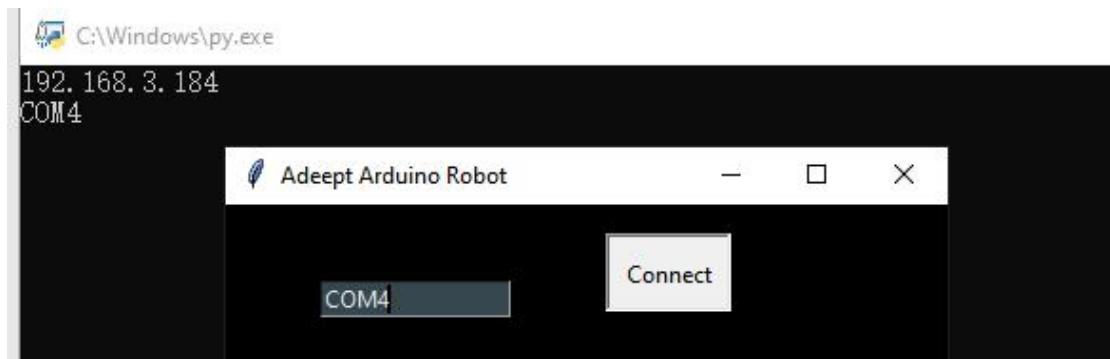
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



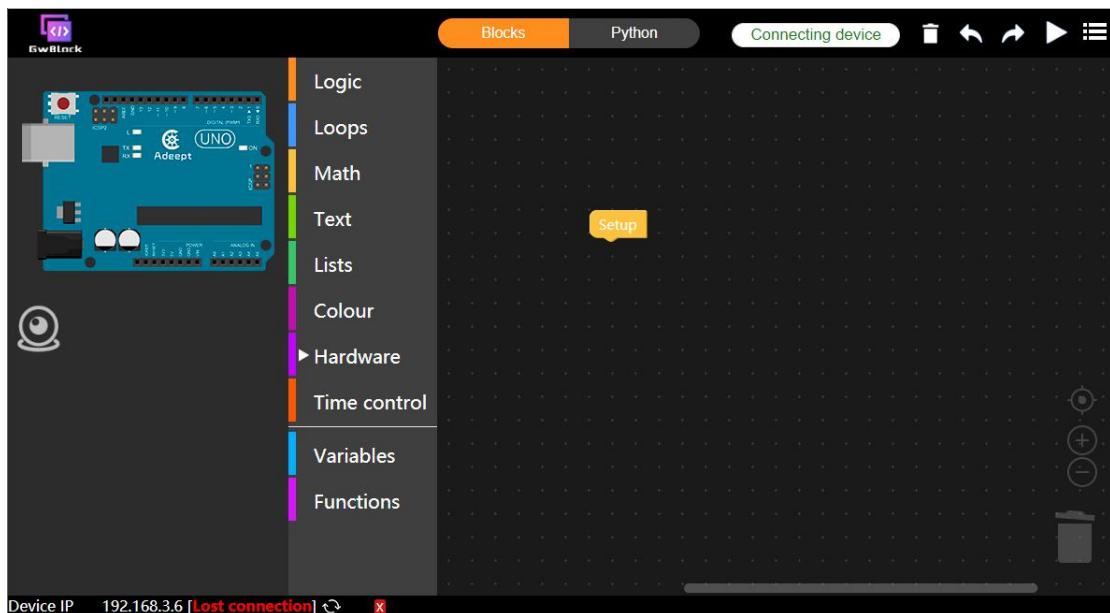
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



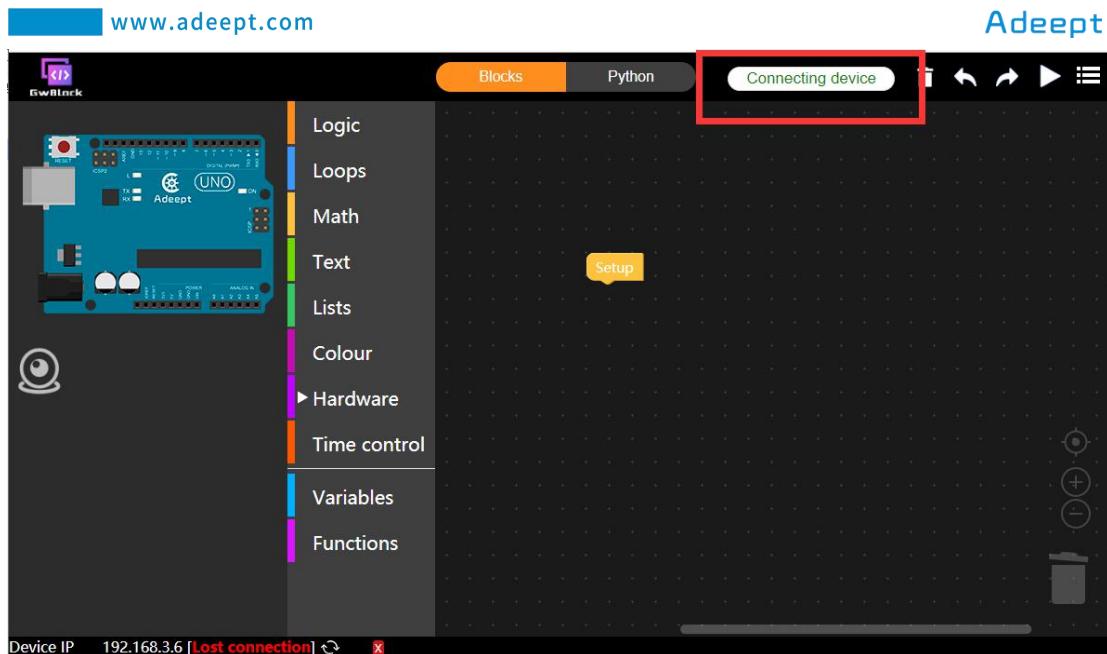
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

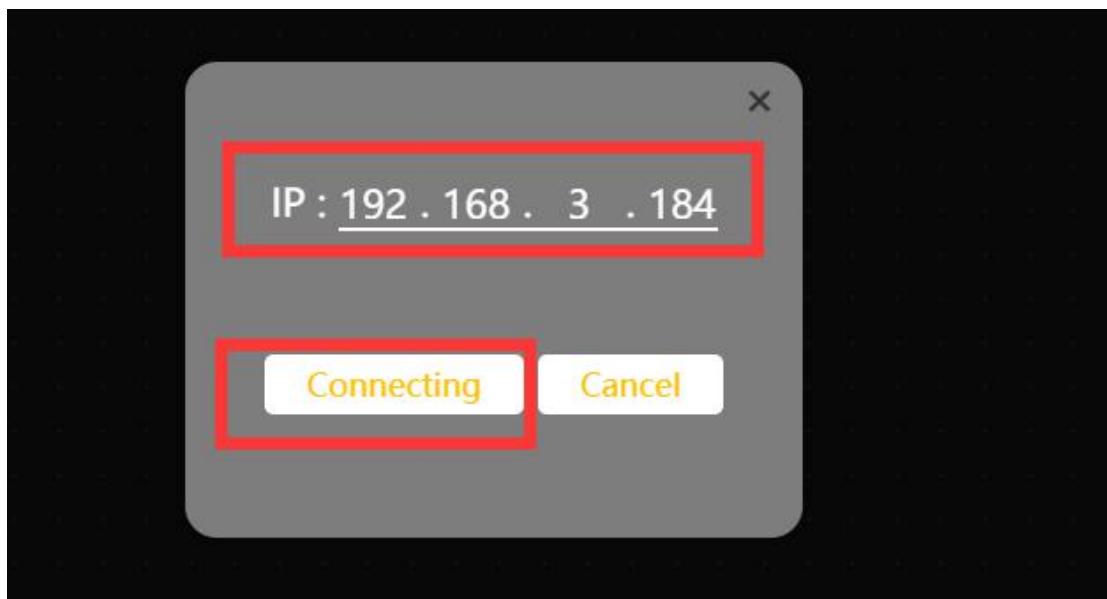
After successfully entering the website, the interface is as follows:



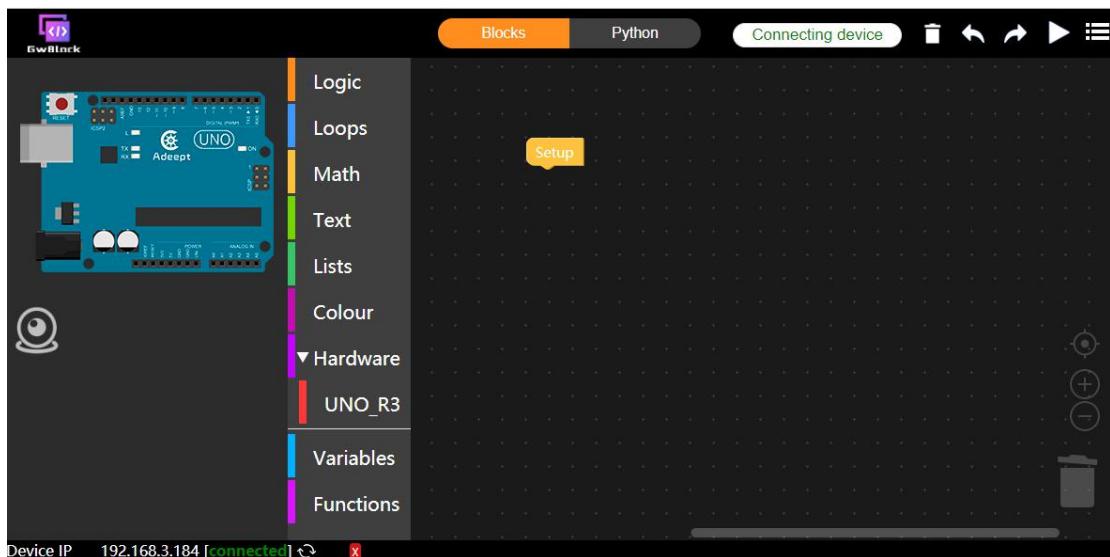
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

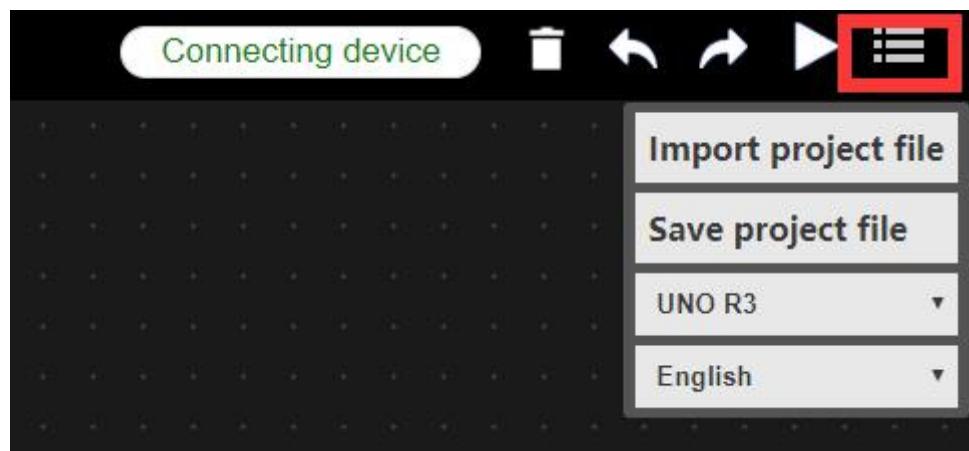


(2)Core code program

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

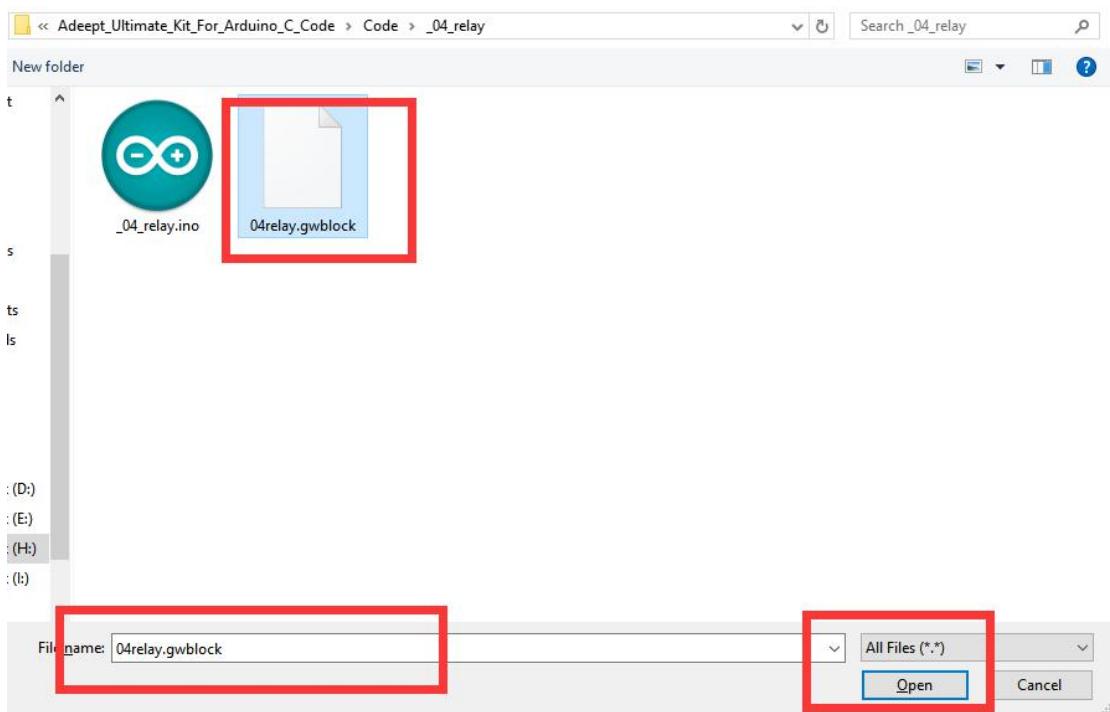
Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_04_relay folder. It will show as below:

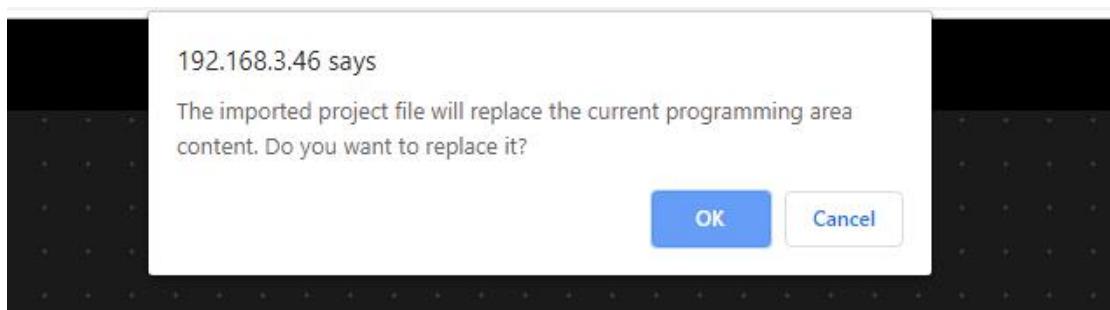
Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	



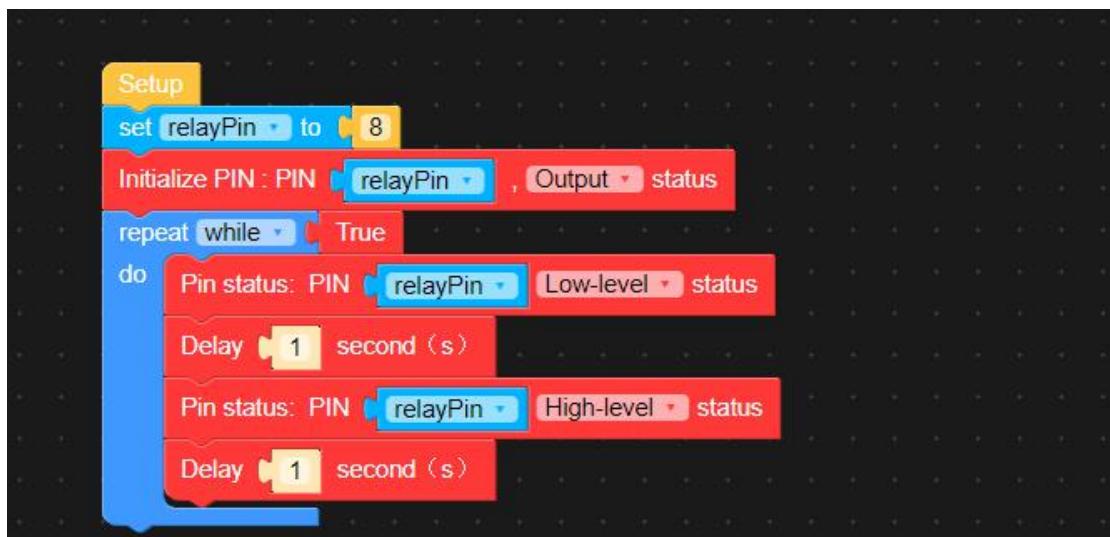
5. Select the "04relay.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



6.Click OK.It will show as below:

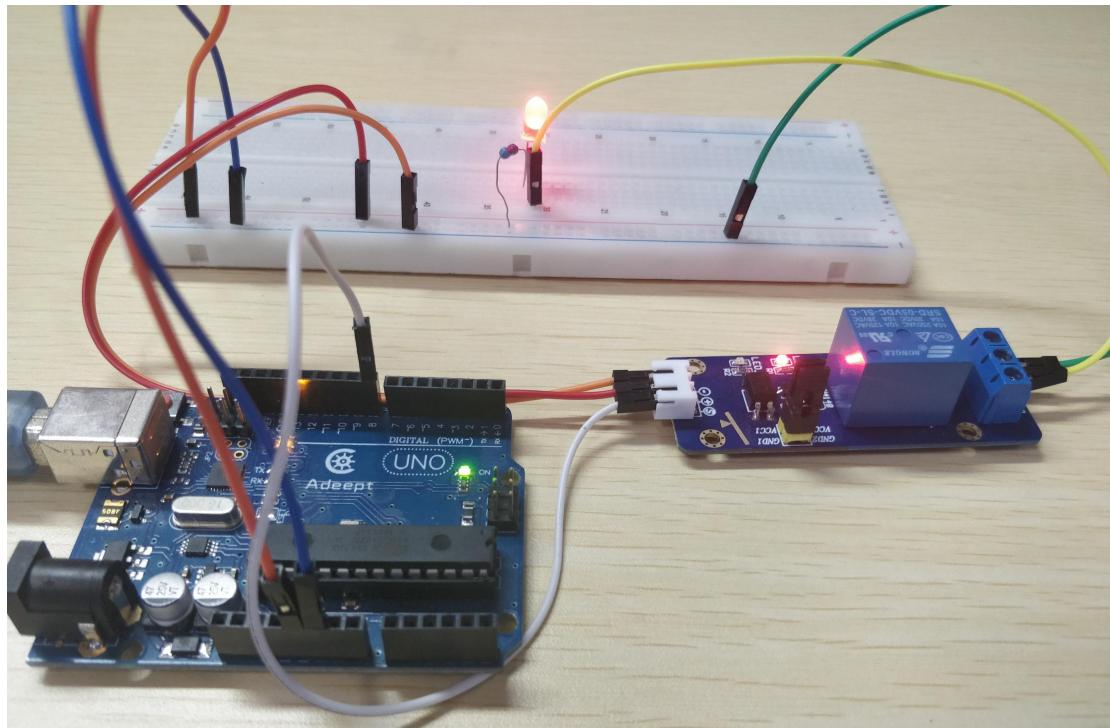


7.It will show as below after successfully opening:



8. Click the run program button  in the upper right corner and observe the

experiment phenomenon. You will find that the relay module will make a sound, and the LED will be on and off, indicating that the experiment test is successful. The physical connection of the experiment is as follows:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we control LED with Relay Module by programming on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

In the GwBlock graphical editor, all code programs are executed from **Setup**. First initialize the `relayPin` pin number to 8 through the instruction block **set relayPin to 8**, and the `relayPin` is set to the Output mode through the instruction block **Initialize PIN : PIN relayPin , Output status**; in the while loop statement, sets `relayPin` to Low-level state through the instruction block **Pin status: PIN relayPin Low-level status**. At this time, the Relay Module is closed and connected, and the LED will light up, delaying 1s through the instruction **Delay 1 second (s)**; similarly, the `relayPin` is set to High-level state

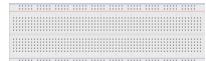
through the instruction block Pin status: PIN relayPin High-level status . At this time, the Relay Module is in the off state and the LED will be off, delaying 1s through the instruction Delay 1 second (s) .



Lesson 5 Controlling LED with Potentiometer

In this lesson, we will learn how to control LED with Potentiometer.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	1	
LED	1	
Potentiometer(10KΩ)	1	

2. Introduction of Potentiometer

(1) Potentiometer

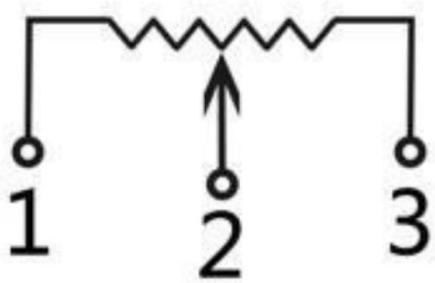
The potentiometer is a resistance element with three terminals and the resistance value can be adjusted according to a certain change law, which is equivalent to a variable resistor. Because its role in the circuit is to obtain a certain relationship with the input voltage (external voltage) to output Voltage, so called potentiometer. Potentiometers can be divided into rotary potentiometers, push-pull potentiometers, straight slide potentiometers, etc. according to the adjustment method. Our course experiment uses a rotary potentiometer. Its three pins are showed as below:



(2)Working principle of potentiometer

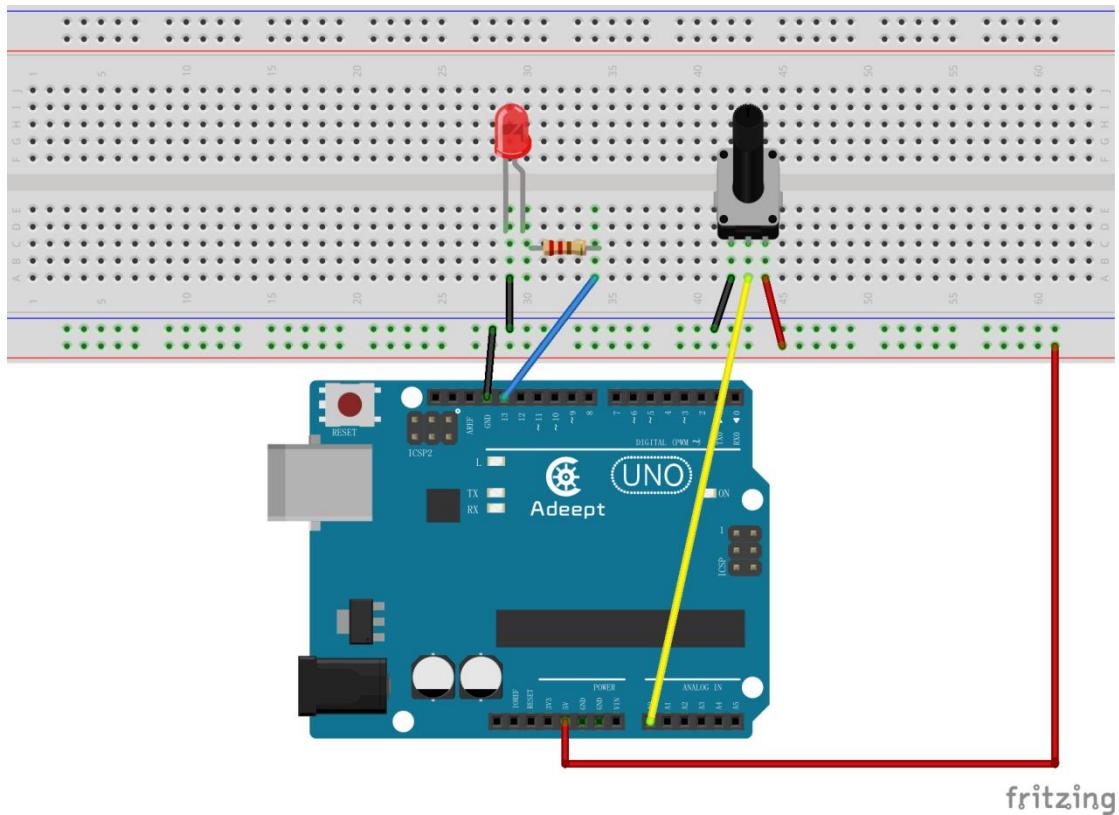
The rotary potentiometer is an adjustable resistance element. It is composed of a resistor and a rotating system. When a voltage is applied between the two fixed contacts of the resistive body, the position of the contact on the resistive body is changed by the rotating system, and a voltage that has a certain relationship with the position of the moving contact can be achieved between the moving contact and the fixed contact. Potentiometer can be used to adjust the voltage and current.

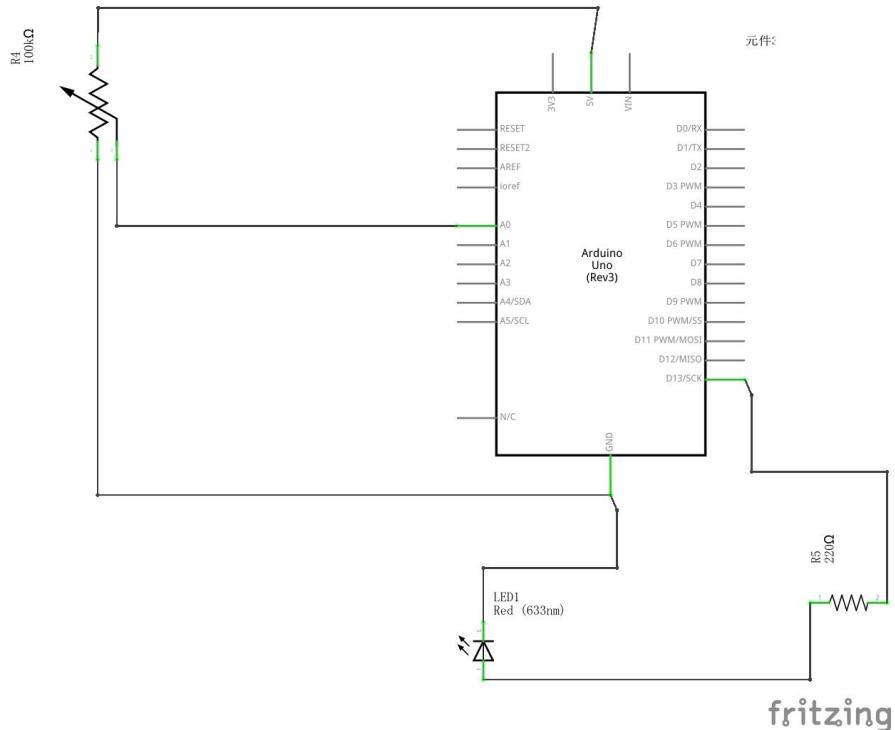
Our course uses a rotary potentiometer. Its structure is as shown in the figure below. By rotating the knob, the position of pin 2 is changed, thereby changing the resistance value from pin 2 to both ends. In the experiment. Connect pin 1 and pin 3 to the GND and 5V of the development board respectively. And then read the voltage divided by the pin 2 of the potentiometer through the analog input pin A0. The range is between 0V and 5V. The analog input function of Arduino has 10-bit precision, that is, it can convert the voltage signal of 0 to 5V into an integer form of 0 to 1024.



3.Wiring diagram (Circuit diagram)

Before the experiment, we connected them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. Use 220Ω resistor. As shown in the following figure:





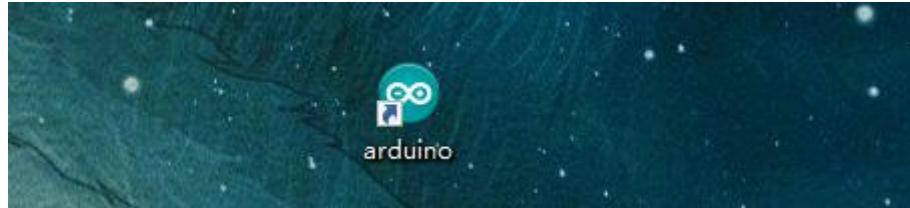
4. Controlling LED with Potentiometer

We provide two different methods to control the active buzzer. One is to program the active buzzer on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program the active buzzer on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

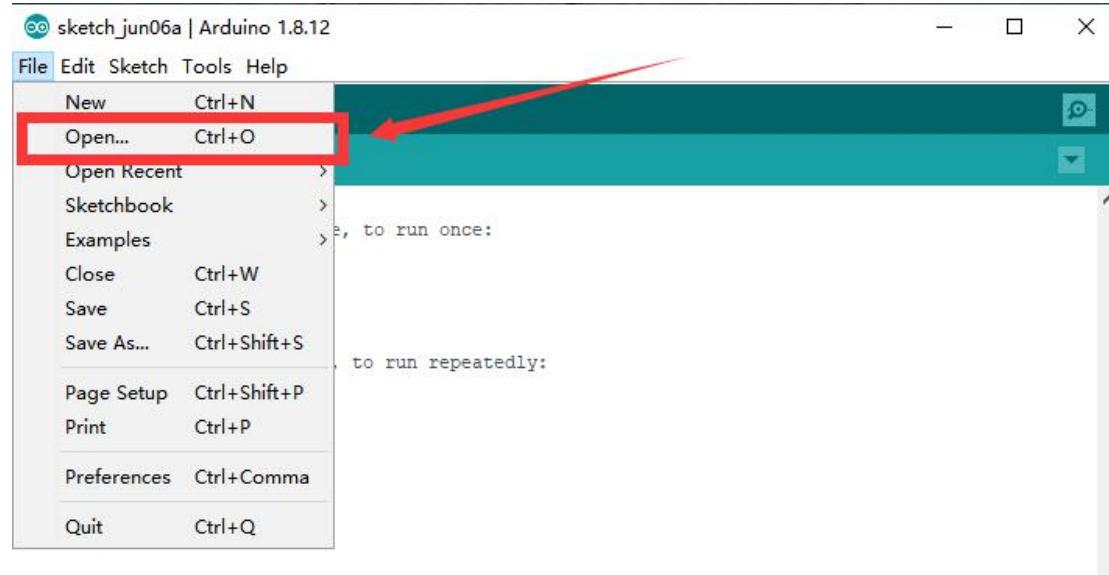
1. Using C language to program Potentiometer to control LED on Arduino UNO

(1) Compile and run the code program of this course

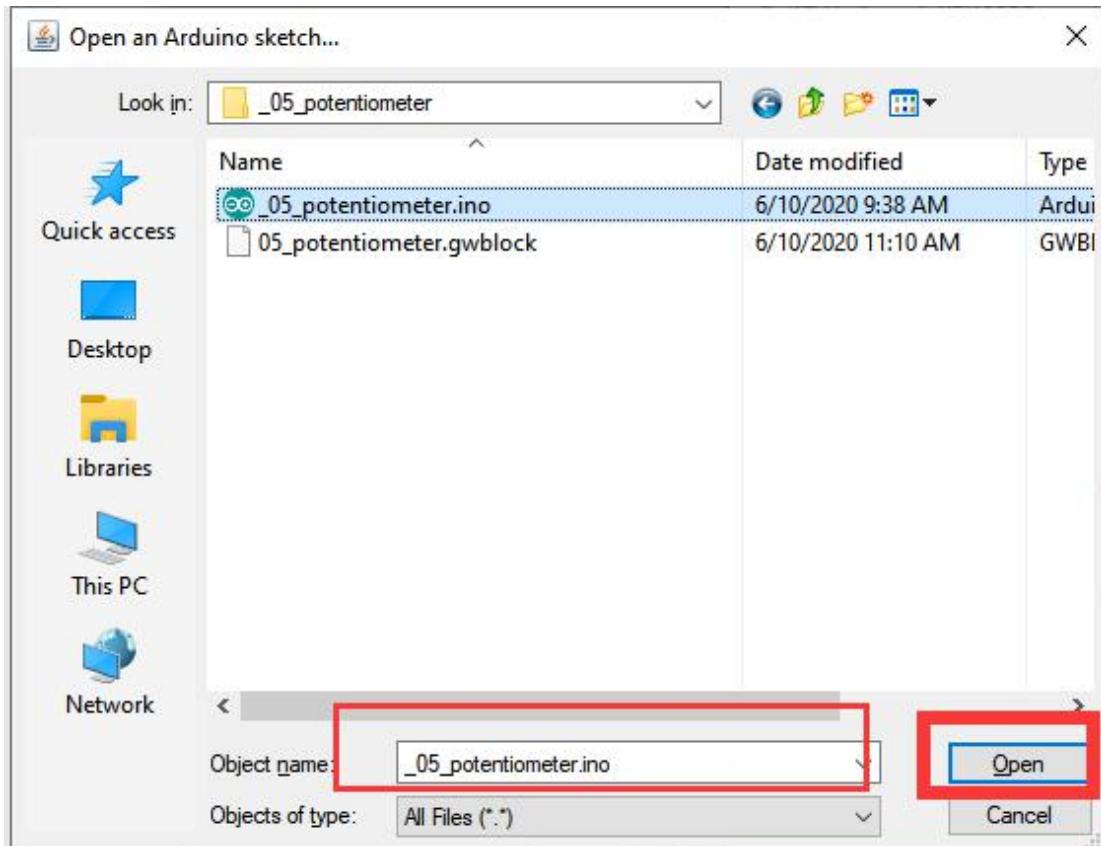
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\05_potentiometer directory. Select _05_potentiometer.ino. This file is the code program we need in this course. Then click Open.



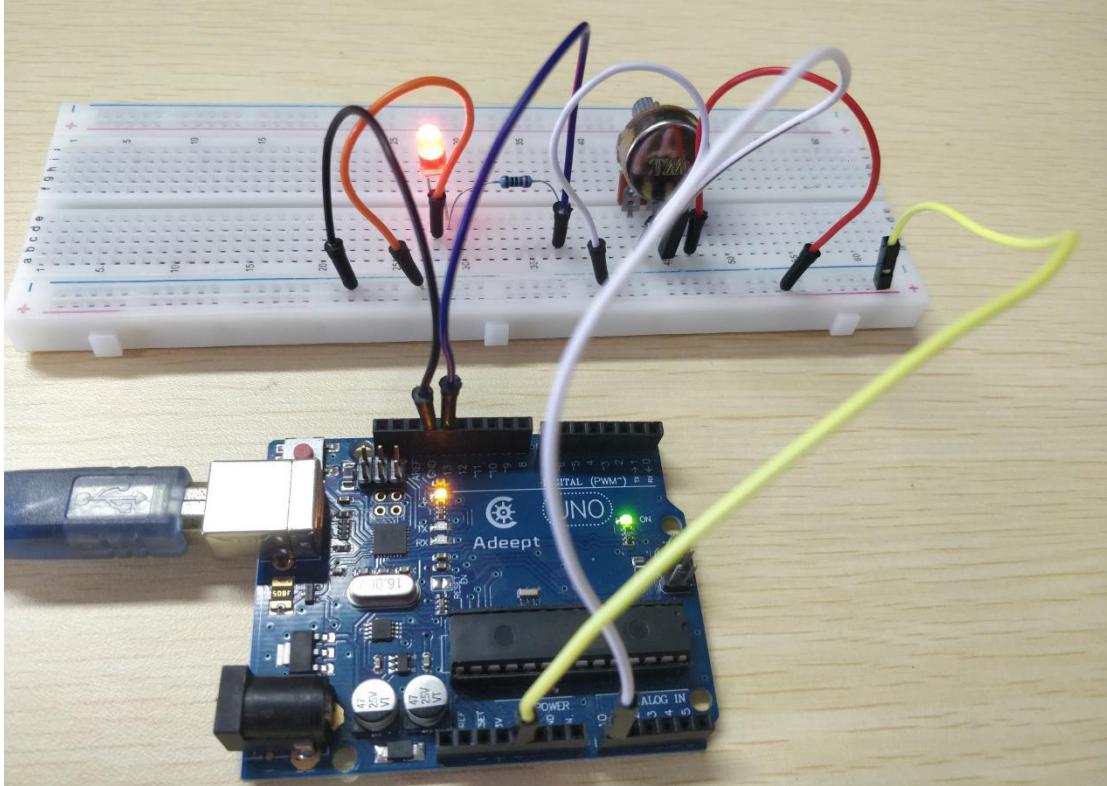
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After running the program, turn the potentiometer counterclockwise to light up the LED; turn the potentiometer clockwise to turn off the LED. This shows that our experimental test is successful. The physical connection diagram of the experiment is as below:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to control LED with Potentiometer on Arduino UNO. We will introduce how our core code can be achieved:

1. Set the relayPin pin to Output mode through `pinMode(relayPin, OUTPUT)` in the `setup()` method;
2. Use the `analogRead()` function in the `loop()` method to read the voltage value input by the potentiometer. And then determine whether the read value is greater than 500 through the if judgment statement. When it is greater than 500, we set the relayPin pin to HIGH through `digitalWrite(relayPin, HIGH)`, and the LED will be on; when it is less than 500, set relayPin to LOW through `digitalWrite(relayPin, LOW)`, and the LED will be off.

```

int ledpin=13;           //definition digital 13 pins as pin to control the LED
int potentiometer = 0;    //potentiometer PIN A0
void setup()
{
    pinMode(ledpin,OUTPUT);//Set digital 13 port mode, the OUTPUT for the output
}
void loop()
{
    int a = analogRead(potentiometer);
    if(a>500)
        digitalWrite(ledpin, HIGH);
    else
        digitalWrite(ledpin, LOW);
}

```

2. Using graphical code blocks to program to control LED with Potentiometer on Arduino UNO

(2)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

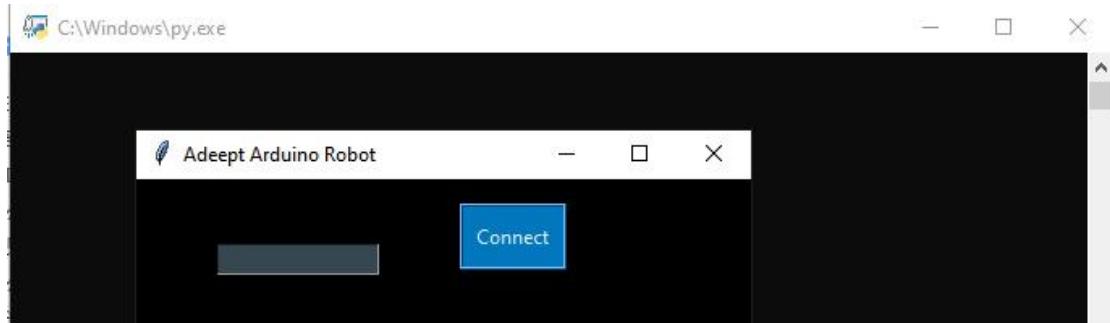
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

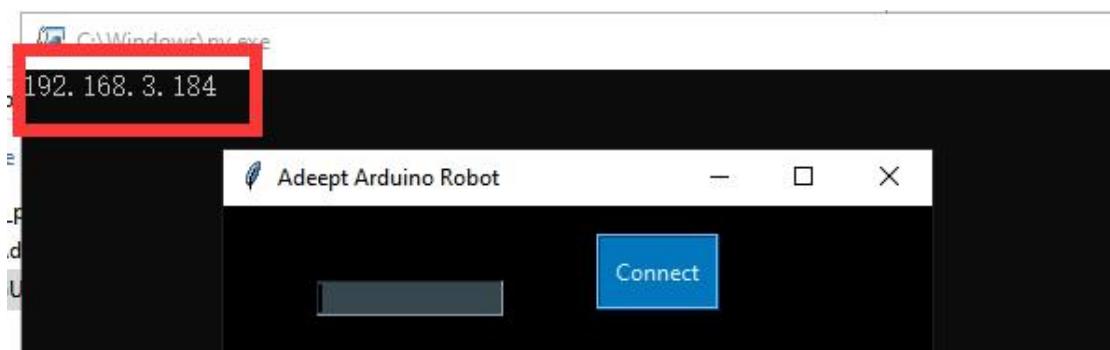
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

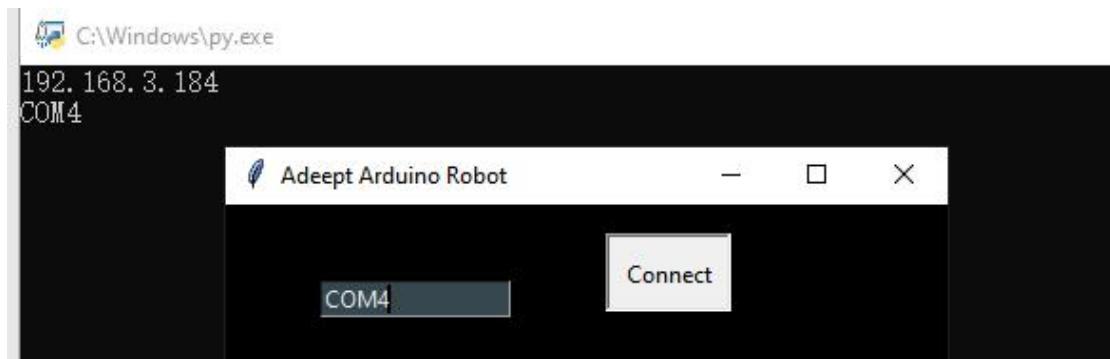
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



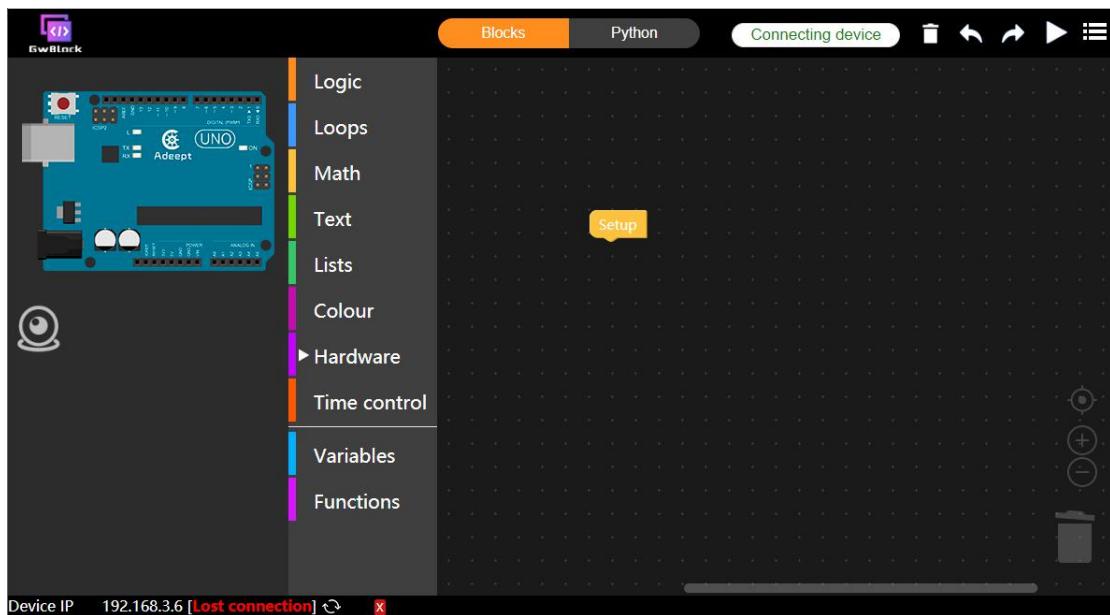
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



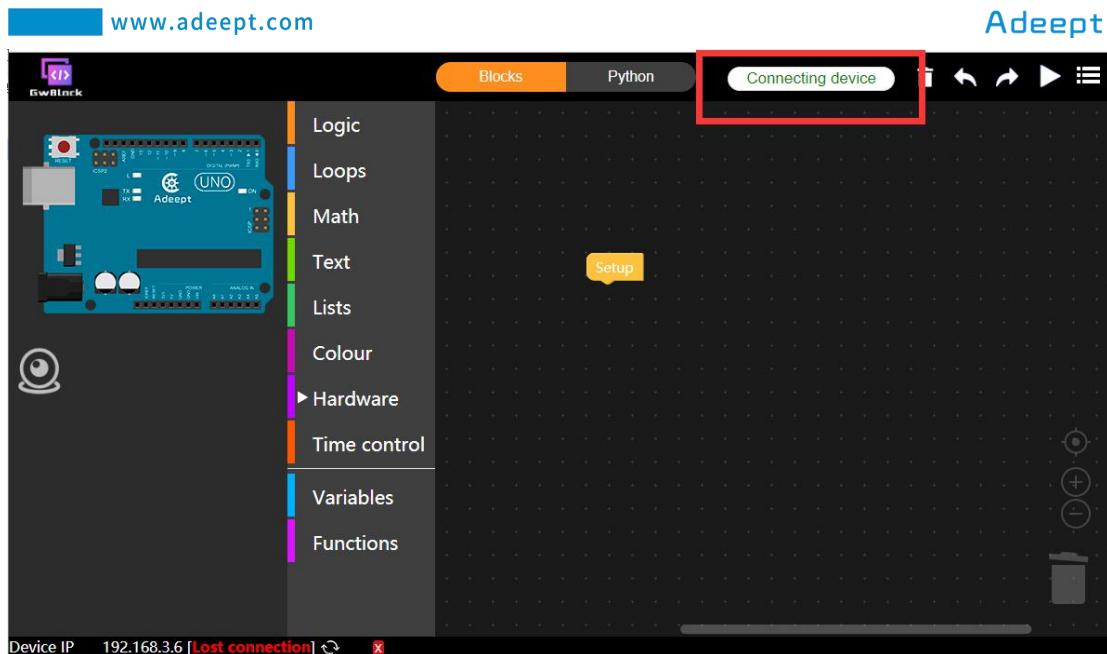
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

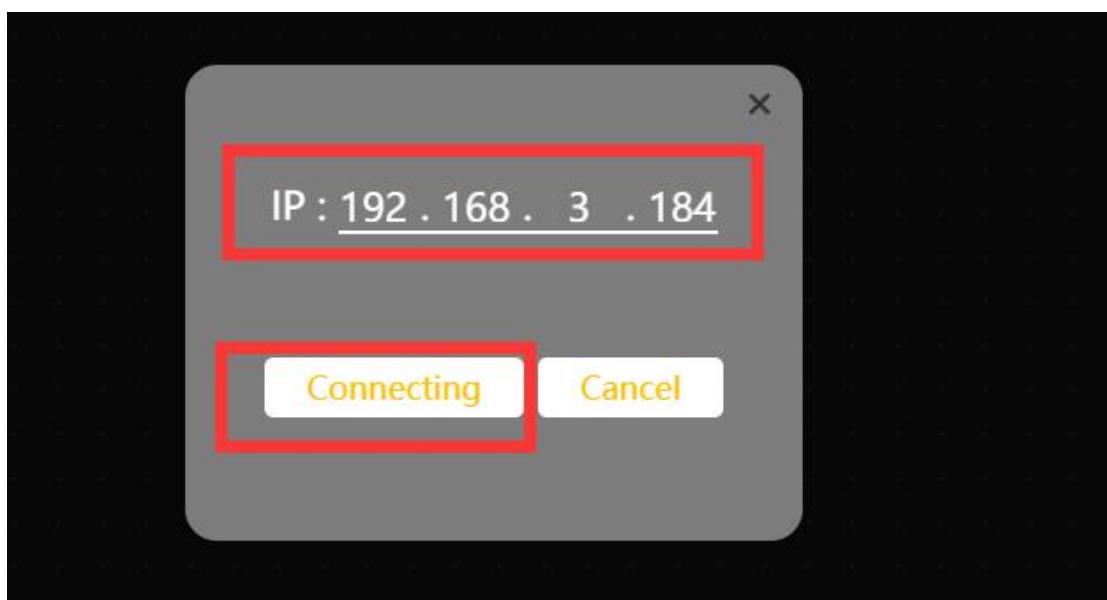
After successfully entering the website, the interface is as follows:



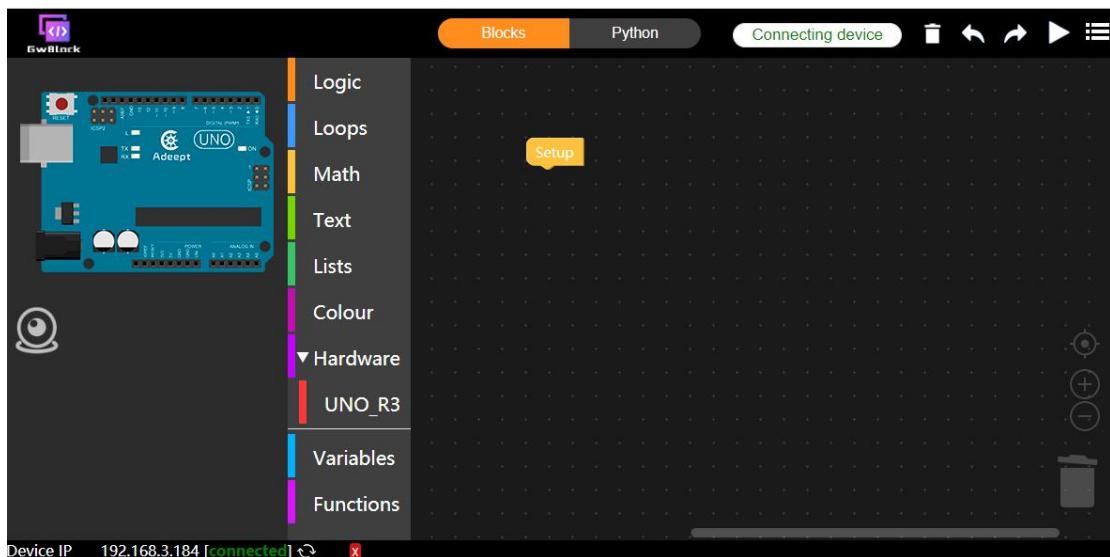
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

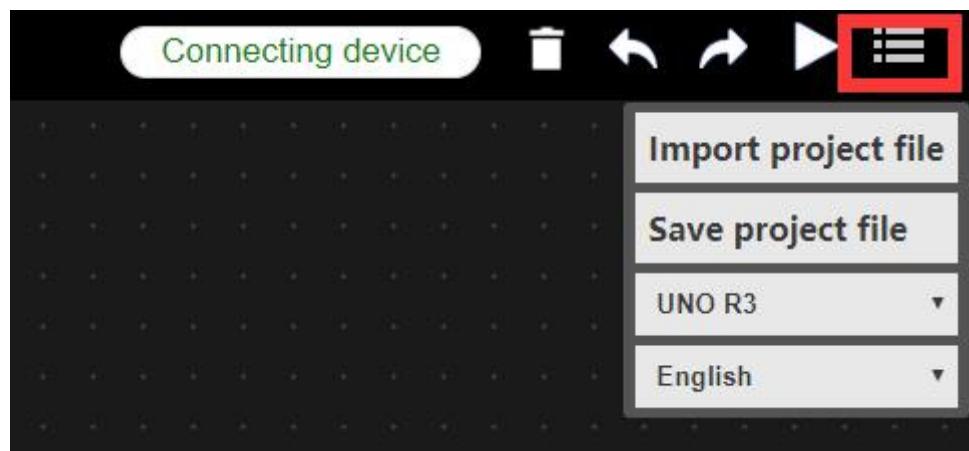


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

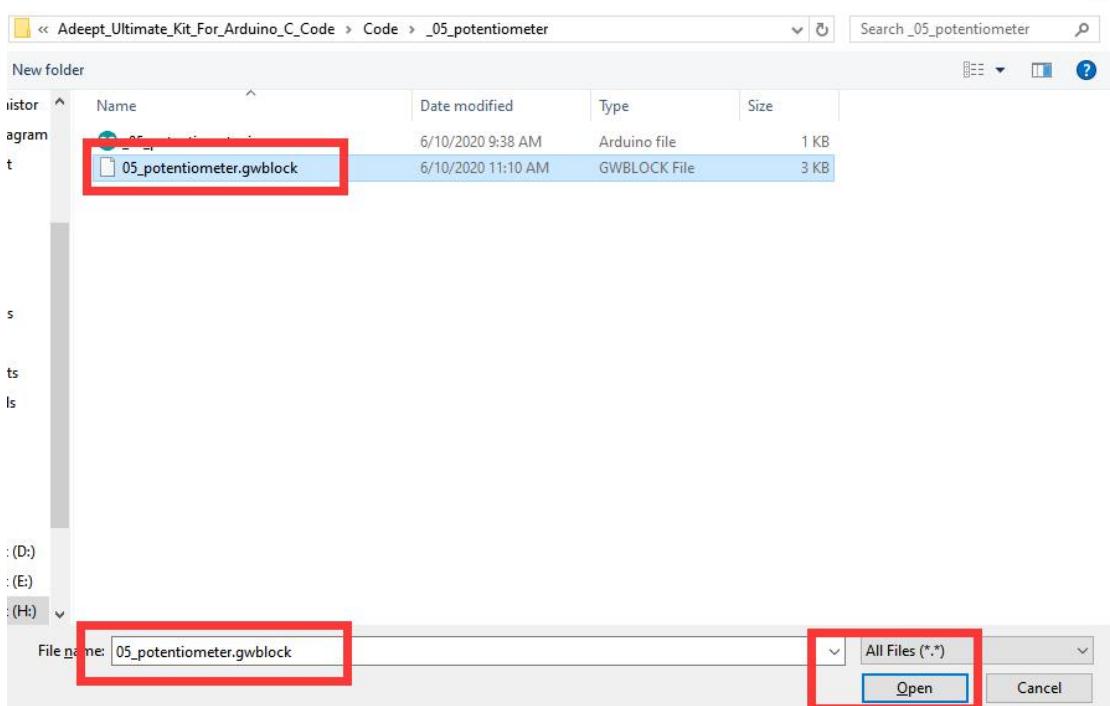
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_05_potentiometer

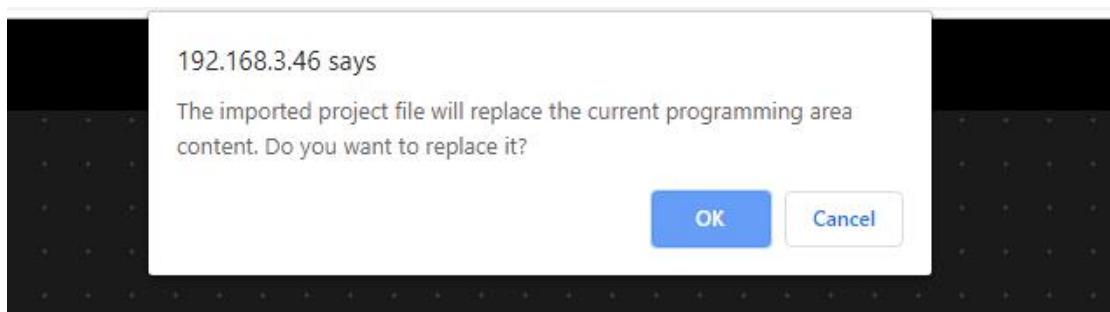
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

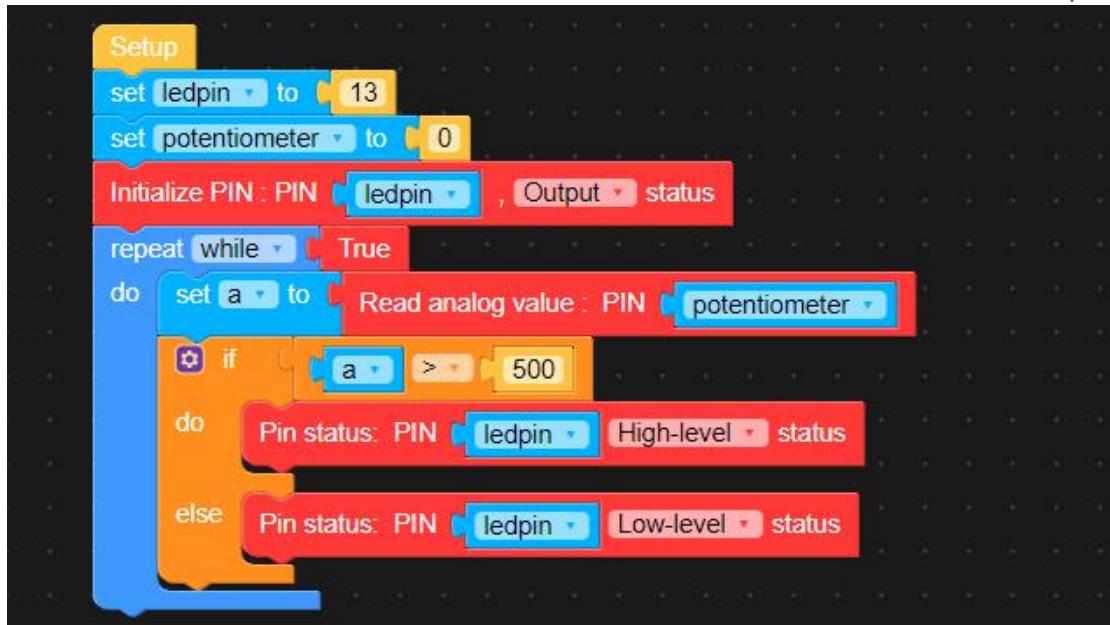
5. Select the "05_potentiometer.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



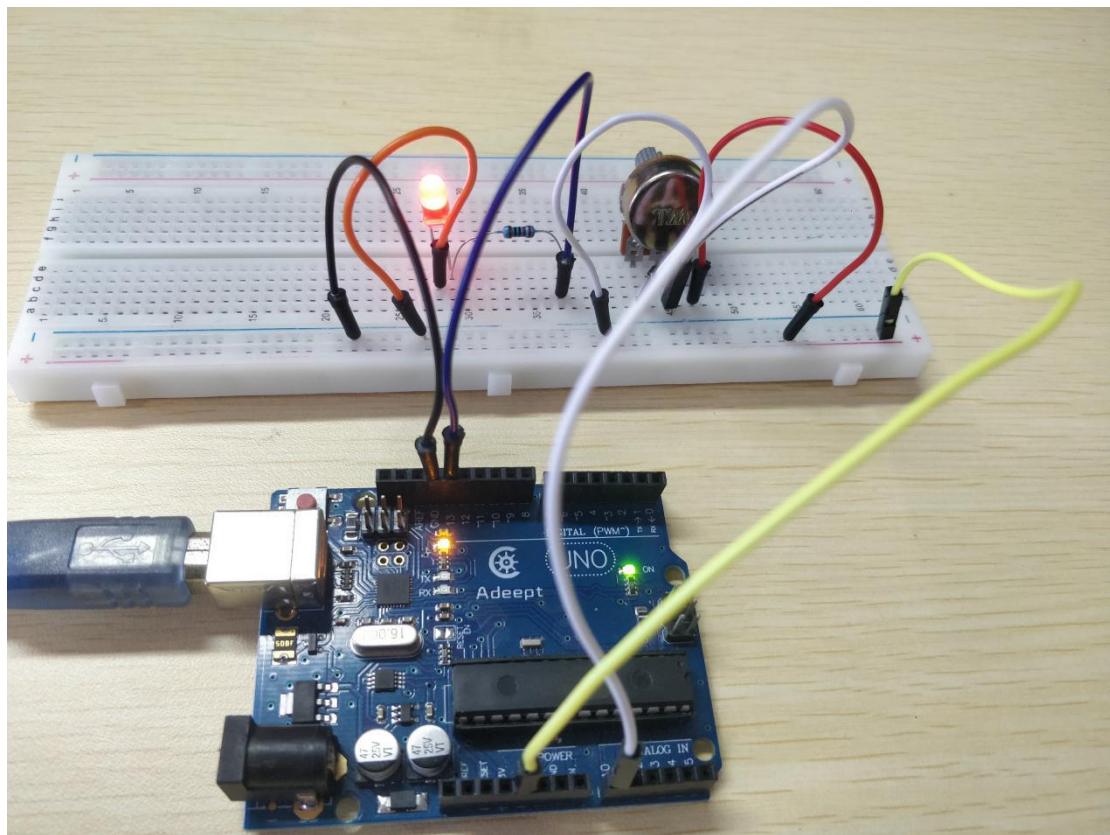
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



8. After running the program, turn the potentiometer counterclockwise to light up the LED; turn the potentiometer clockwise to turn off the LED. This shows that our experimental test is successful. The physical connection diagram of the experiment is as below:

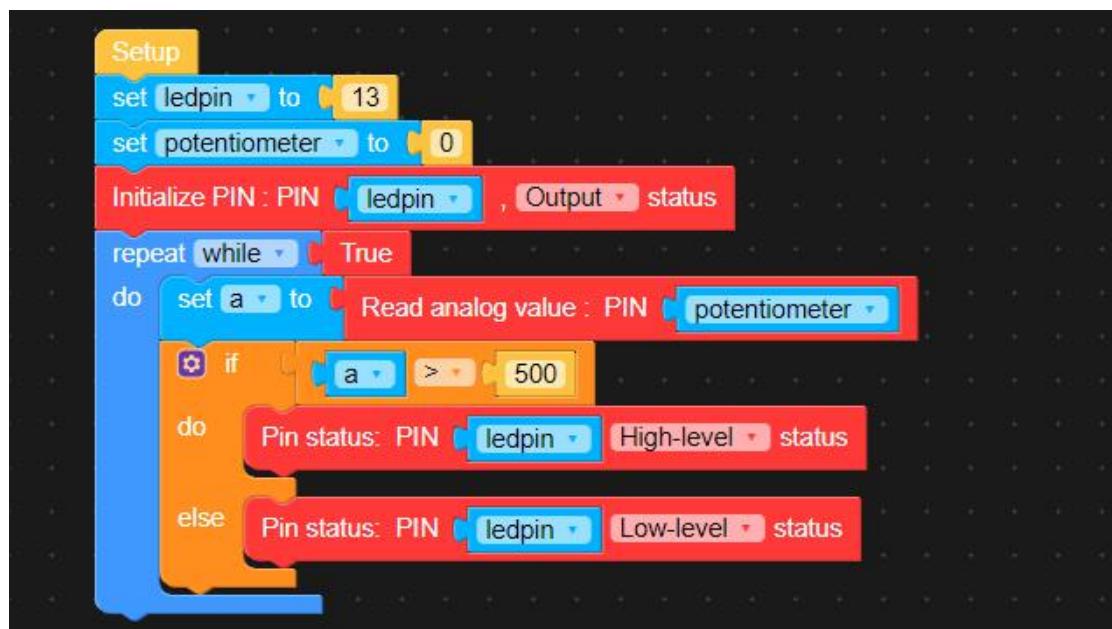


(3) Core code program

After the above hands-on operation, you must be very interested to know how we

control LED with Potentiometer by programming on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

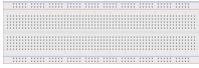
In the GwBlock graphical editor, all code programs are executed from **Setup**. First initialize the pin number of ledpin to 13 through the instruction block **set ledpin to 13**, and the ledpin is set to the Output mode through the instruction block **Initialize PIN : PIN ledpin , Output status**; In the while loop statement, read the voltage value input by the potentiometer through the instruction block **Read analog value : PIN potentiometer**; use the if judgment statement to judge whether the read value is greater than 500. If the obtained value is greater than 500, then set the ledpin to High-level through the instruction block **Pin status: PIN ledpin High-level status**, and the LED will be on. If the obtained value is less than 500, then set the ledpin to Low-level through the instruction block **Pin status: PIN ledpin Low-level status**, and the LED will be off.



Lesson 6 Making a Flowing LED with LED

In this lesson, we will learn how to make Flowing LED with LED.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	8	
LED	8	

2. The introduction of the Flowing LED

(1)What is a Flowing LED

Flowing lights refer to lights that are on and off in sequence according to the set order and time under the control of the microcomputer, visually feeling flowing of the lights. It is widely used in the decoration of architecture, interior and advertising signs.

(2)The principle of making the Flowing LED

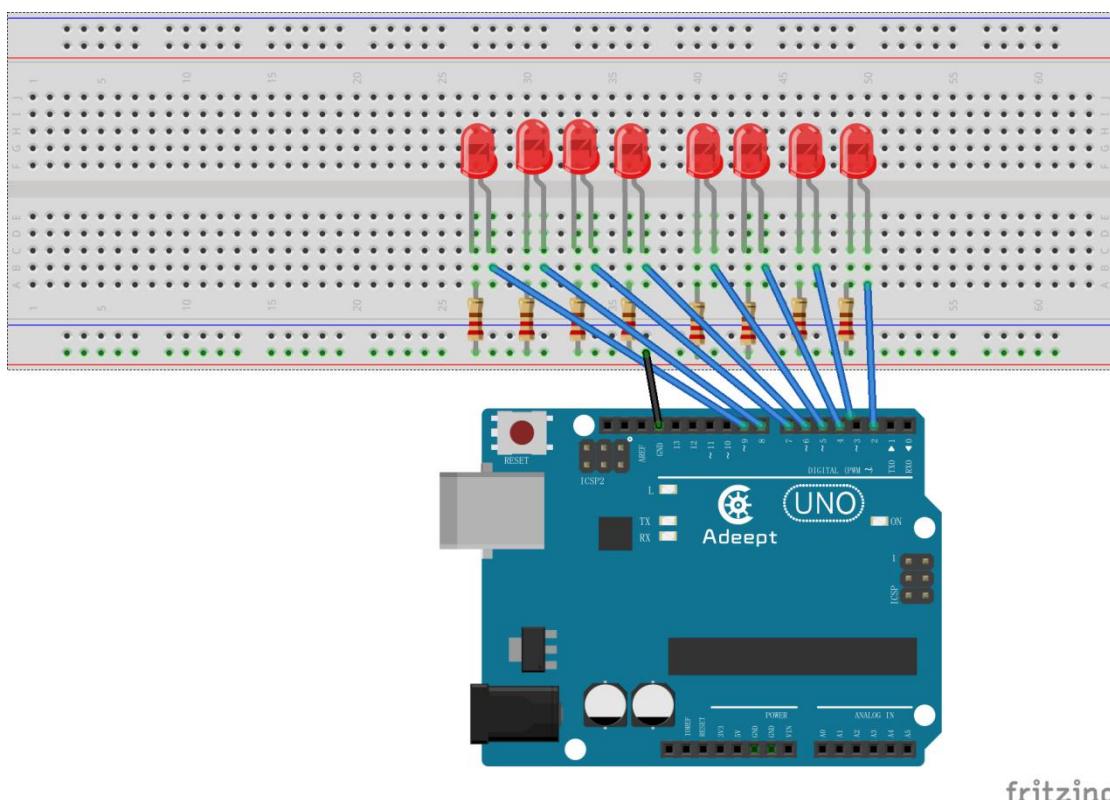
We programmed pin number to 2,3,4,5,6,7,8,9 which are connected to the Arduino UNO. Connect the pins to GND. During program design, by controlling their lighting sequence and delay time, from left to right, and then from right to left,

making 8 LEDs flash out of the effect of Flowing Lights.

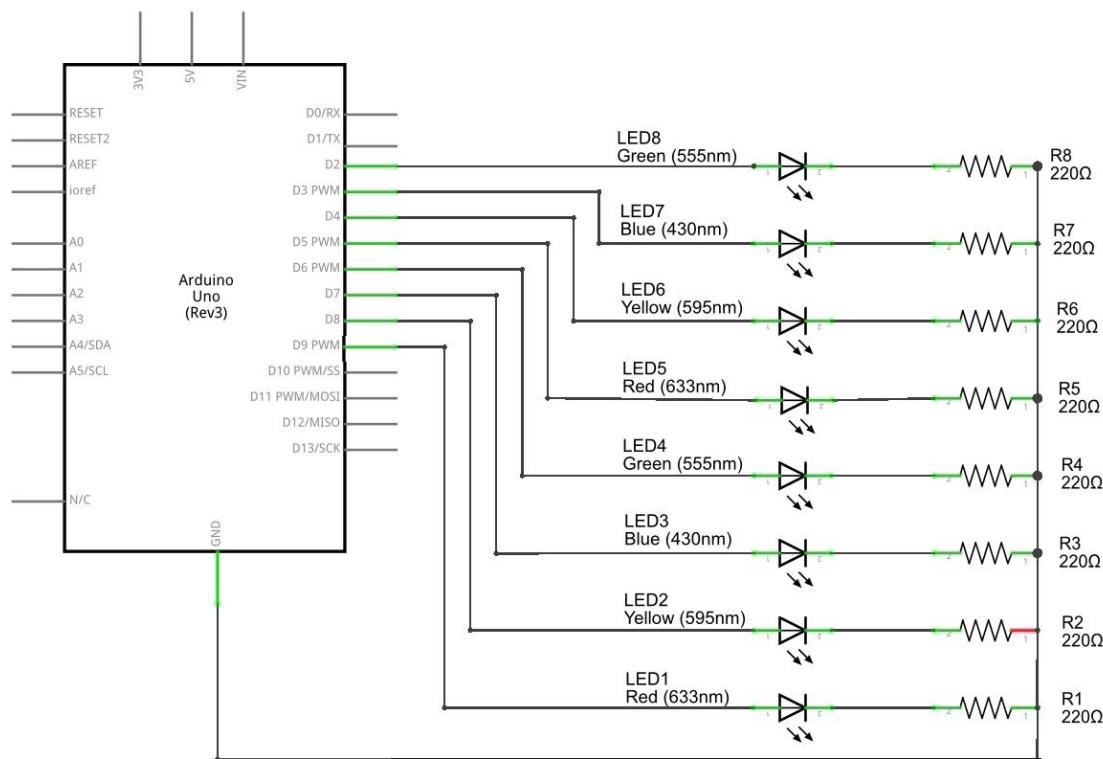
3.Wiring diagram (Circuit diagram)

Before the experiment, we connected them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes, as shown in the following figure:

Before the experiment, we connected them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. Use $220\ \Omega$ resistor. As shown in the following figure:



fritzing



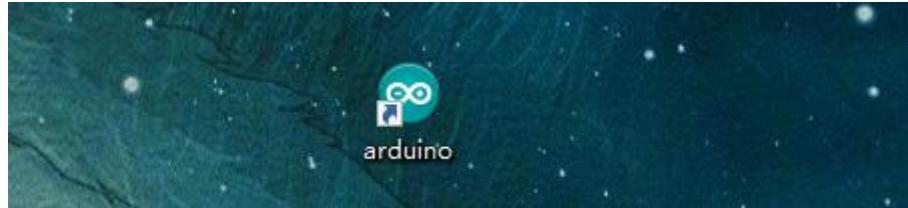
4. Making Flowing LED with LED

We provide two different methods to control the active buzzer. One is to program the active buzzer on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program the active buzzer on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

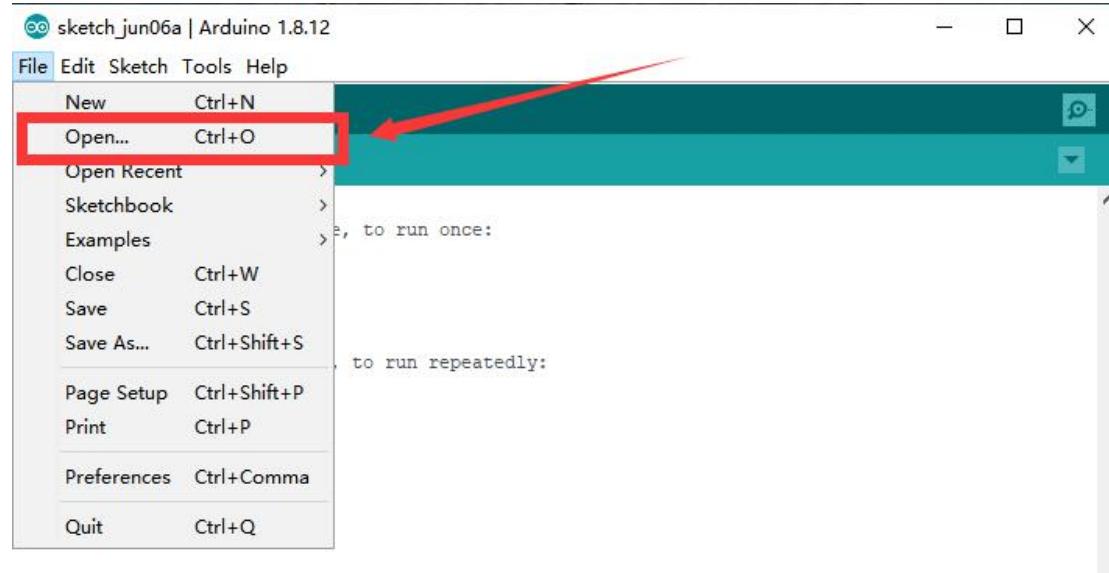
1. Using C language to program to make Flowing LED on Arduino UNO

(1) Compile and run the code program of this course

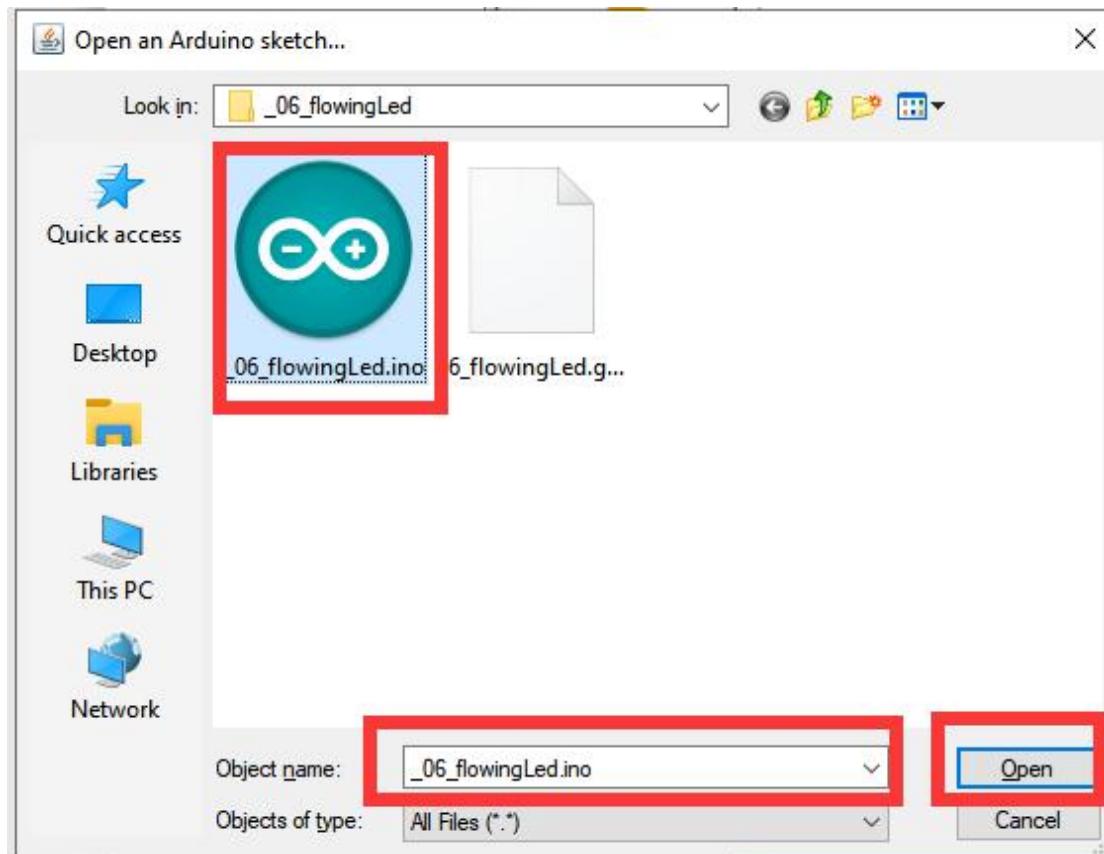
1. Open the Arduino IDE software, as shown below:



2.Click Open in the File drop-down menu:



3.Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\06_flowinLed directory. Select _06_flowinLed.ino. This file is the code program we need in this course. Then click Open.



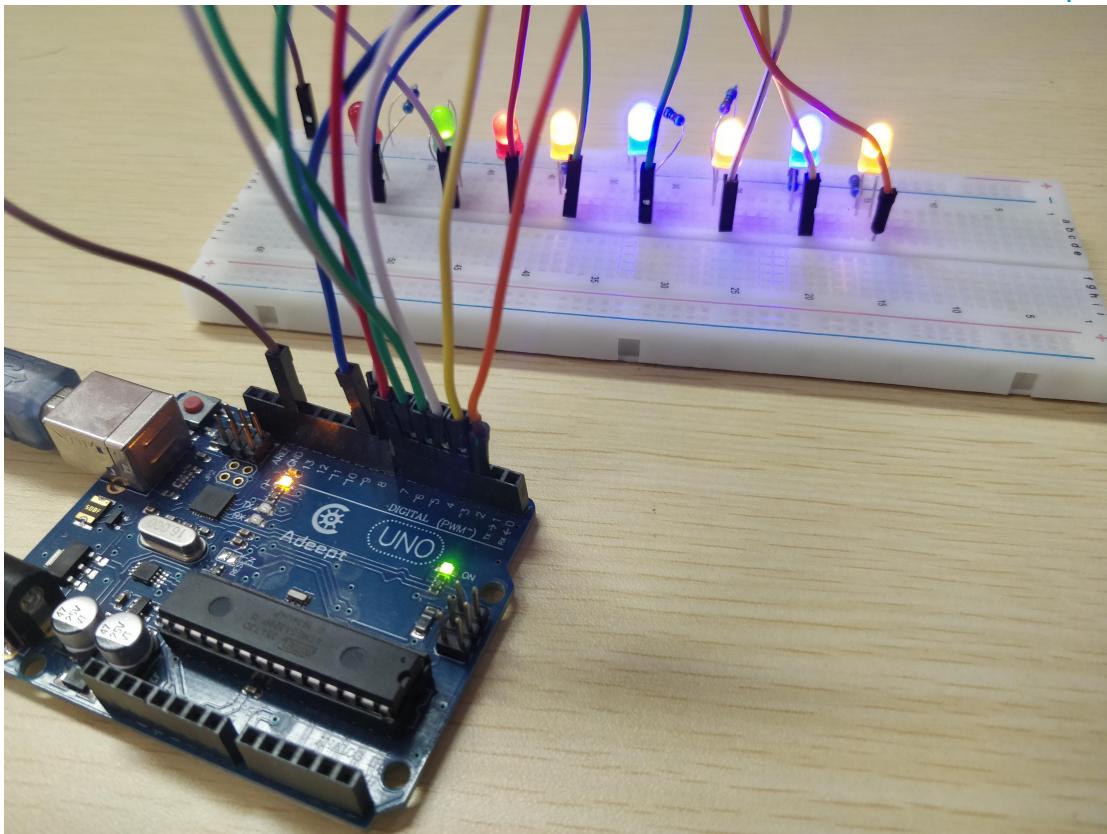
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After running the program, it can be observed that the LED will be on and off from right to left, then from left to right, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to make Flowing LED on Arduino UNO. We will introduce how our core code can be achieved:

2. Set the ledpin to Output mode through pinMode(ledpin, OUTPUT) in the setup() method;

```
void setup()
{
    []
    unsigned char ledPin;           //ledPin will be set to 2,3,4,5,6, 7 , 8, 9
    for(ledPin=2;ledPin<=9;ledPin++)//In turn set 2 ~ 9digital pins to output mode
        pinMode(ledPin,OUTPUT);      //Set the ledPin pin to output mode
}
```

2. In the loop() method, set the ledpin to HIGH through digitalWrite(ledpin, HIGH), then light the LED in sequence according to the pin numbers from 2 to 9, and delay 300ms through delay(300); then set ledpin to LOW through digitalWrite(ledpin, LOW), and the LED will be off; similarly, turn on and turn off the LED in order according to the pin number sequence of 9 to 2. In this way, the effect of Flowing LED is realized.

```

void loop()
{
    unsigned char ledPin;           //ledPin will be set to 2,3,4,5,6, 7 , 8, 9
    for(ledPin=2;ledPin<=9;ledPin++)//Every 200ms on in order LED2 ~ 9
    {
        digitalWrite(ledPin,HIGH);   //led on
        delay(300);
        digitalWrite(ledPin,LOW);    //led OFF
    }

    for(ledPin=9;ledPin>=2;ledPin--)//Every 200ms on in order LED2 ~ 9
    {
        digitalWrite(ledPin,HIGH);   //led on
        delay(300);
        digitalWrite(ledPin,LOW);    //led OFF
    }
}

```

2.Using graphical code blocks to program to make Flowing LED on Arduino UNO

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1.Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

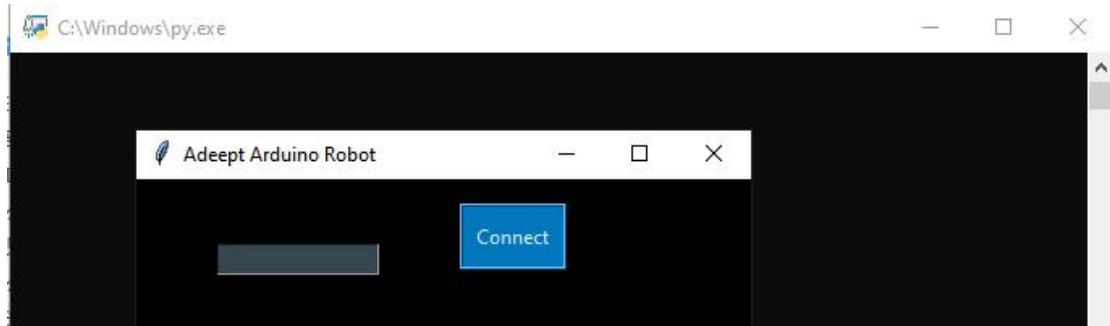
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

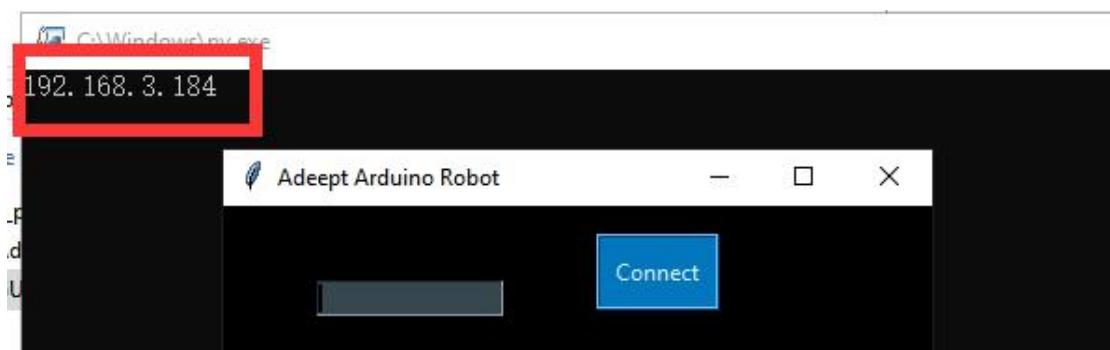
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

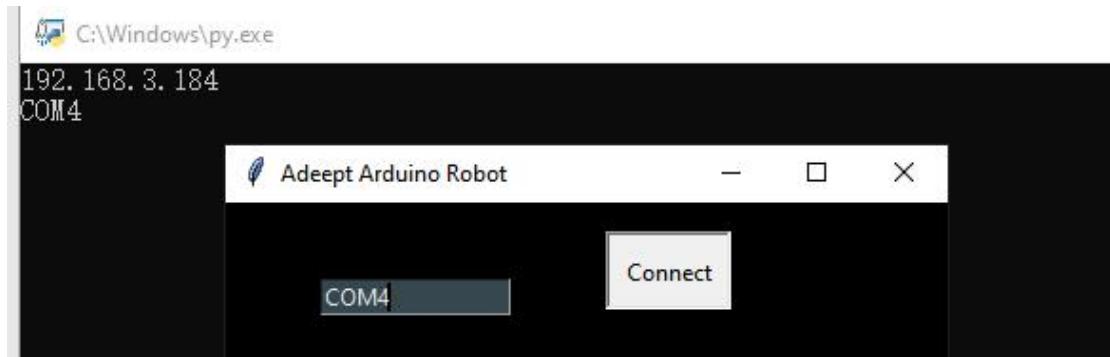
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



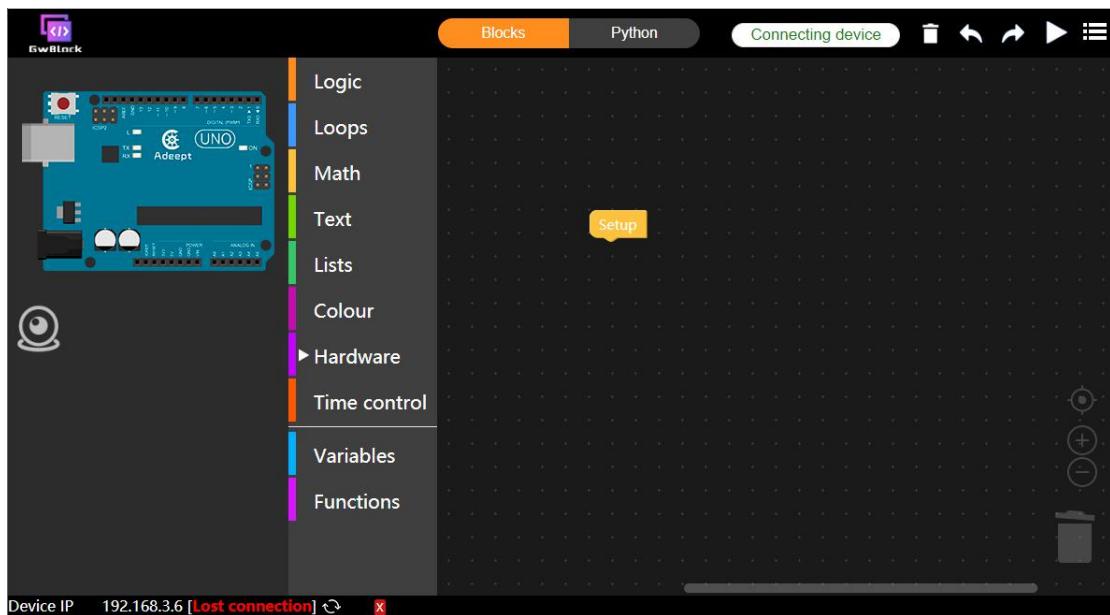
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



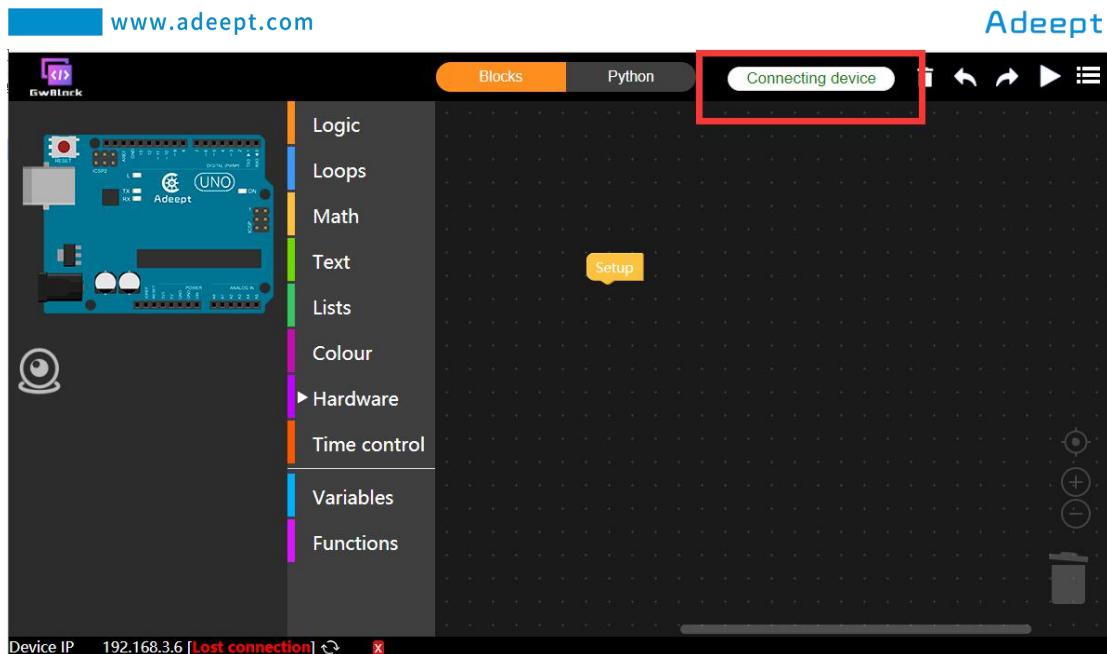
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

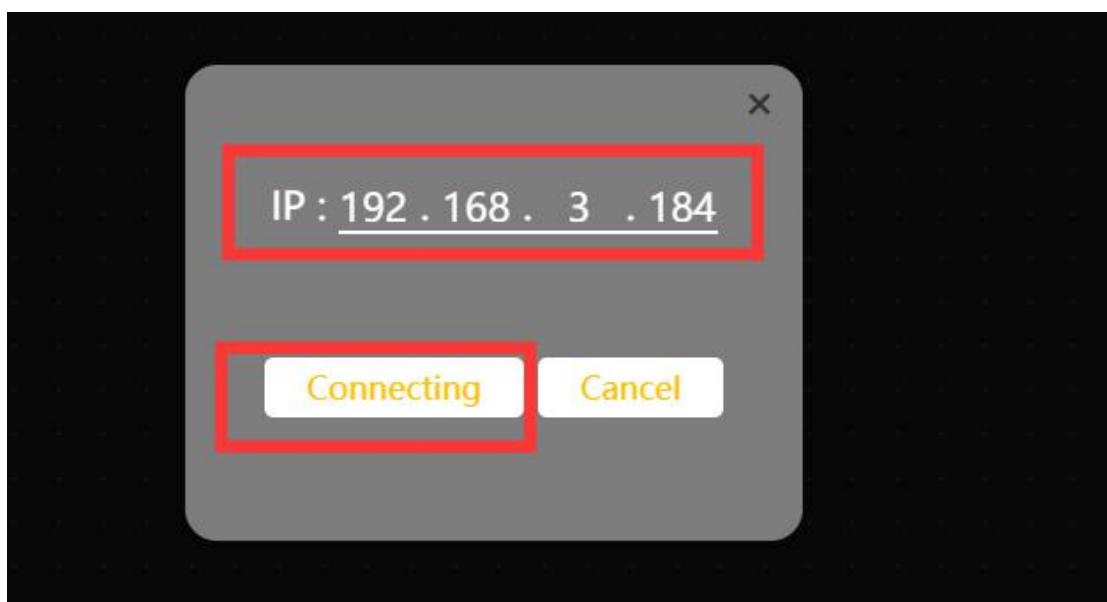
After successfully entering the website, the interface is as follows:



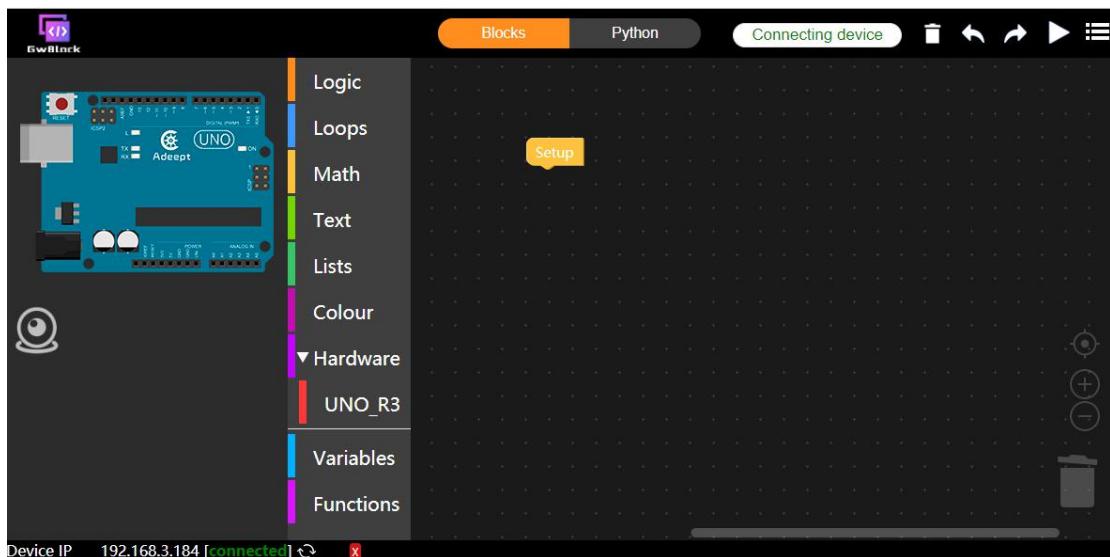
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

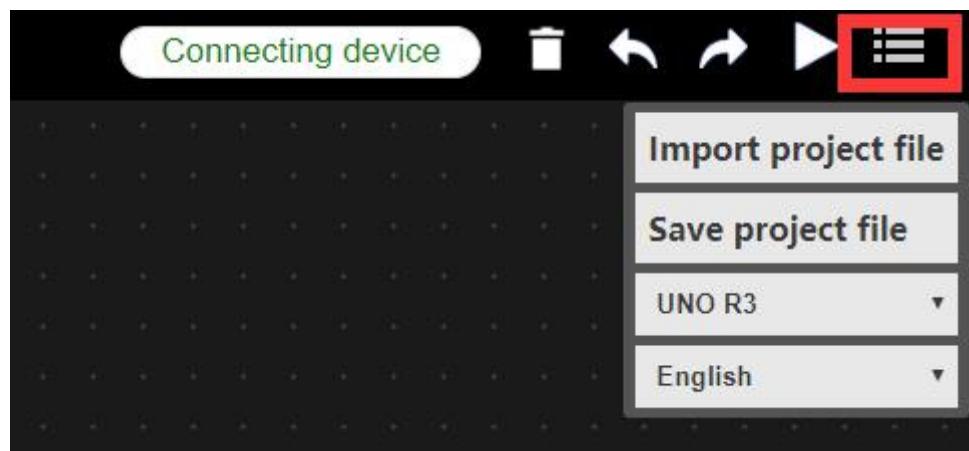


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

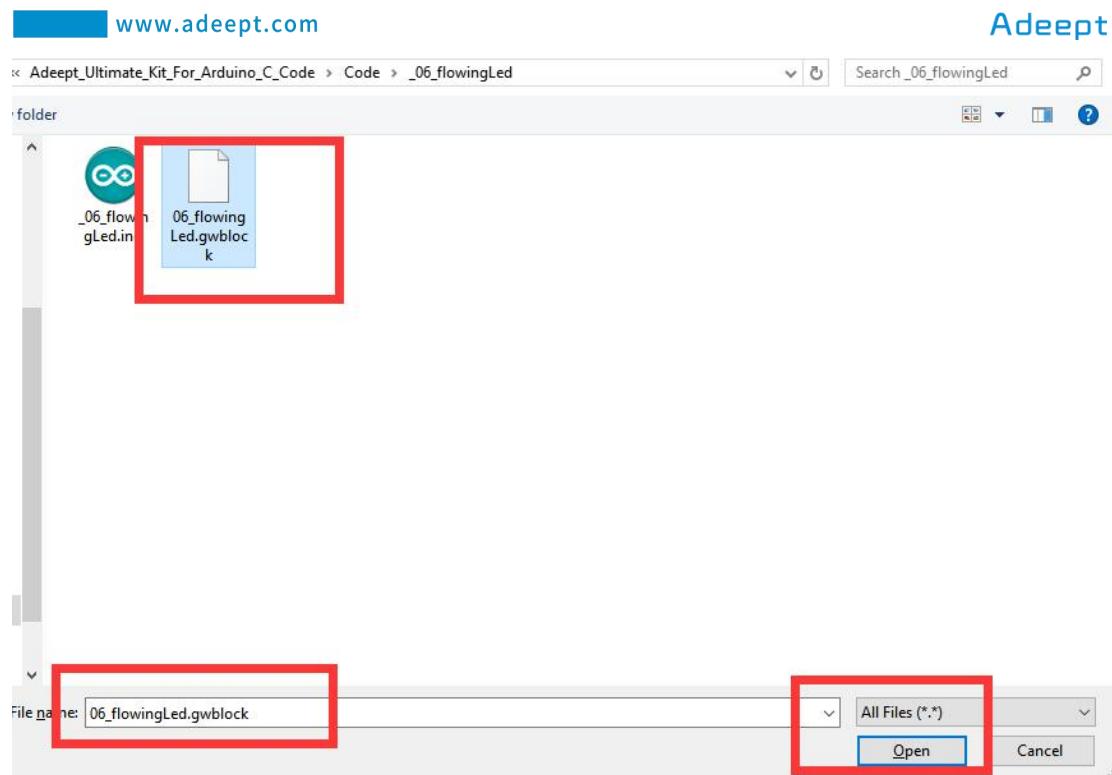
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_06_flowinLed

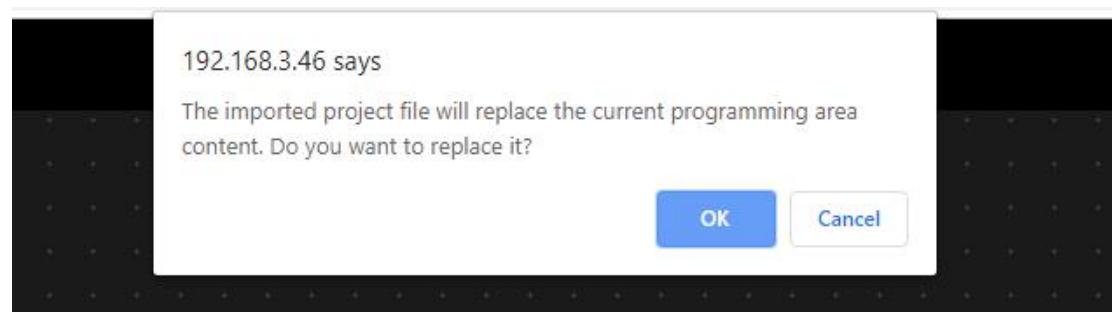
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

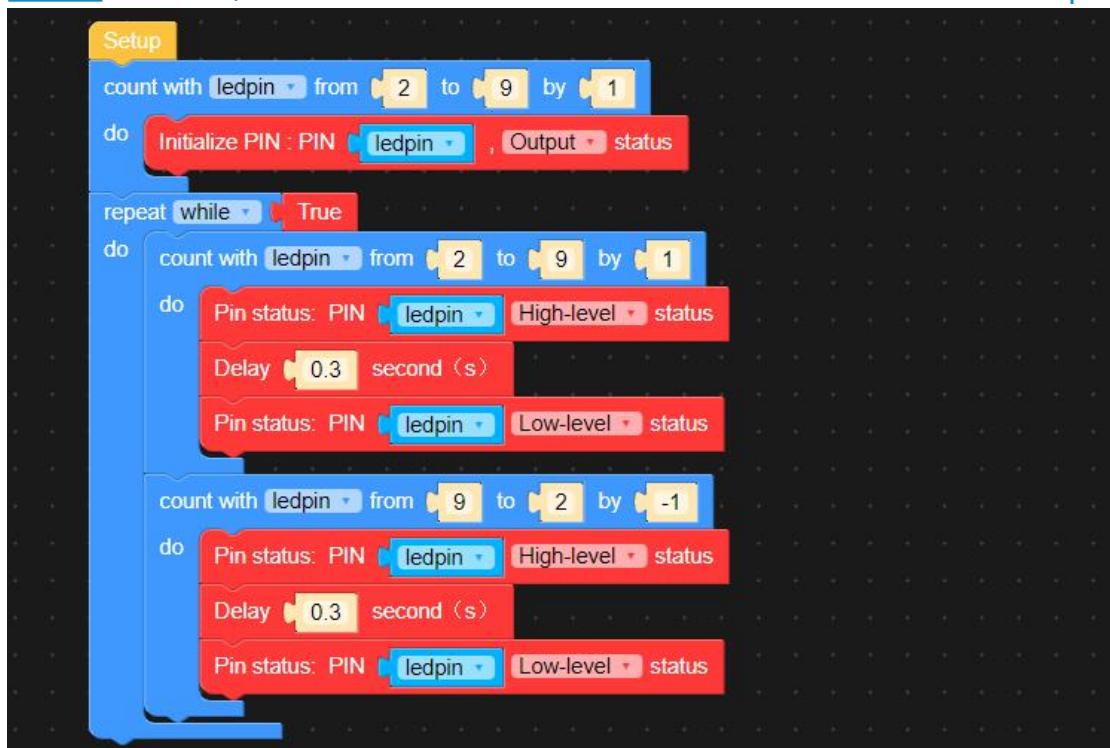
5. Select the "06_flowinLed.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



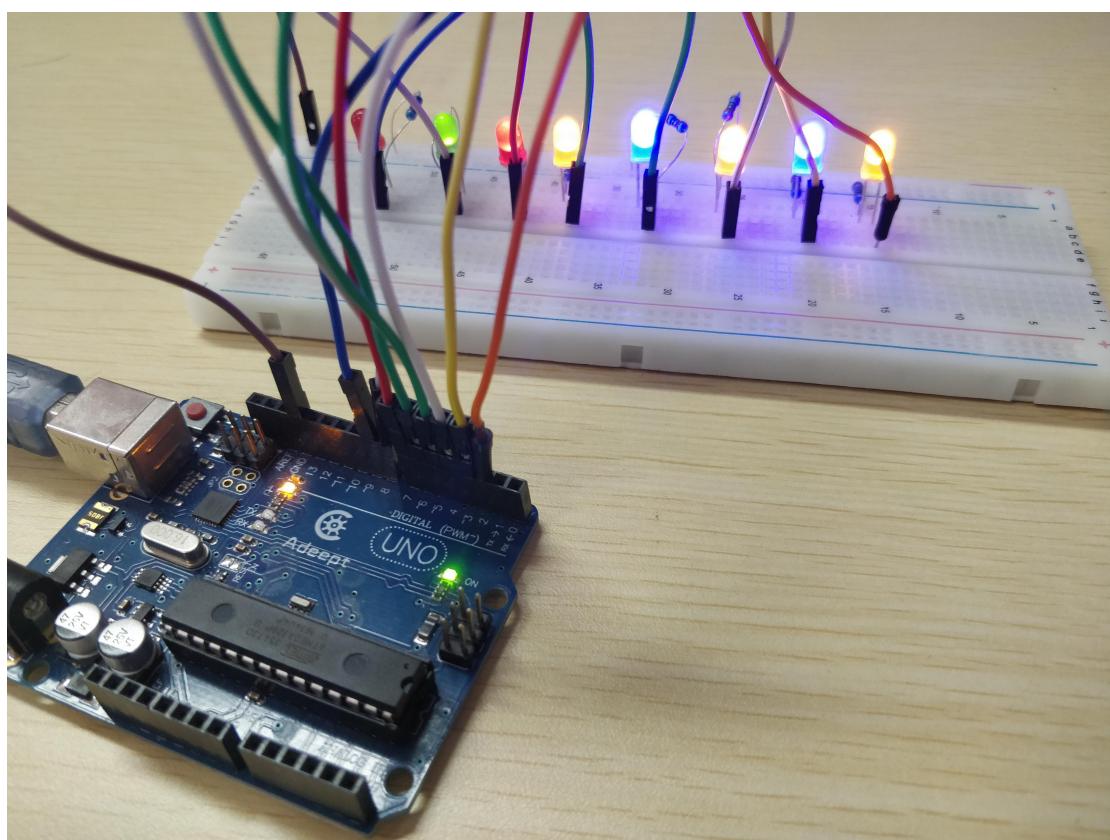
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



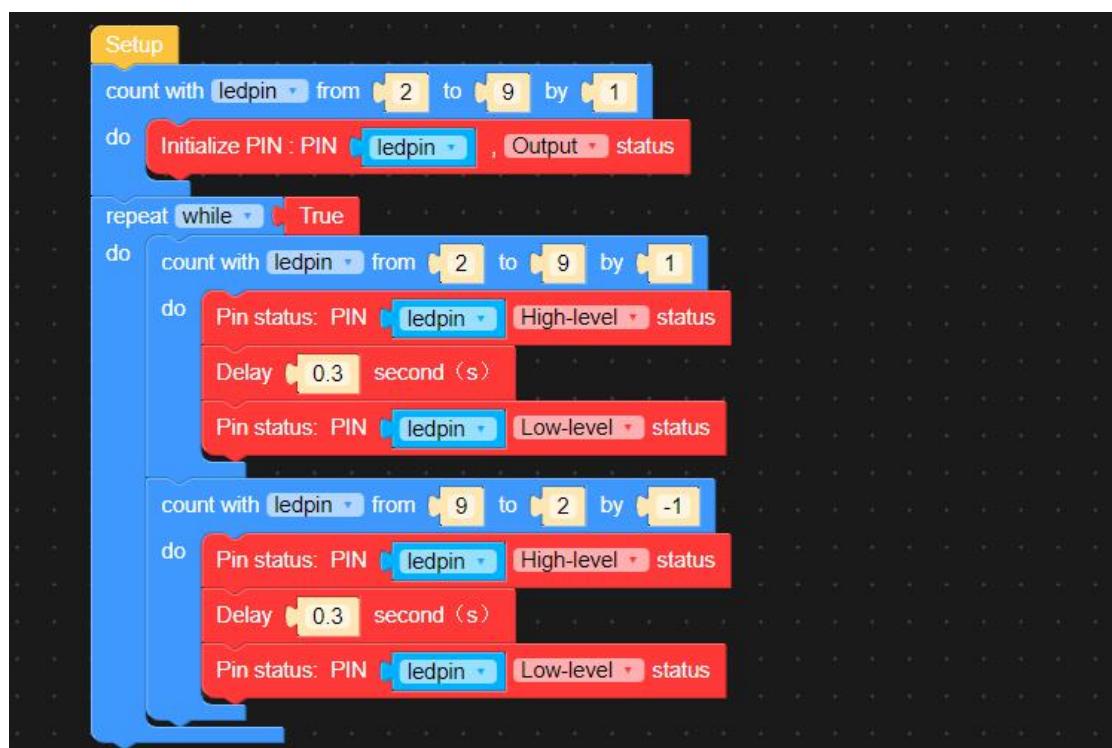
8. After running the program, it can be observed that the LED will be on and off from right to left, then from left to right, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



(4) Core code program

After the above hands-on operation, you must be very interested to know how we program to make Flowing LED on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

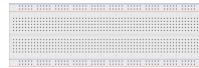
In the GwBlock graphical editor, all code programs are executed from **Setup**. First initialize the pin number of ledpin to 2,3,4,5,6,7,8,9 through the instruction block **count with ledpin from 2 to 9 by 1**, and the ledpin is set to the Output mode through the instruction block **Initialize PIN : PIN ledpin , Output status**; In the while loop statement, set the ledpin to High-level in order of pin number sequence through the instruction block **Pin status: PIN ledpin High-level status**, and the LED will be on. Delay 300ms of time through **Delay 0.3 second (s)**. Turn off the LED through the instruction block **Pin status: PIN ledpin Low-level status**; set the ledpin to High-level and Low-level according to the pin number sequence of 9 to 2, and the LED will be on and off in sequence. Then turn on and off the LED in sequence. Delay 300ms of time through **Delay 0.3 second (s)**.



Lesson 7 Controlling LED Bar with Potentiometer

In this lesson, we will learn how to control LED bar with Potentiometer.

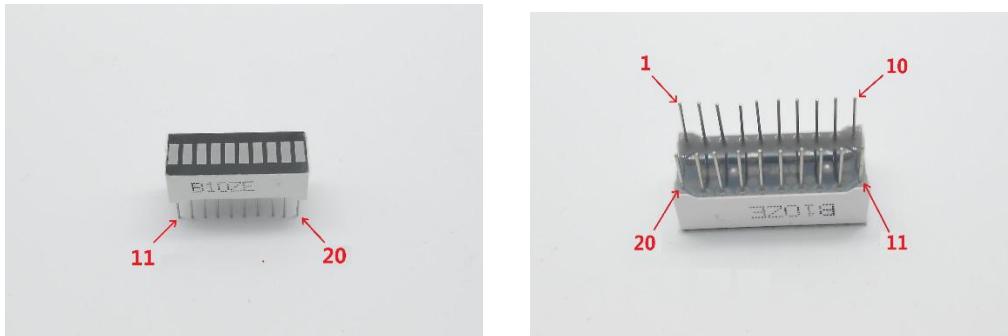
1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	10	
Potentiometer(10KΩ)	1	
LED Bar Graph	1	

2. Introduction of LED Bar

(1)LED Bar

The bar LED module is composed of a 10-segment bar LED. Each LED has two pins. One side is a positive pole and the other side is a negative pole. There is a total of twenty pins. Ten LED can be controlled respectively. LED Bar displays are widely used, such as audio equipment, dashboards and digital readout displays.



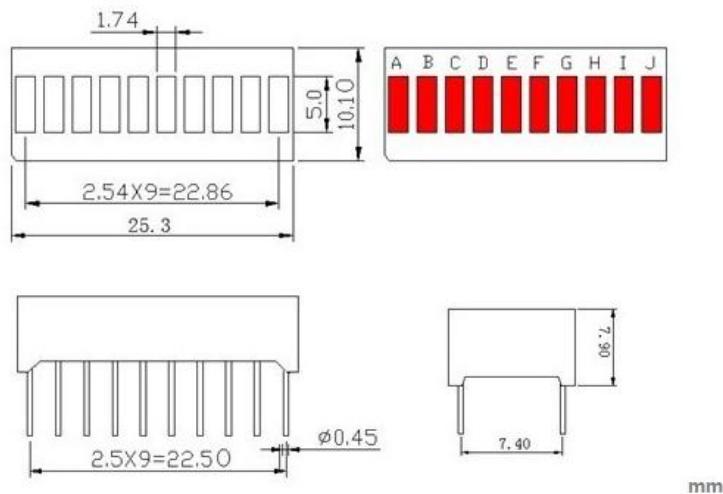
(2) Experimental principle of controlling LED Bar

A series of LED in a row. They can be applied to any series of digital outputs.

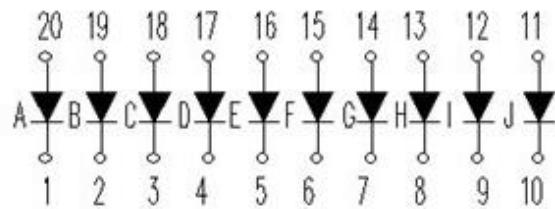
This tutorial borrows from the For Loop and Arrays tutorial as well as the Analog Input tutorial.

The sketch works like this: first read the input. Map the input to the output range which is 0-10 in this case since ten LED are used. Then you set up a **for** loop to iterate over the outputs. If the number in the array of the output is lower than the mapped input minimum, it is on. If not, it's off.





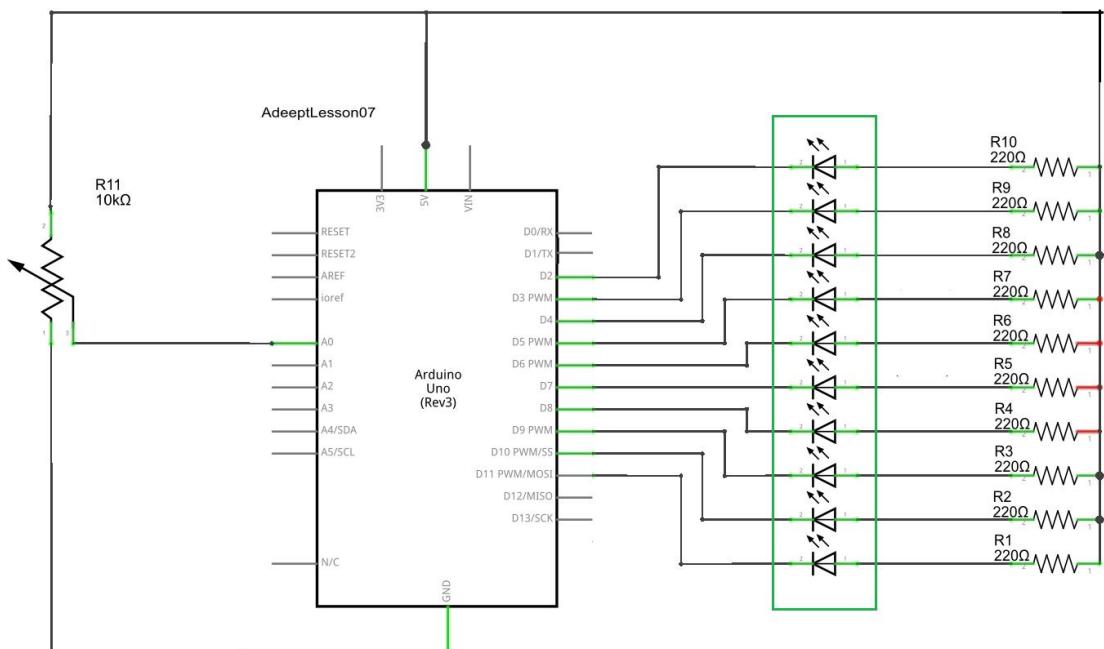
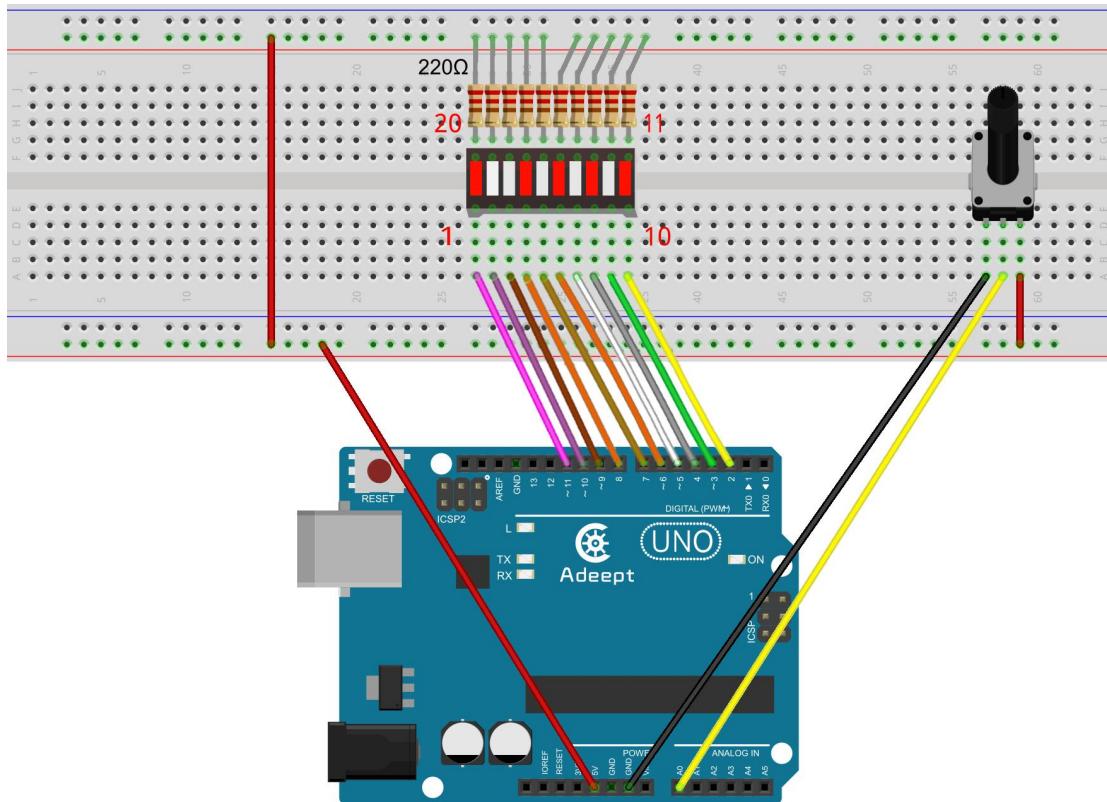
The internal schematic diagram for the LED bar graph is as shown below:



A potentiometer, informally a pot, is a three-terminal resistor with a sliding or rotating contact that forms an adjustable voltage divider. If only two terminals are used, one end and the wiper, it acts as a variable resistor or rheostat.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connected them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. Use $220\ \Omega$ resistor. As shown in the following figure:



4. Controlling LED Bar with Potentiometer

We provide two different methods to control the active buzzer. One is to program the active buzzer on the Arduino UNO with C language through the Arduino IDE. You

need to master the C language; the other is to program the active buzzer on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

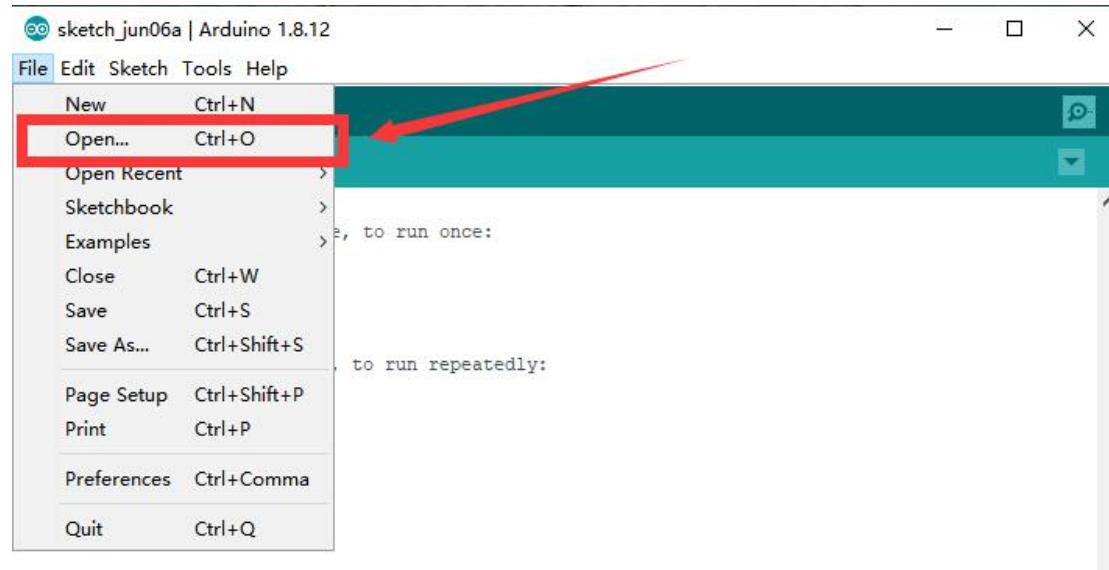
1.Using C language to program Potentiometer to control LED Bar on Arduino UNO

(1)Compile and run the code program of this course

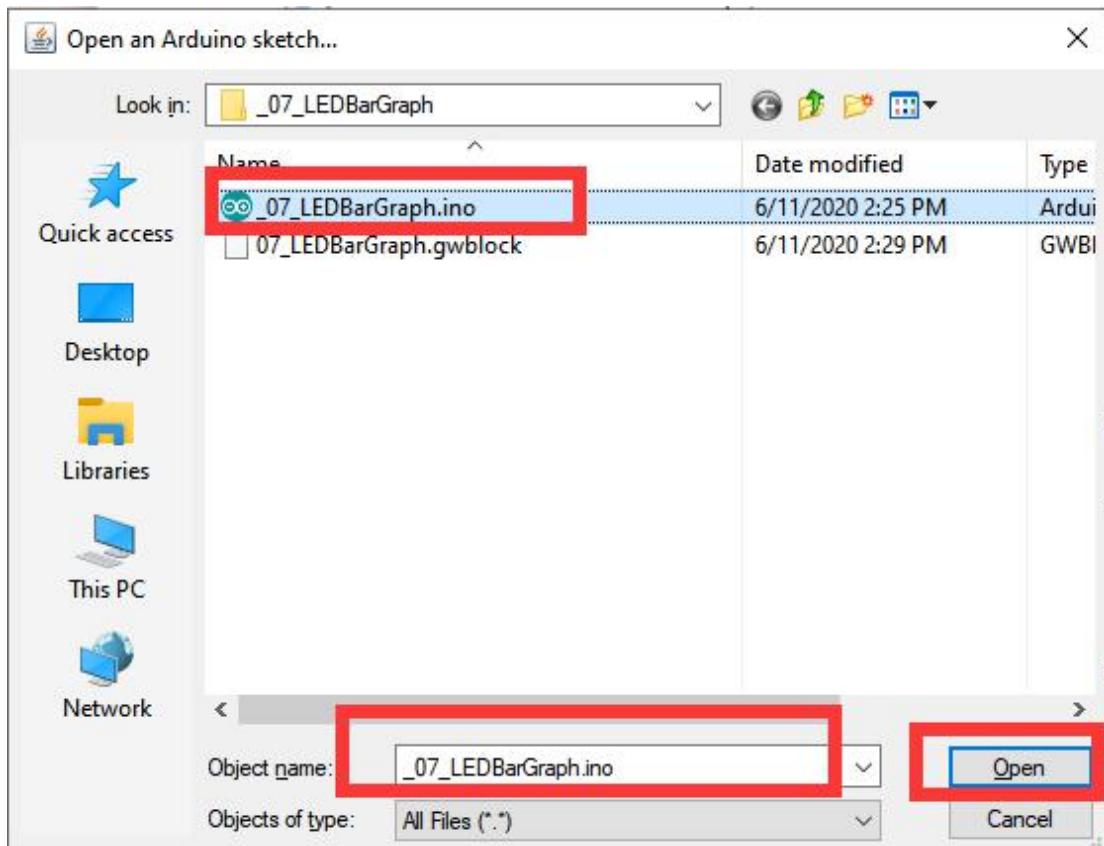
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\07_LEDBarGraph directory. Select _07_LEDBarGraph.ino. This file is the code program we need in this course. Then click Open.



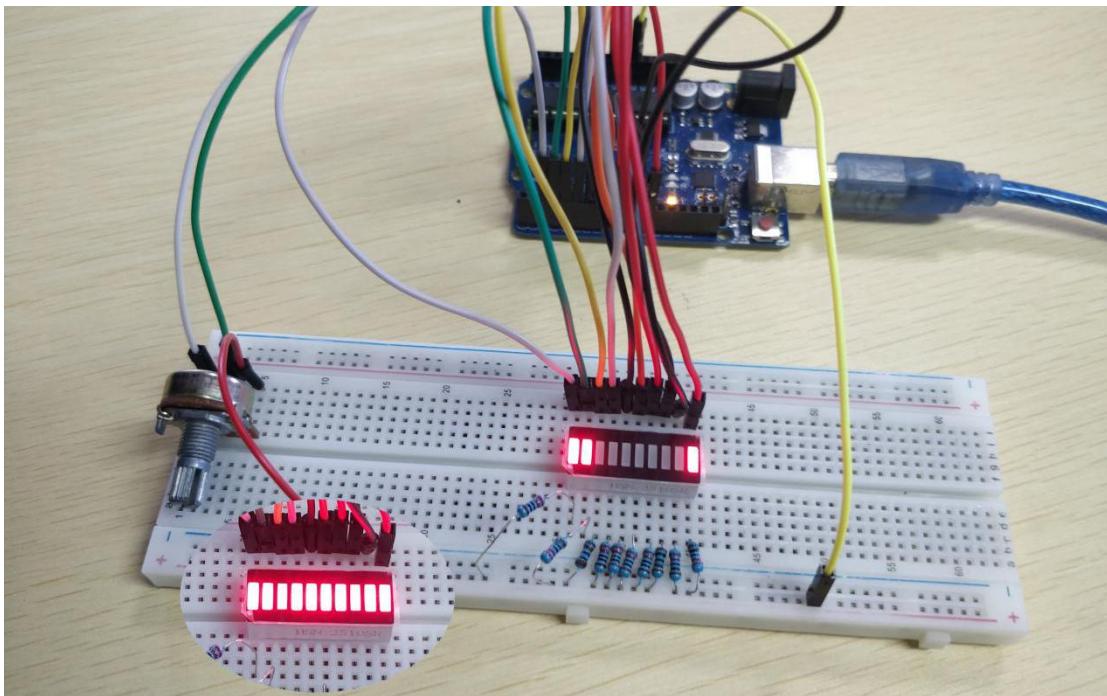
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After running the program, turn the potentiometer counterclockwise to light up each LED of LED bar; turn the potentiometer clockwise to turn off each LED of LED bar. This shows that our experimental test is successful. The physical connection diagram of the experiment is as below:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program Potentiometer to control LED Bar on Arduino UNO. We will introduce how our core code can be achieved:

3. Set the pin number of pin1 to pin10 to Output mode through pinMode(pin, OUTPUT) in the setup() method;

```
void setup()
{
    pinMode(pin1,OUTPUT); //Set the digital 1 port mode, OUTPUT: Output mode
    pinMode(pin2,OUTPUT); //Set the digital 2 port mode, OUTPUT: Output mode
    pinMode(pin3,OUTPUT); //Set the digital 3 port mode, OUTPUT: Output mode
    pinMode(pin4,OUTPUT); //Set the digital 4 port mode, OUTPUT: Output mode
    pinMode(pin5,OUTPUT); //Set the digital 5 port mode, OUTPUT: Output mode
    pinMode(pin6,OUTPUT); //Set the digital 6 port mode, OUTPUT: Output mode
    pinMode(pin7,OUTPUT); //Set the digital 7 port mode, OUTPUT: Output mode
    pinMode(pin8,OUTPUT); //Set the digital 8 port mode, OUTPUT: Output mode
    pinMode(pin9,OUTPUT); //Set the digital 9 port mode, OUTPUT: Output mode
    pinMode(pin10,OUTPUT); //Set the digital 10 port mode, OUTPUT: Output mode
}
```

2. In the loop() method, read the voltage value input by the potentiometer by using the analogRead() function. Then turn off each LED of LED bar in turn through digitalWrite(i, LOW) in the for loop statement; digitalWrite(j, HIGH) is the LED in the bar LED module.

```

void loop()
{
    float a = analogRead(potentiometerPin); //Read the voltage photoresistance
    a = map(a,0,1023,0,11); //Photoresistor voltage value converted from 0-1023 to 0-11
    for(int i=1;i<=a;i++){
        digitalWrite(i,LOW); //LOW is set to about 5V PIN8
    }
    for(int j=10;j>a;j--){
        digitalWrite(j,HIGH); //HIGH is set to about 5V PIN8
    }
    delay(50); //delay 50ms
}

```

2. Using graphical code blocks to program to control LED Bar with Potentiometer on Arduino UNO

(1) Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

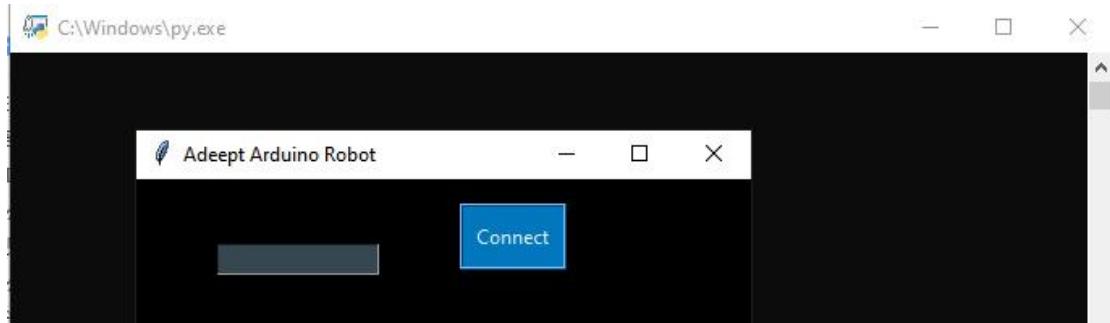
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

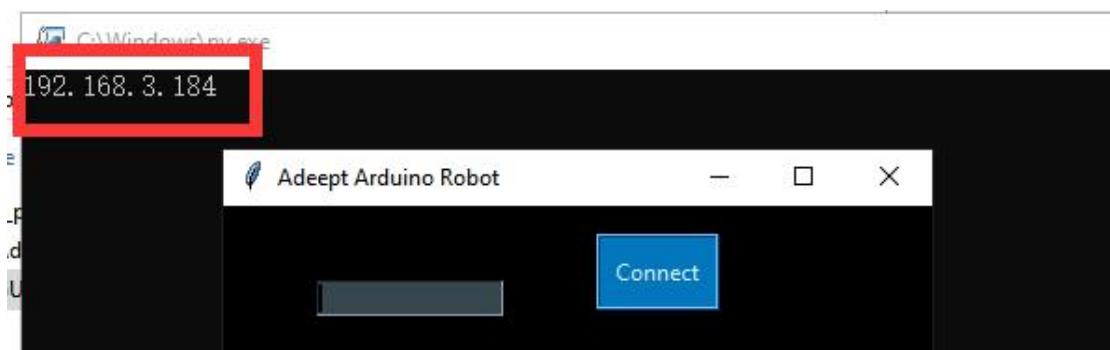
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

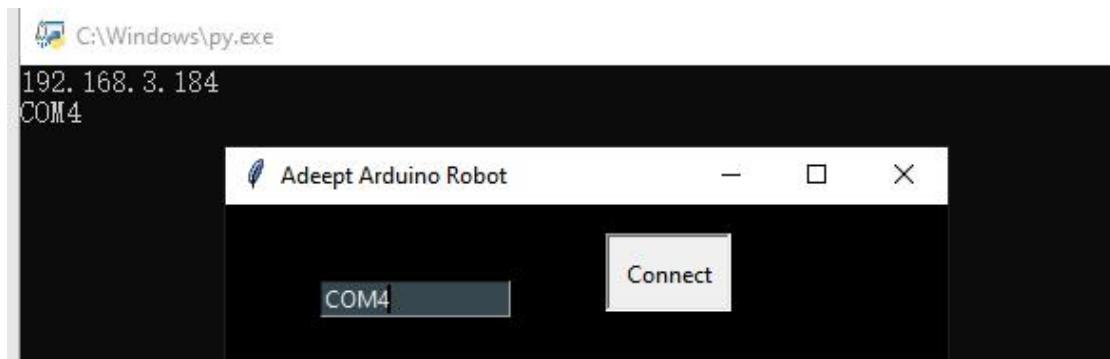
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



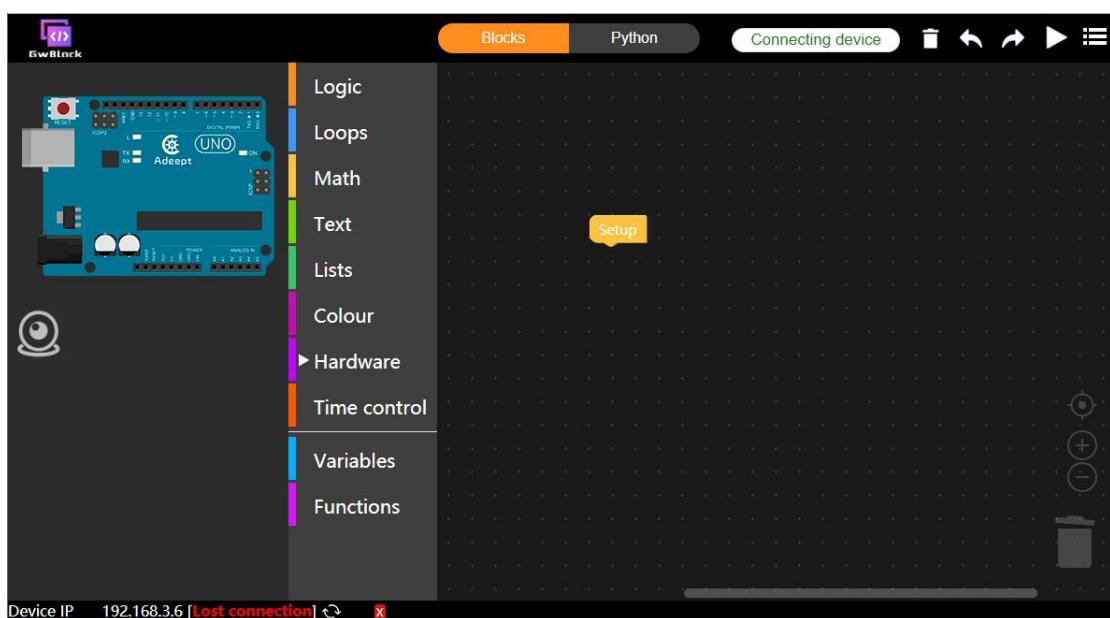
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



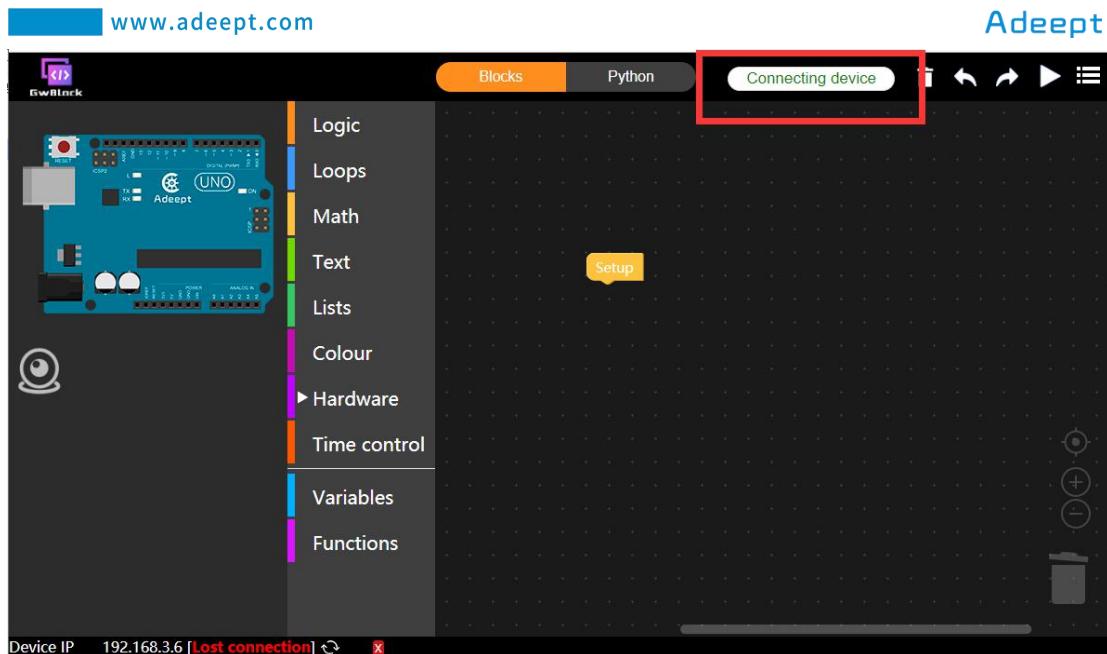
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

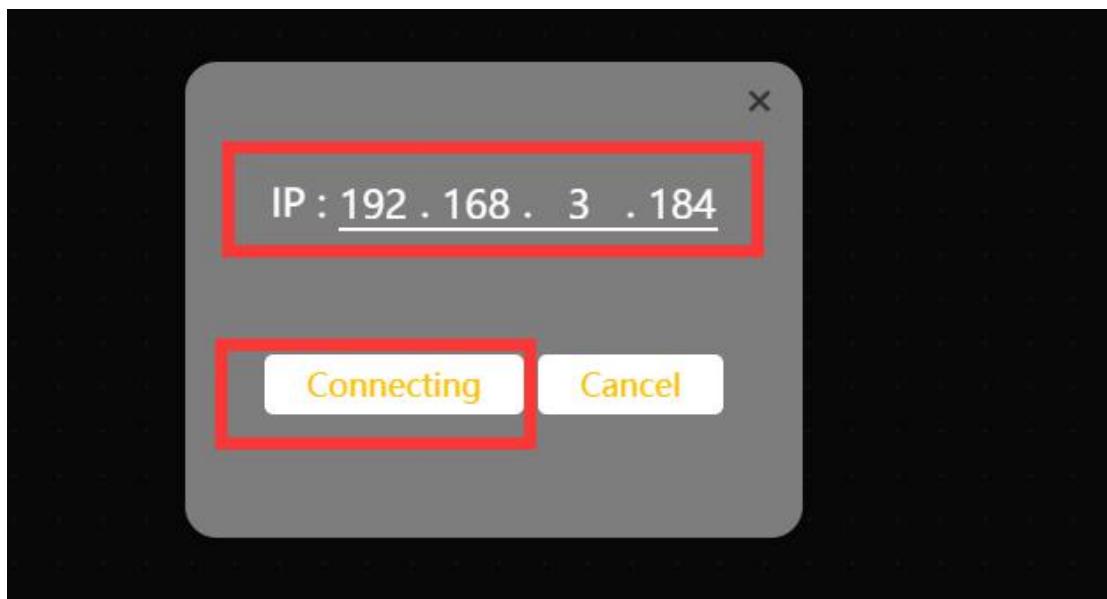
After successfully entering the website, the interface is as follows:



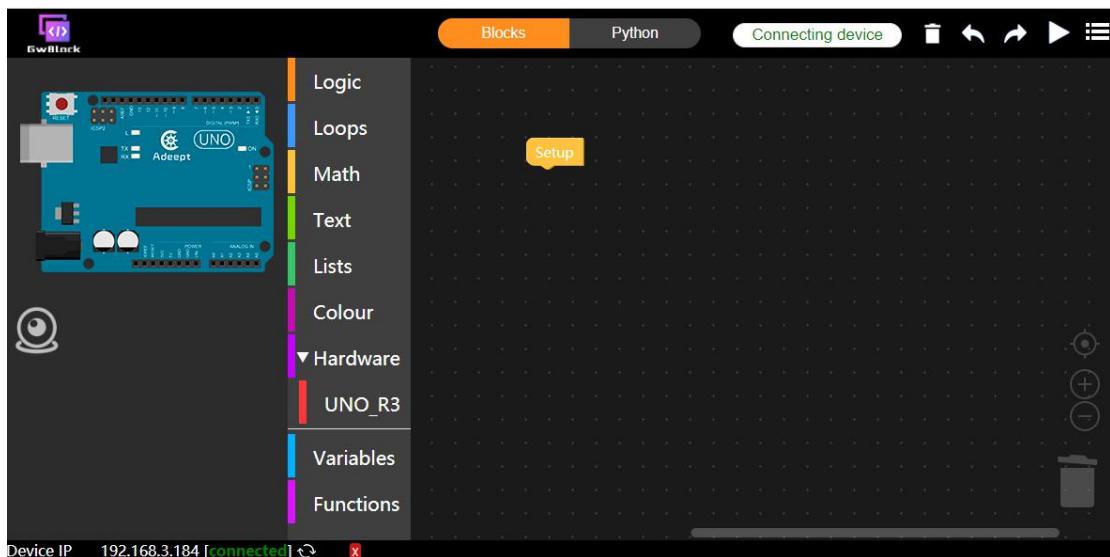
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

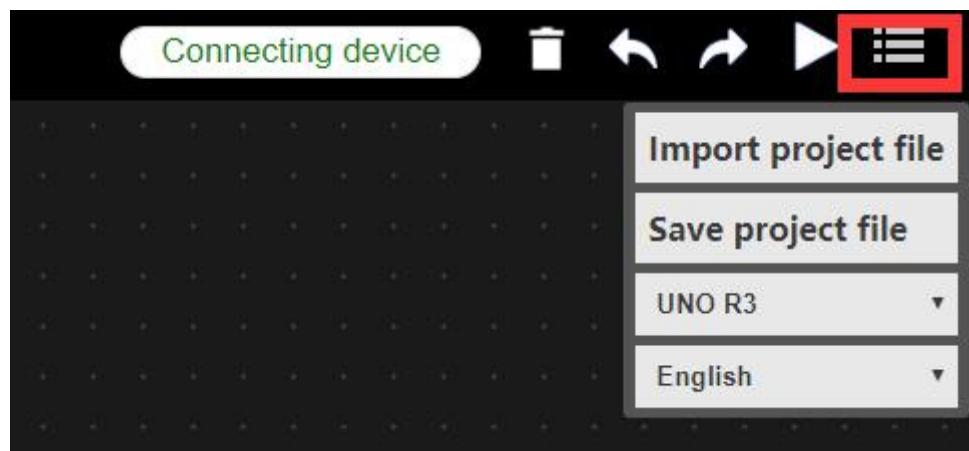


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

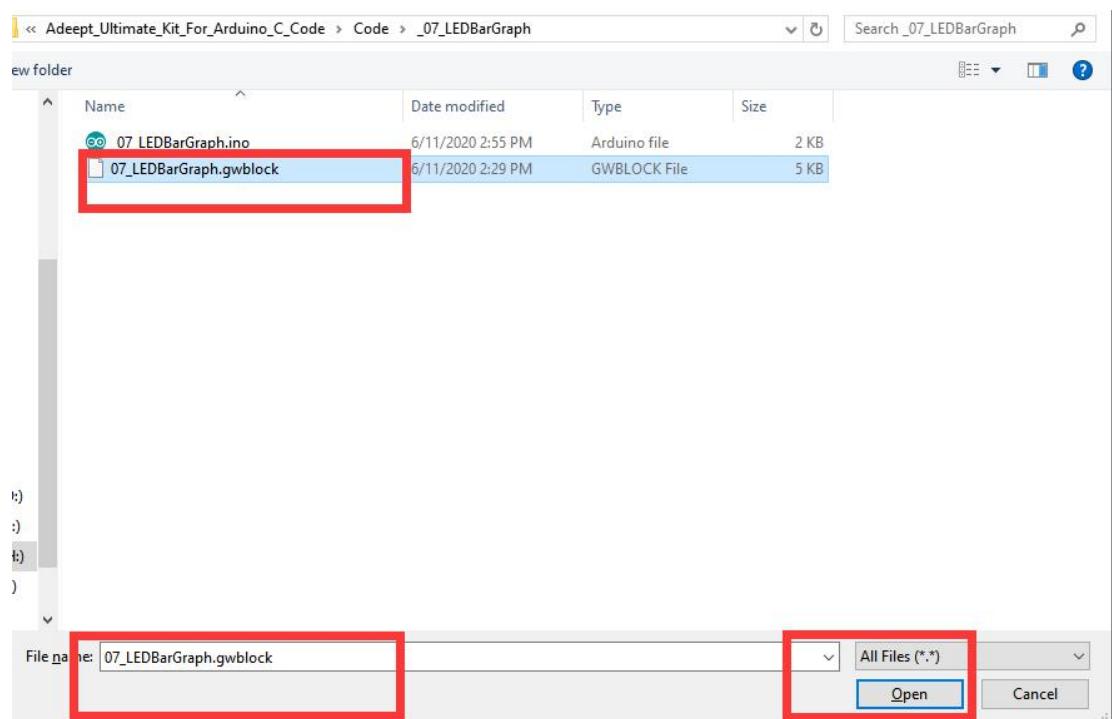
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code\07_LEDBarGraph

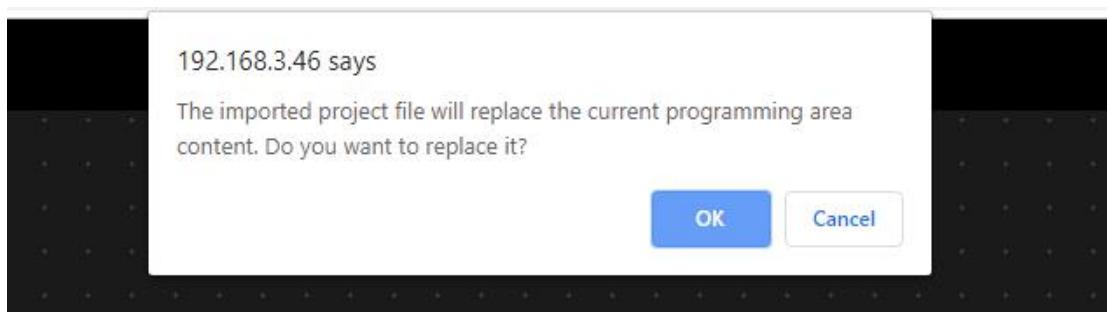
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

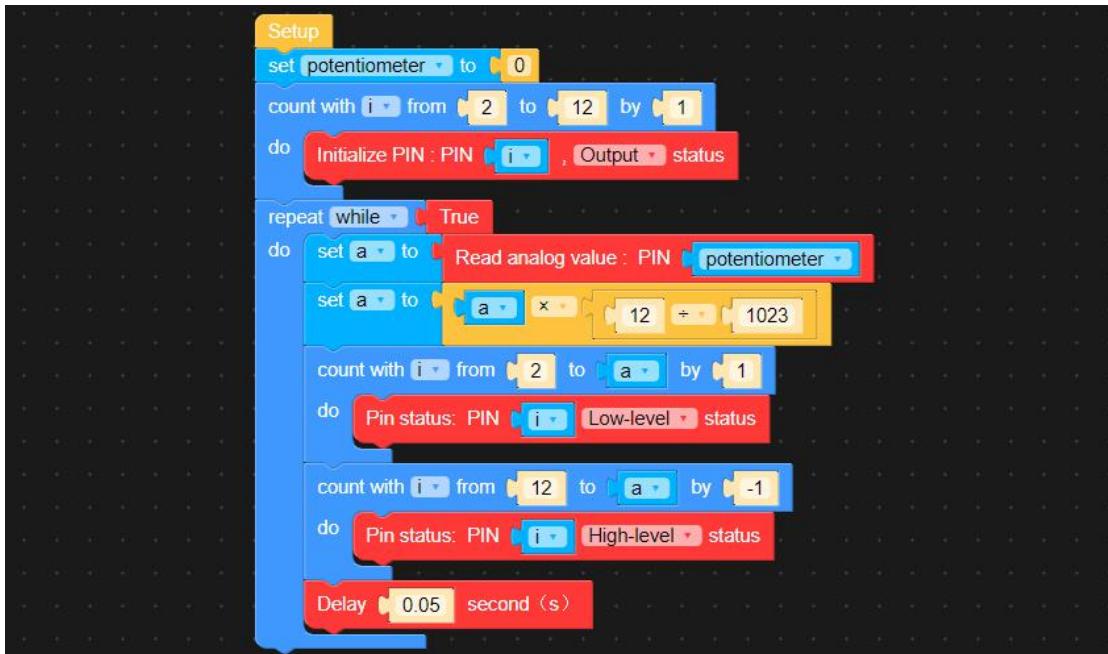
5. Select the "07_LEDBarGraph.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



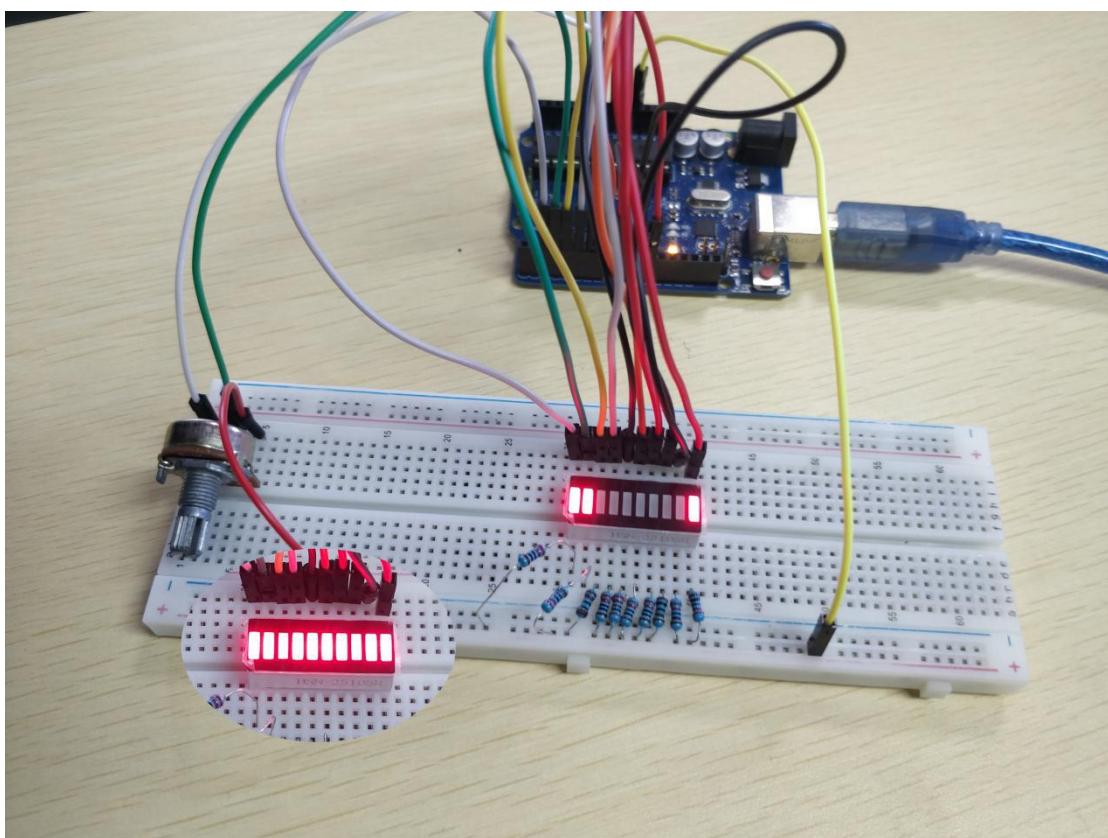
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



8. After running the program, turn the potentiometer counterclockwise to light up the LED; turn the potentiometer clockwise to turn off the LED. This shows that our experimental test is successful. The physical connection diagram of the experiment is as below:



(5) Core code program

After the above hands-on operation, you must be very interested to know how we control LED bar with Potentiometer by programming on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

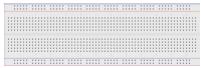
In the GwBlock graphical editor, all code programs are executed from **Setup**. First initialize the pin number from 2 to 11 through the instruction block **count with i from 2 to 12 by 1**, and the pin numbers are set to the Output mode through the instruction block **Initialize PIN : PIN i, Output status**. In the while loop statement, set the pin number from 2 to 11 to Low-level through the instruction block **Pin status: PIN i Low-level status**, and the LED of LED bar will be off; set the pin number from 2 to 11 to High-level through the instruction block **Pin status: PIN i High-level status**, and the LED of LED bar will be on. Delay 0.05s of time through **Delay 0.05 second (s)**.



Lesson 8 Making Breathing LED with LED

In this lesson, we will learn how to make Breathing LED with LED.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	1	
LED	1	

2. The introduction of the Breathing LED

(1)Breathing LED

Breathing LED refers to the LED whose light gradually changes from bright to dark on the Arduino. It feels like a person who is breathing. It is widely used on mobile phones and serves as a reminder.



(2)The working principle of Breathing LED

(1)The working principle of the breathing light

How does our breathing light achieve a similar lighting effect? The breath light controls the brightness of the LED by controlling the frequency (time) of the light flashing, and this process is repeated over and over again to create the effect of "breathing".

In this lesson, the Arduino analog output function is used to control the brightness of the LED light through PWM pulse width modulation to present the breathing light effect.

Arduino generates a square wave with a certain duty cycle on the pin through PWM technology. Changing the duty cycle is equivalent to changing the equivalent voltage output by the pin, thereby causing the change of brightness of the LED.

In the circuit, the GND of the LED is connected to the GND on the Arduino UNO development board through a 220Ω current limiting resistor, and the VCC is connected to pin 11 of the GPIO on the Arduino UNO development board. The higher the duty cycle of the PWM signal output by the GPIO, the lighter the LED; otherwise, the lower the duty cycle, the darker the LED. We gradually increase the duty cycle of the PWM signal through programming, and then the brightness of the LED will

gradually increase; otherwise, the brightness of the LED will gradually decrease. The brightness of the LED gradually changes from dark to bright, and then changes from bright to dark, and alternately produces a respiration-like effect.

How to control the frequency (time) of LED flashing? It is through PWM (pulse width modulation) principle to achieve. We will introduce the principle of PWM as follow.

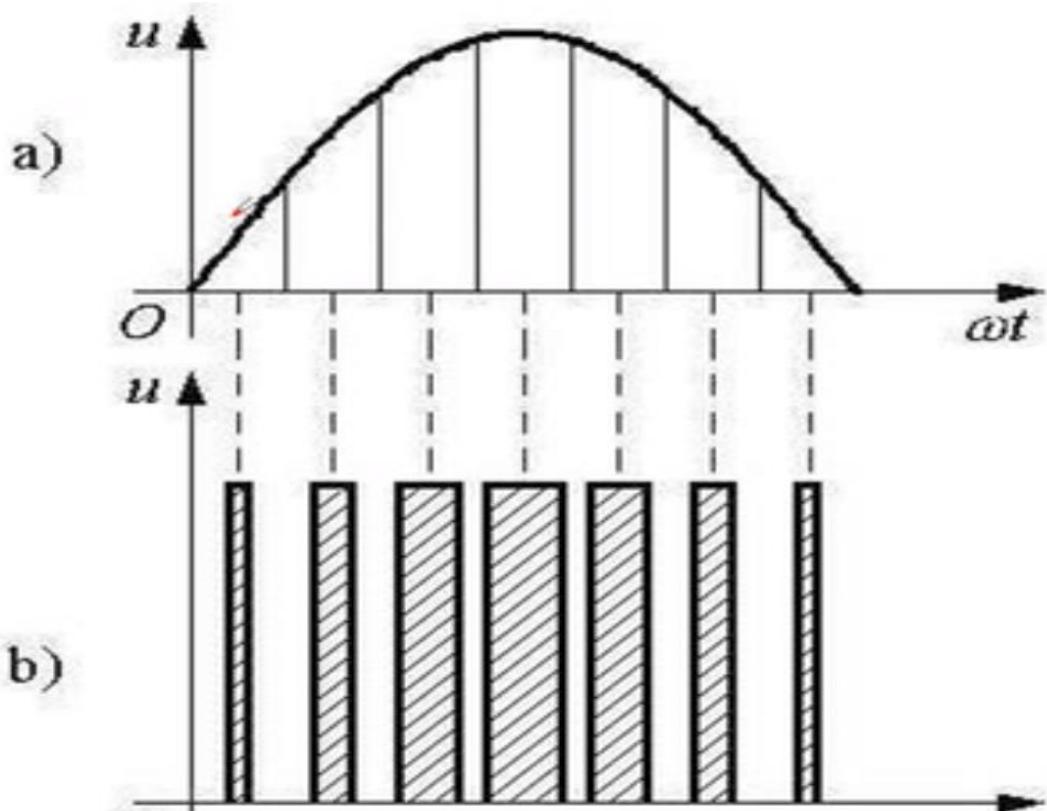
(2) Pulse width modulation (PWM)

1. Definition of pulse width modulation:

Pulse width modulation is an analog control method, which modulates the bias of the transistor base or MOS gate according to the change of the corresponding load to realize the change of the transistor or MOS tube conduction time, so as to realize the change of the output of the switching voltage stabilized power supply. This method can make the output voltage of the power supply remain constant when the working conditions change, and it is a very effective technique to control the analog circuit with the digital signal of the microprocessor. Pulse width modulation is a very effective technique for controlling analog circuits with the digital output of microprocessors. It is widely used in many fields, from measurement and communication to power control and transformation.

2. Principle of pulse width modulation

The control mode is to control the on-off of the inverter circuit switching device, so that the output end can get a series of pulses with the same amplitude, and use these pulses to replace the sine wave or the required waveform. In other words, multiple pulses are generated in the half period of the output waveform, so that the equivalent voltage of each pulse is a sinusoidal waveform, so that the output obtained is smooth and has fewer low-order harmonics. The width of each pulse can be modulated according to certain rules, which can not only change the output voltage of the inverter circuit, but also change the output frequency.



The above method may not be easy for you to understand, but you can understand it through the following sentence:

In the range of 0-1s: the lighting time of led lamp between 0-1ms is 0us; Light 1us in 1-2ms, light 2us in 2-3ms... Bright 999us in 999-1000ms.

3. Change duty cycle to realize breathing light:

The lighting and extinguishing of LED is the result of the level change, which can be regarded as a cycle. Each cycle will be shown as LED flicker. When the cycle is very short, that is, when the frequency is very high, this flicker will not be recognized by the naked eye, which will make people have the sense of continuous LED glow. In one cycle, the ratio between the duration of high level and the duration of one cycle is called duty cycle ratio. The higher the duty cycle is, the larger the current passing through the LED will be, and the brighter the visual perception will be. Now, you should have the train of thought that makes breathing light, namely change duty cycle! If the duty cycle is slightly stepped up, there will be a sense of brightening of LED. Otherwise it will get dark.

(3)Experimental Principle

Digital signals are signals whose levels are discontinuously changed by 0 and 1, while analog signals are used to represent information by continuously changing physical quantities. The signal changes continuously with time. Most of the exposures in our lives are analog signals, such as changes in sound, light, and temperature.

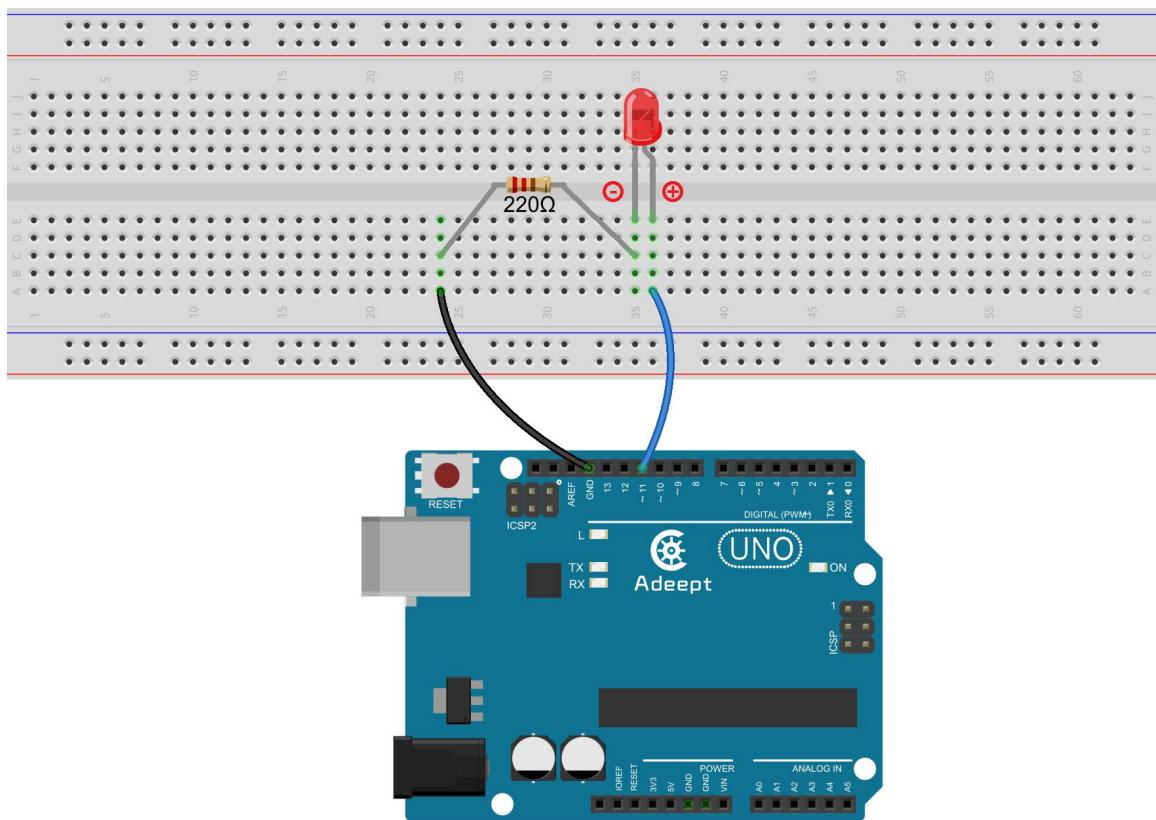
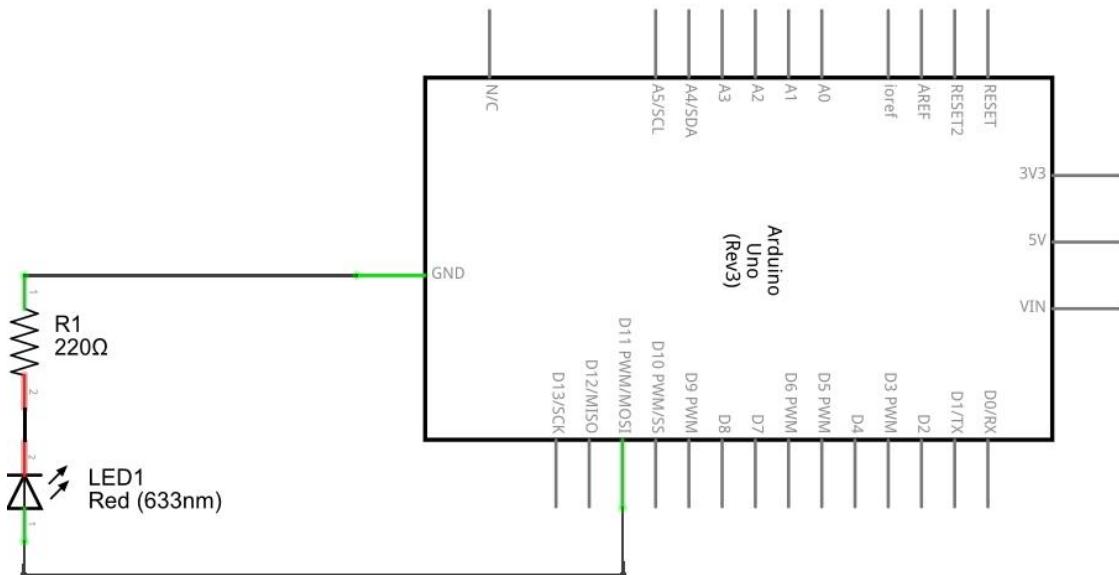
On our Arduino Uno R3 development board, pins 3, 5, 6, 9, 10, 11 have PWM (Pulse Width Modulation) function.

By using the `analogWrite()` function, a square wave with a fixed period is output through the continuous conversion of high and low levels on the specified pin, and by changing the proportion of high and low levels in each cycle (duty cycle). Then we get different voltage output.

When the high level appears longer in a cycle, the output voltage will be higher and the LED light will be brighter. When the high level time is shorter, the output voltage will be lower and the LED brightness will be darker.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connected them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes, as shown in the following figure:


AdeeptLesson08


4. Making Breathing LED with LED

We provide two different methods to control the active buzzer. One is to program the active buzzer on the Arduino UNO with C language through the Arduino IDE. You

need to master the C language; the other is to program the active buzzer on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

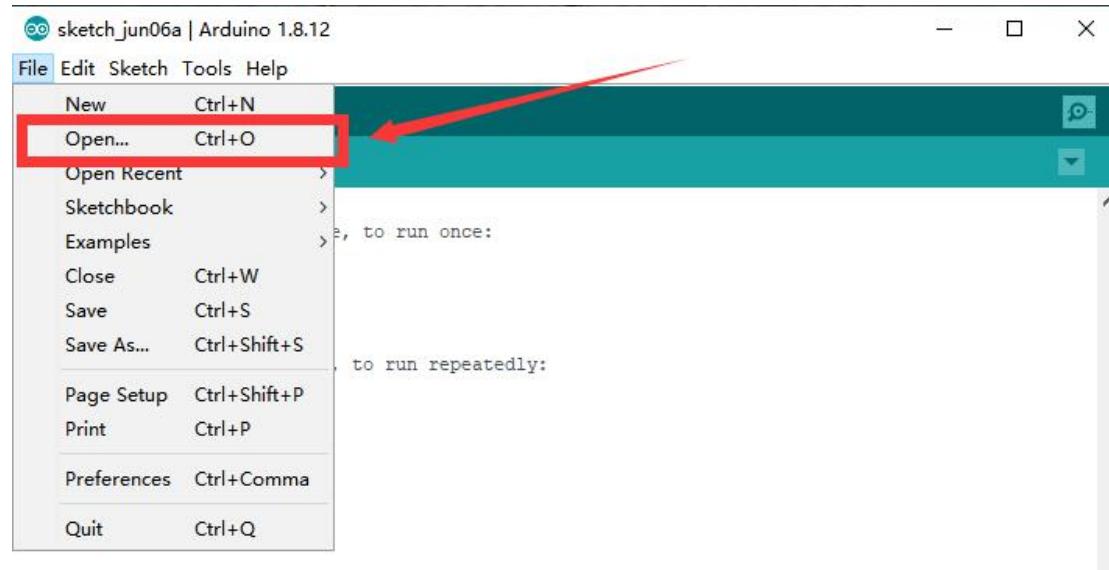
1. Using C language to program to make Breathing LED on Arduino UNO

(1) Compile and run the code program of this course

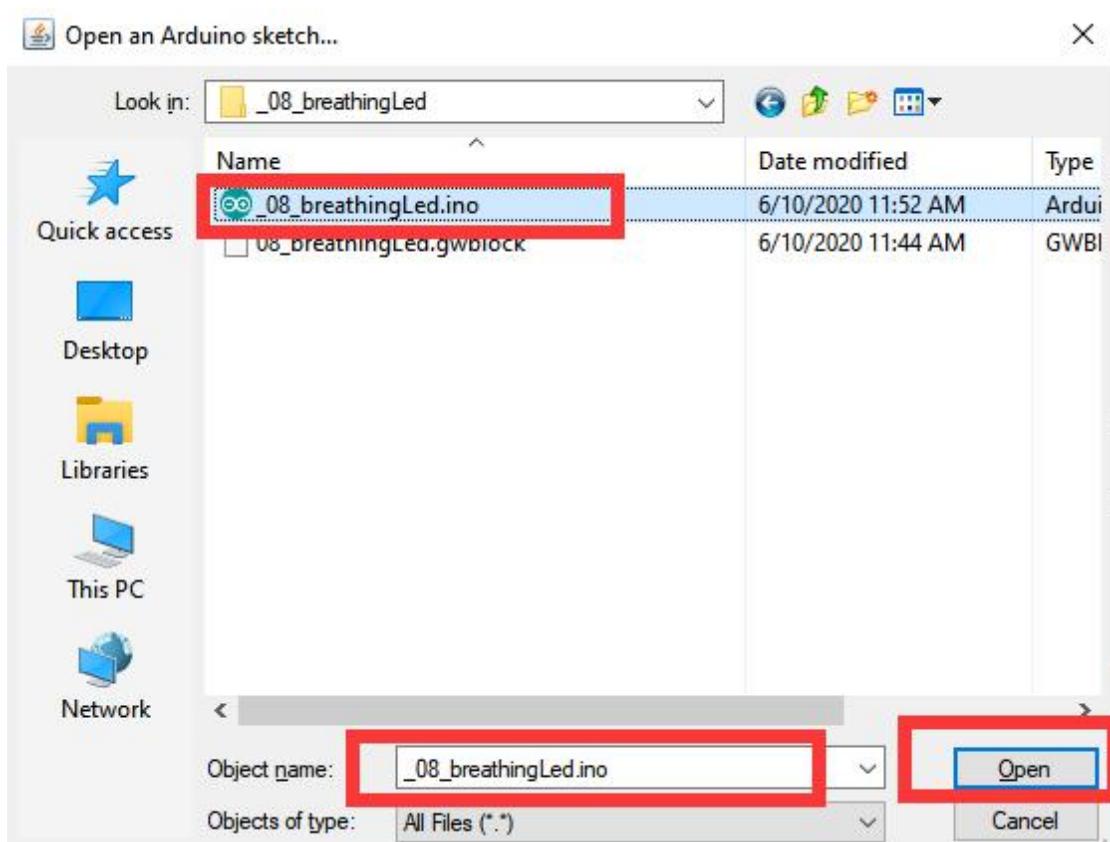
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\08_breathingLed directory. Select_08_breathingLed.ino. This file is the code program we need in this course. Then click Open.



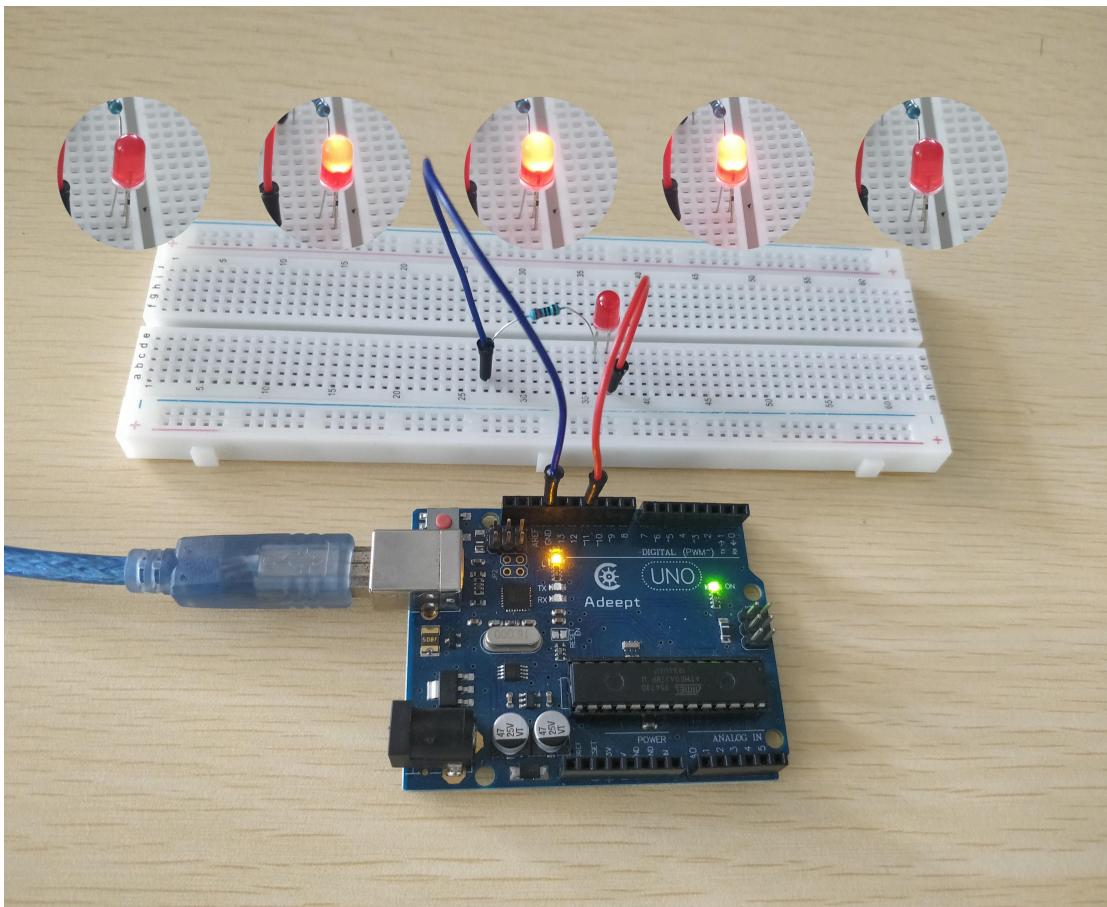
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After successfully running the program, we will observe that the LED changes slowly from the extinguished state to the increasingly brighter state, and then slowly changes from the brightest state to the increasingly darker state, until it turns off. Repeat this process non-stop to form the breathing light effect. This shows that our experimental test is successful. The physical connection diagram of the experiment is as below:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to make Breathing LED on Arduino UNO. We will introduce how our core code can be achieved:

1. Define pin 11 as the pin of the LED. Represent it to ledpin; in the setup() function, use pinMode(ledpin, OUTPUT) to set the ledpin to OUTPUT mode.

```
int ledpin=11; //definition digital 11 pins as pin to control the LED

void setup ()
{
    pinMode(ledpin,OUTPUT); //Set digital 11 port mode, the OUTPUT for the output
}
```

2. In the loop() function, we use the principle of PWM to control the brightness of LED to gradually increase. Then we use the principle of PWM to control the brightness of LED to gradually decrease. We control the state of the LED through the delay() function and repeat this process continuously to achieve the effect of breathing

to LED.

```

void loop()
{
    for (int a=0; a<=65;a++) //Loop, PWM control of LED brightness increase
    {
        analogWrite(ledpin,a); //PWM output value a (0~255)
        if(a<20)
            delay(70);
        else
            delay(10);
    }

    for (int a=65; a>=0;a--) //Loop, PWM control of LED brightness Reduced
    {
        analogWrite(ledpin,a); //PWM output value a (255~0)
        if(a<20)
            delay(70);
        delay(10);
    }
    delay(200); //200ms delay
}

```

2.Using graphical code blocks to program to make Breathing LED on Arduino UNO

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1.Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

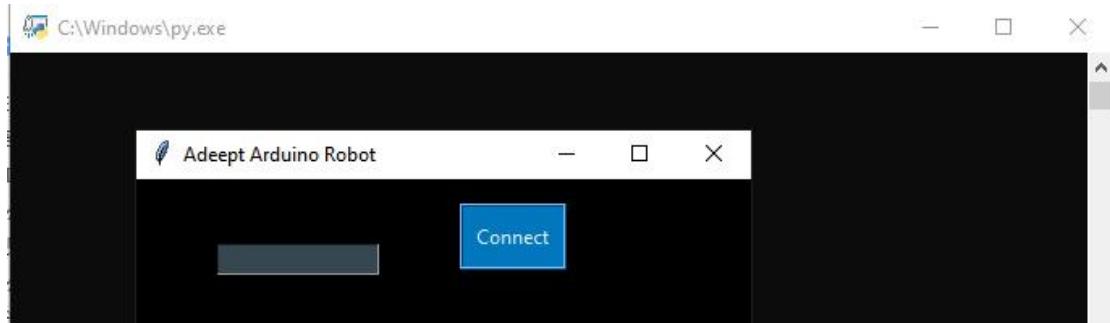
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

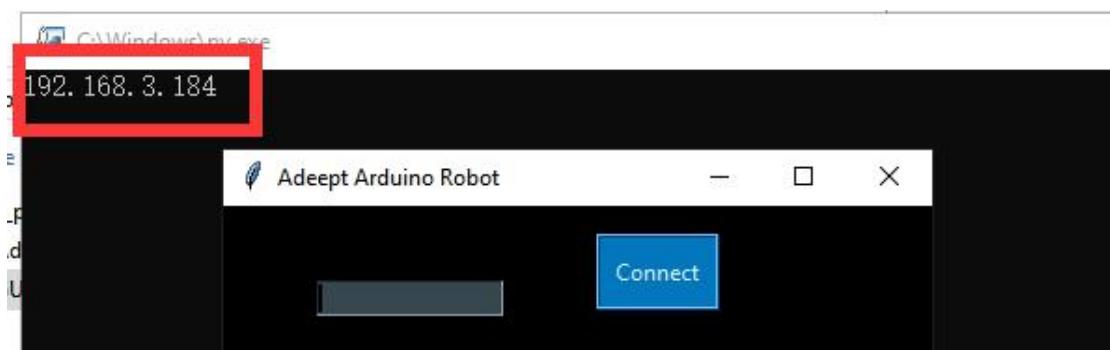
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

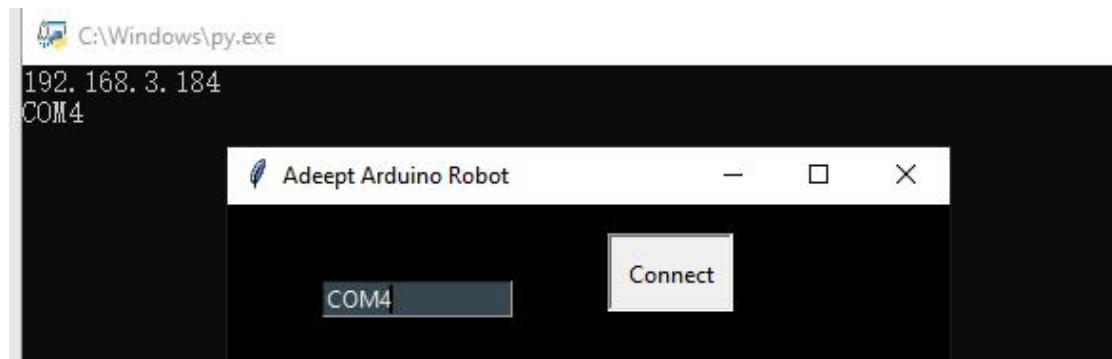
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



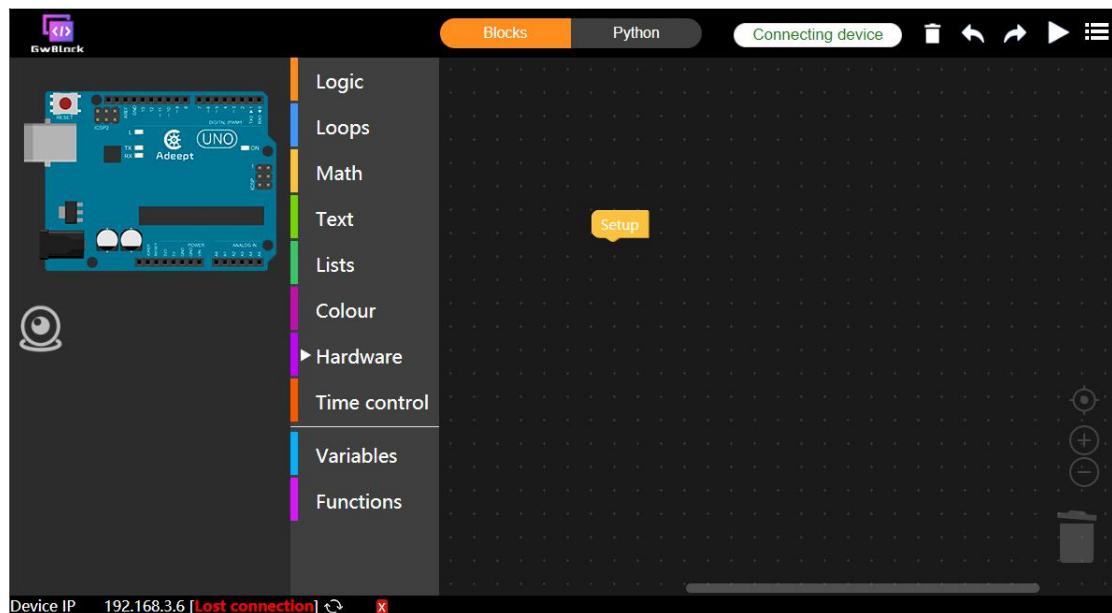
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



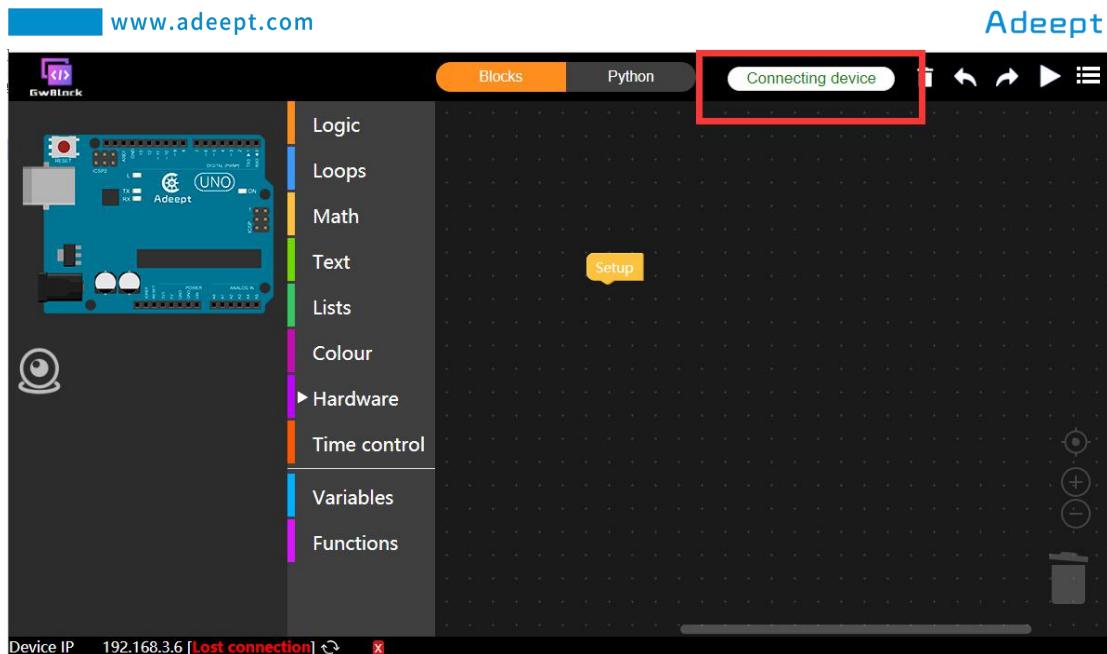
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

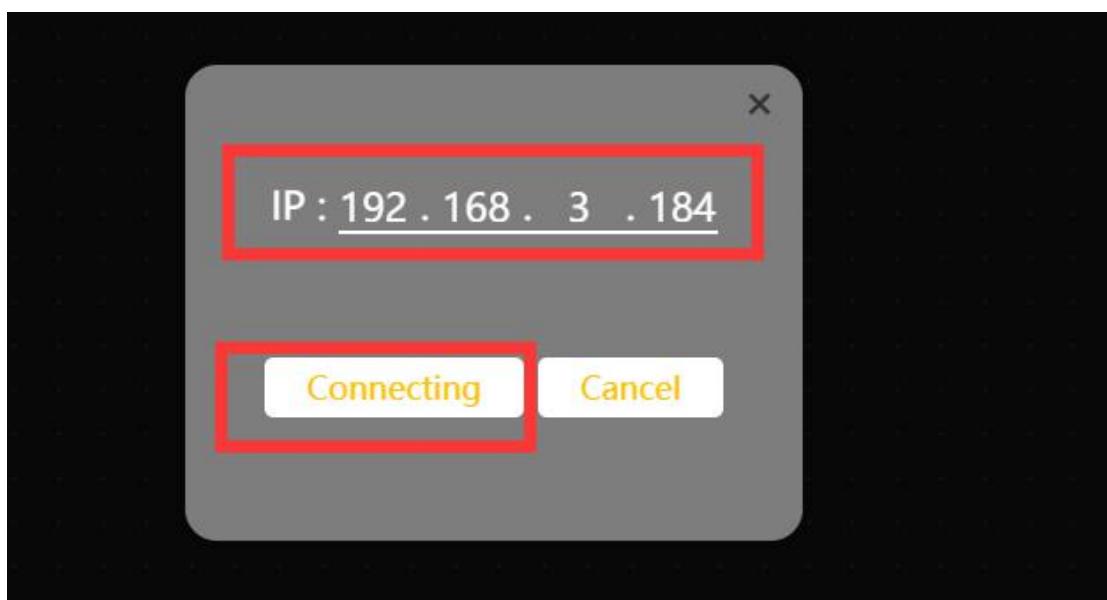
After successfully entering the website, the interface is as follows:



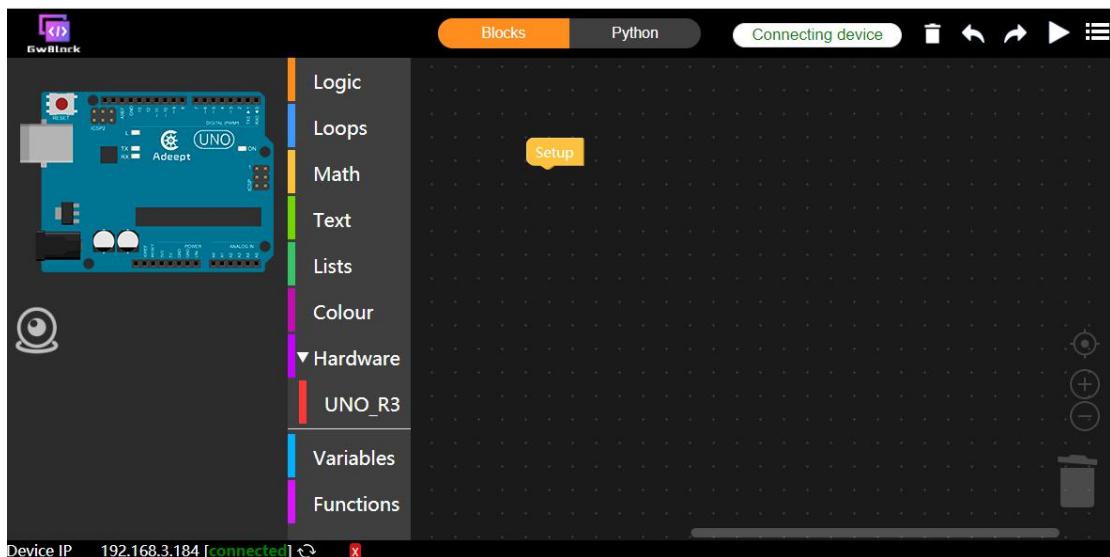
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:



(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3.The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

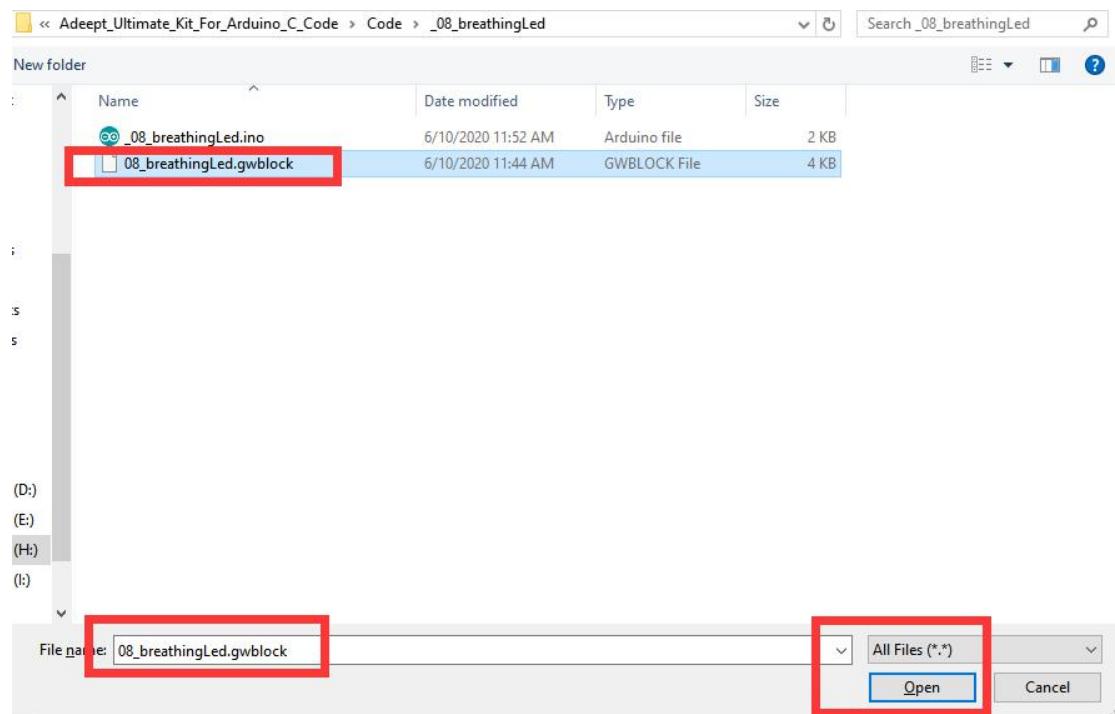
4.Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_08_breathingLed

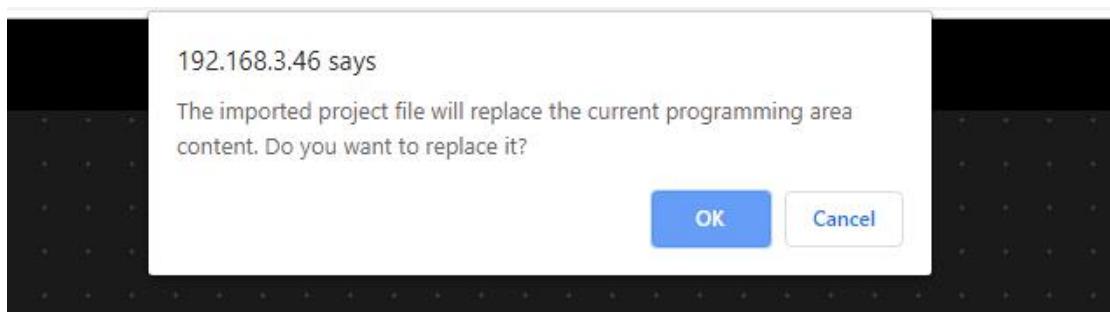
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

5. Select the "08_breathingLed.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



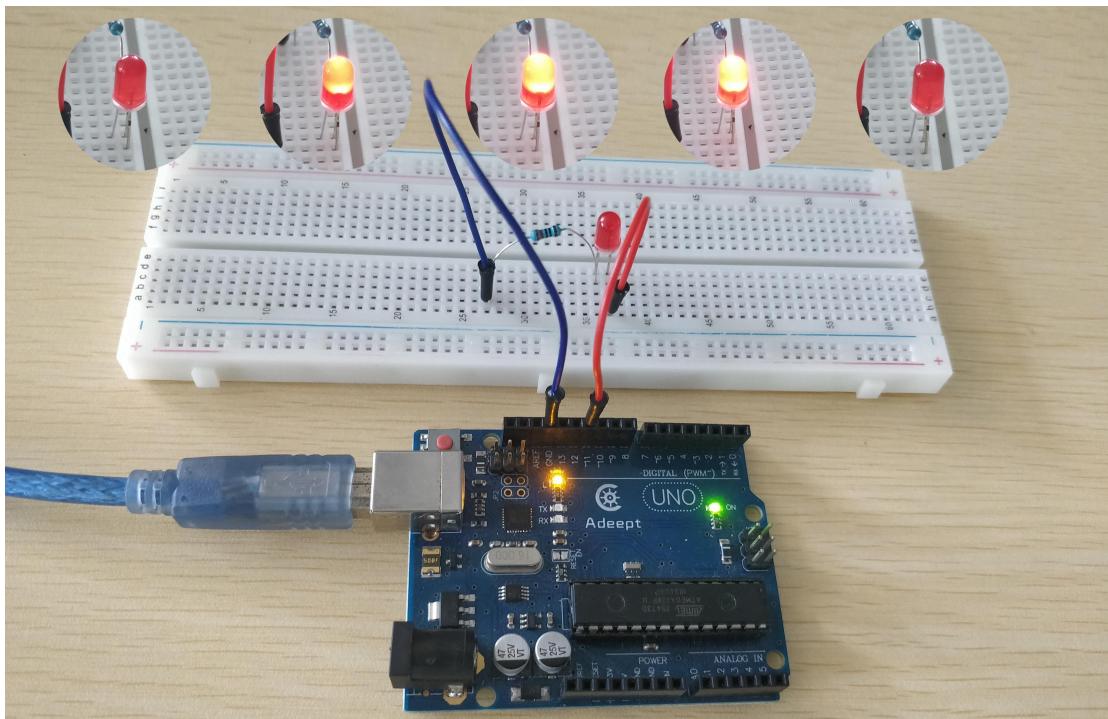
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



8. After successfully running the program, we will observe that the LED changes slowly from the extinguished state to the increasingly brighter state, and then slowly changes from the brightest state to the increasingly darker state, until it turns off. Repeat this process non-stop to form the breathing light effect. This shows that our experimental test is successful. The physical connection diagram of the experiment is as below:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we program to make Breathing LED on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

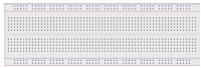
In the GwBlock graphical editor, all code programs are executed from **Setup**. First initialize the pin number of ledpin to 11 through the instruction block **set ledpin to 11**, and the ledpin is set to the Output mode through the instruction block **Initialize PIN : PIN ledpin , Output status k ;** in the while loop statement, increase the brightness of LED with PWM. Use the command **Delay 0.015 second (s)** to keep the LED in the corresponding state for 0.015s. Decrease the brightness of LED with PWM through the instruction block **PWM, set up PWM port ledpin , status: a**. Use the command **Delay 0.015 second (s)** to keep the LED in the corresponding state for 0.015s again. Then we achieve the effect of breathing to LED.



Lesson 9 The Application of the RGB LED

In this lesson, we will learn how to control RGB LED.

1. Components used in this course

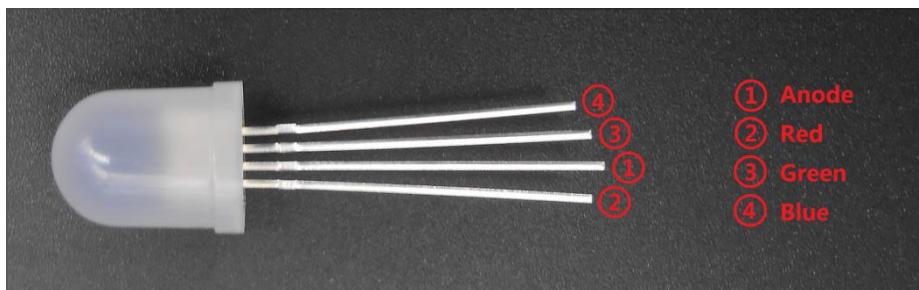
Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	3	
RGB LED	1	

2. The introduction of the RGB LED

(1)The RGB LED

RGB LED modules are commonly known as three primary colors (red, green and blue) light-emitting diodes. Commonly have four pins, a common terminal and three colors (one red, one green, one blue) control terminal. Any combination of the three colors can produce other colors. The longest pin is connected to the positive electrode. A four-wire connection with a common lead (anode or cathode) is usually used. These LED lights may have a common anode lead or a common cathode lead. In this lesson, common anode RGB LED lights are used. The longest pin is the common anode of the three LED lights. This pin is connected to the 5V port of the Arduino UNO, and the remaining three pins are connected to the port9,10,11 of the Arduino UNO

through current limiting resistors.

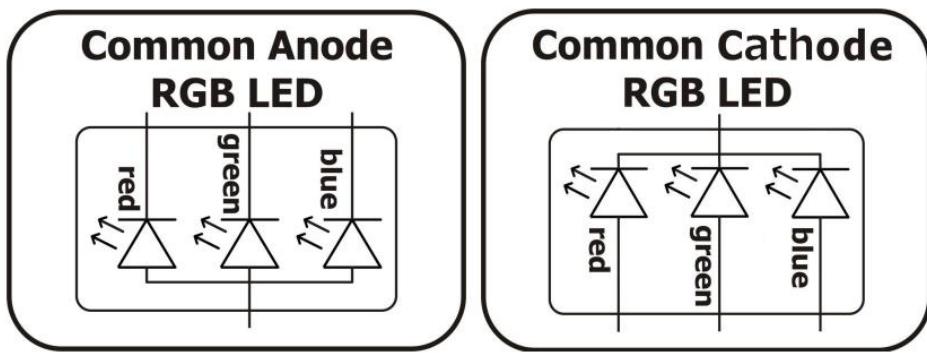


(2)Working principle of the RGB LED

RGB LED light imaging principle: RGB lights are combined with three primary colors to form an image, and there are also blue LED lights with yellow phosphors, and ultraviolet LED lights with RGB phosphors. Overall, both types have their imaging principles, but the attenuation problem and the impact of ultraviolet light on the human body are both It is a problem that is difficult to solve in the short term, so although it can meet the needs of white light, it has different results.

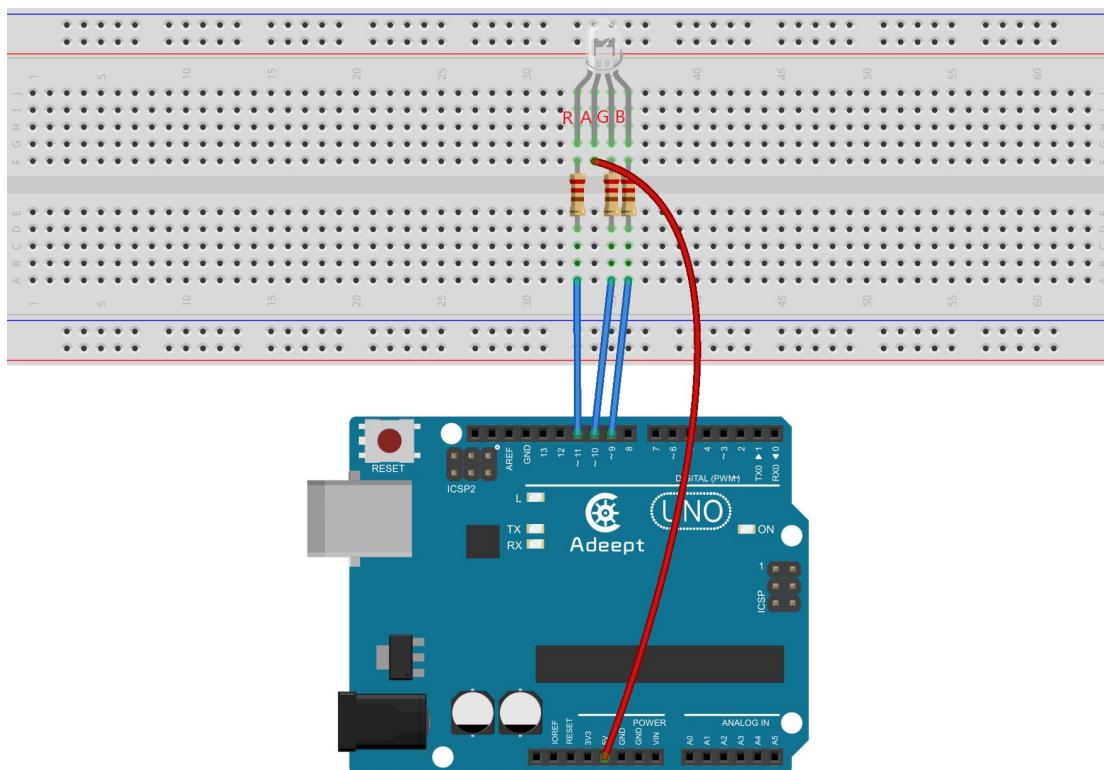
RGB LED color changing principle: When two LED lights are lit by three primary color LED lights, it can emit yellow, purple, and cyan (such as red and blue LED lights emit purple light when lit); if red, green, and blue When the LED lights up at the same time, it produces white light. If there is a circuit that can make the red, green, and blue LED lights light up two by two, individually, and the three primary colors simultaneously, then he can emit seven different colors of light.

We used ordinary anode RGB LED lights in this experiment. The longest pin is the common anode of the three LED lights. This pin is connected to the +3.3V pin of the Raspberry Pi, and the remaining three pins are connected to the pin11, pin12, and pin13 of the Raspberry Pi through current limiting resistors. In this way, we can control the color of RGB LED through 3 PWM signals.

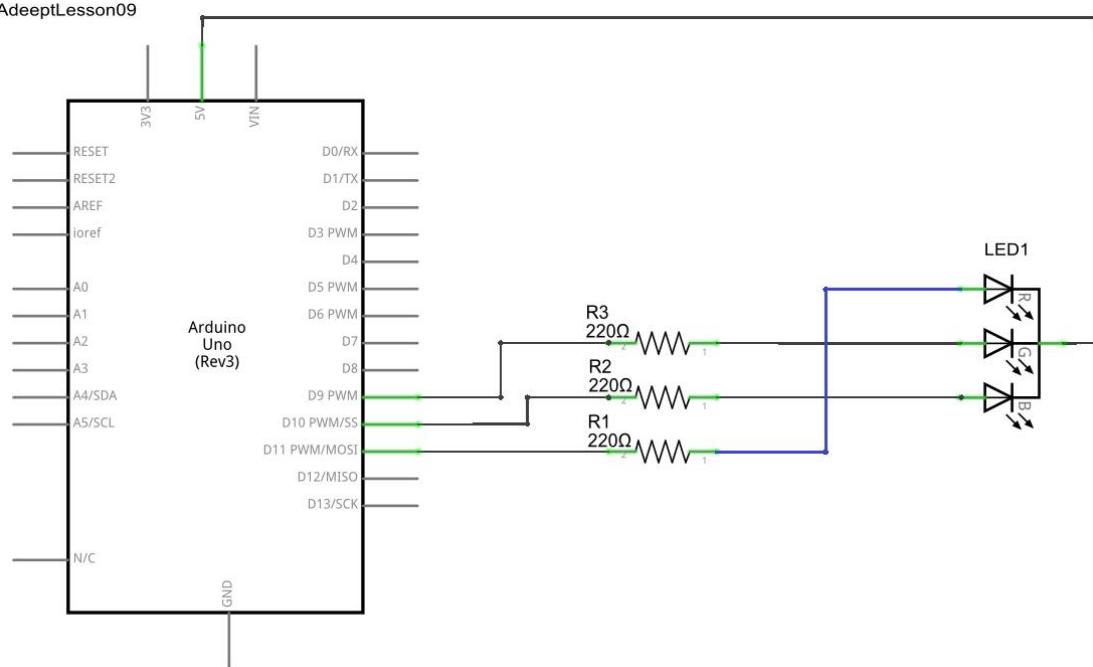


3.Wiring diagram (Circuit diagram)

Before the experiment, we connected them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. Use 220Ω resistor. As shown in the following figure:



AdeeptLesson09



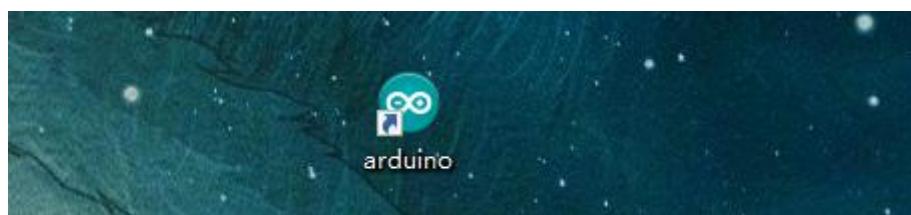
4. How to change the color of RGB LED

We provide two different methods to change the color of RGB LED. One is to program the RGB LED on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program the RGB LED on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

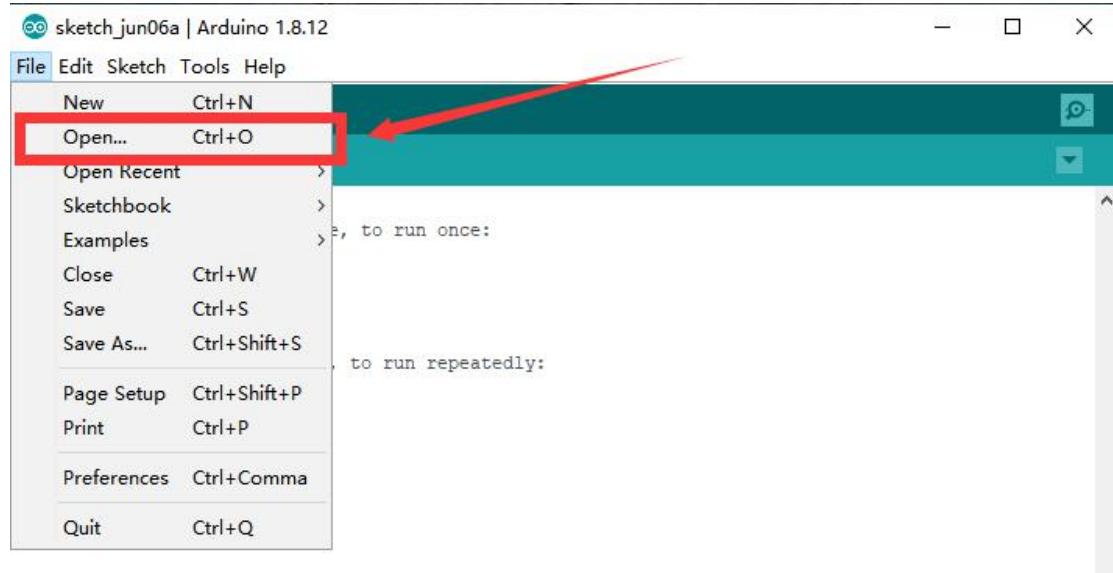
1. Using C language to program to change the color of RGB LED on Arduino UNO

(1) Compile and run the code program of this course

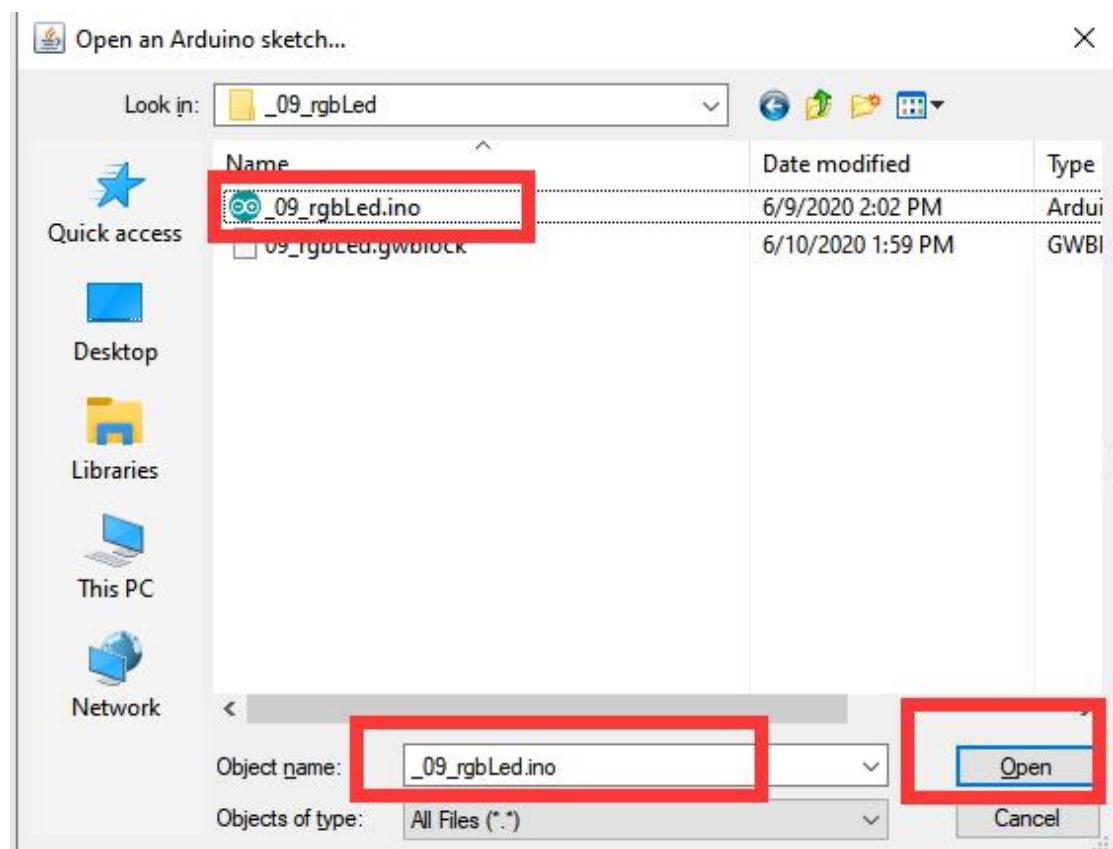
1. Open the Arduino IDE software, as shown below:



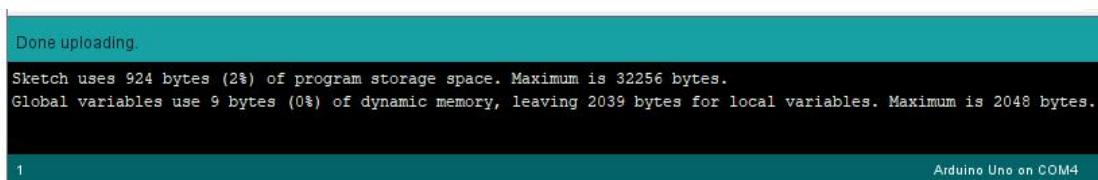
2. Click Open in the File drop-down menu:



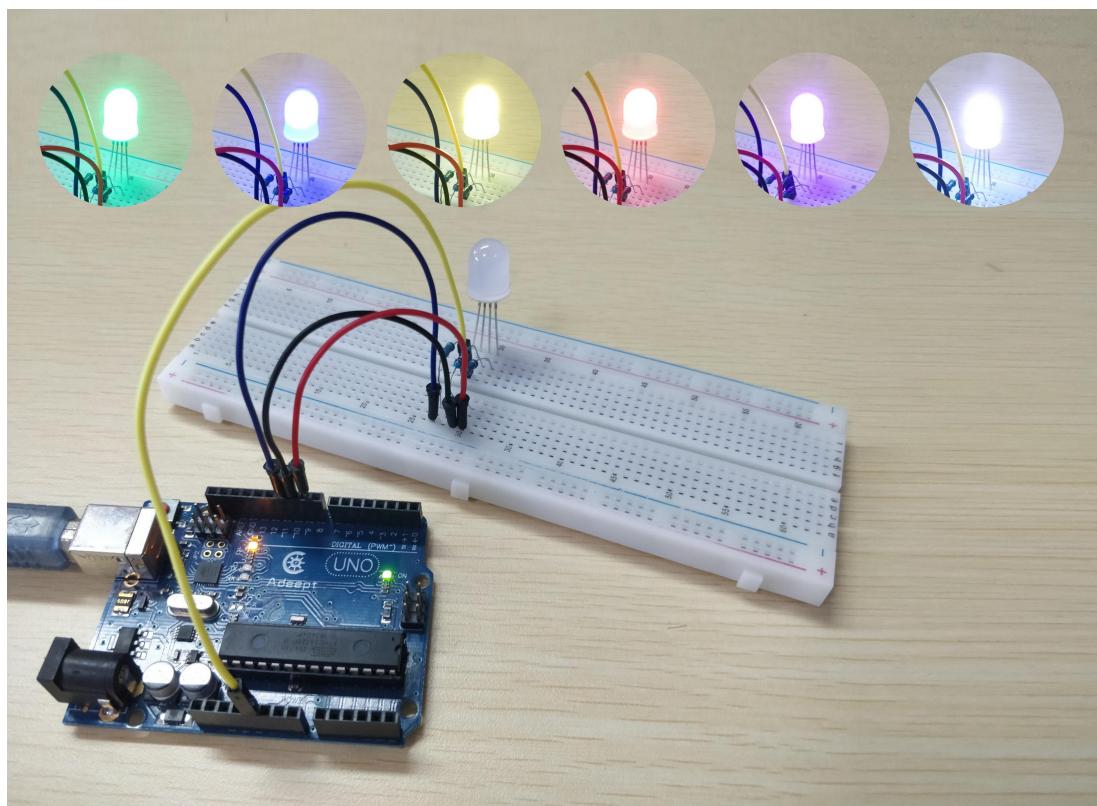
3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\09_rgbLed directory. Select _09_rgbLed.ino. This file is the code program we need in this course. Then click Open.



4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.



5. After successfully running the program, we will observe that the RGB LED lights up, and the colors of red, green, blue, yellow, white, and purple appear in turn, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to change color of RGB LED on Arduino UNO. We will introduce how our core code can be achieved:

1. Define pins of R, G, and B to be connected to ports of 11, 10, and 9 respectively and represent them with redPin, greenPin, and bluePin respectively; in the setup() function,

set the redPin, greenPin, and bluePin pins in turn through the pinMode() function to OUTPUT mode.

```
int redPin = 11; // R petal on RGB LED module connected to digital pin 11
int greenPin = 10; // G petal on RGB LED module connected to digital pin 9
int bluePin = 9; // B petal on RGB LED module connected to digital pin 10
void setup()
{
    pinMode(redPin, OUTPUT); // sets the redPin to be an output
    pinMode(greenPin, OUTPUT); // sets the greenPin to be an output
    pinMode(bluePin, OUTPUT); // sets the bluePin to be an output
}
```

2.In the loop() function, set the color of the RGB LED through the color(R,G,B) function.The value represents the corresponding color. For example, color(255,0,0) means red.Then the RGB LED will become red and light up.

```
void loop() // run over and over again
{
    // Basic colors:
    color(255, 0, 0); // turn the RGB LED red
    delay(1000); // delay for 1 second
    color(0,255, 0); // turn the RGB LED green
    delay(1000); // delay for 1 second
    color(0, 0, 255); // turn the RGB LED blue
    delay(1000); // delay for 1 second

    // Example blended colors:
    color(255,255,0); // turn the RGB LED yellow
    delay(1000); // delay for 1 second
    color(255,255,255); // turn the RGB LED white
    delay(1000); // delay for 1 second
    color(128,0,255); // turn the RGB LED purple
    delay(1000); // delay for 1 second
    color(0,0,0); // turn the RGB LED off
    delay(1000); // delay for 1 second
}
```

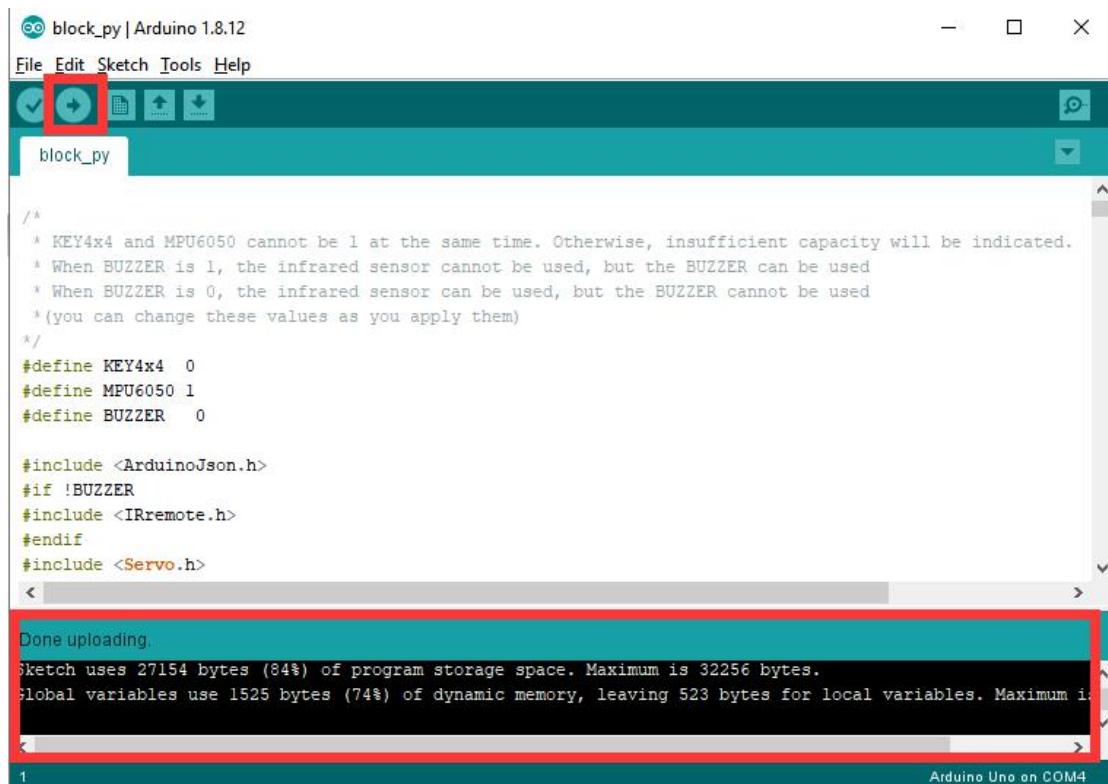
2. Using graphical code blocks to program to change color of RGB LED on Arduino UNO

(1)Connecting to GwBlock graphical editor

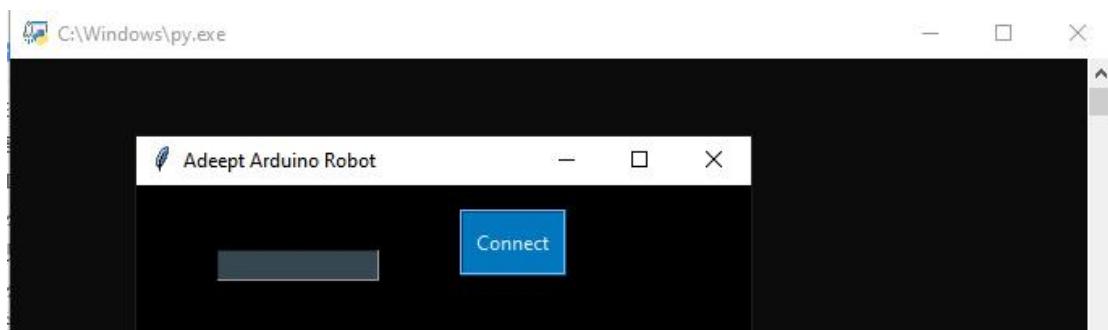
In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

- 1.Open the directory of the folder we provide to the user:

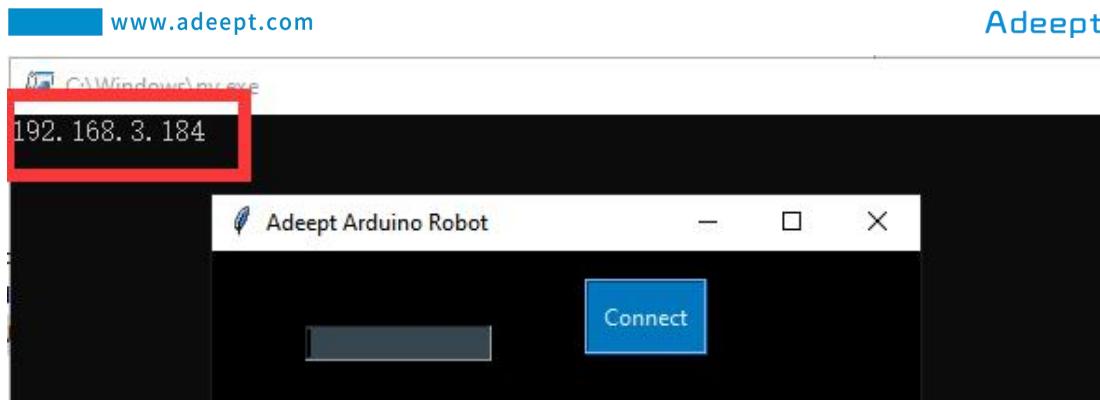
Adept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:



2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3.Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



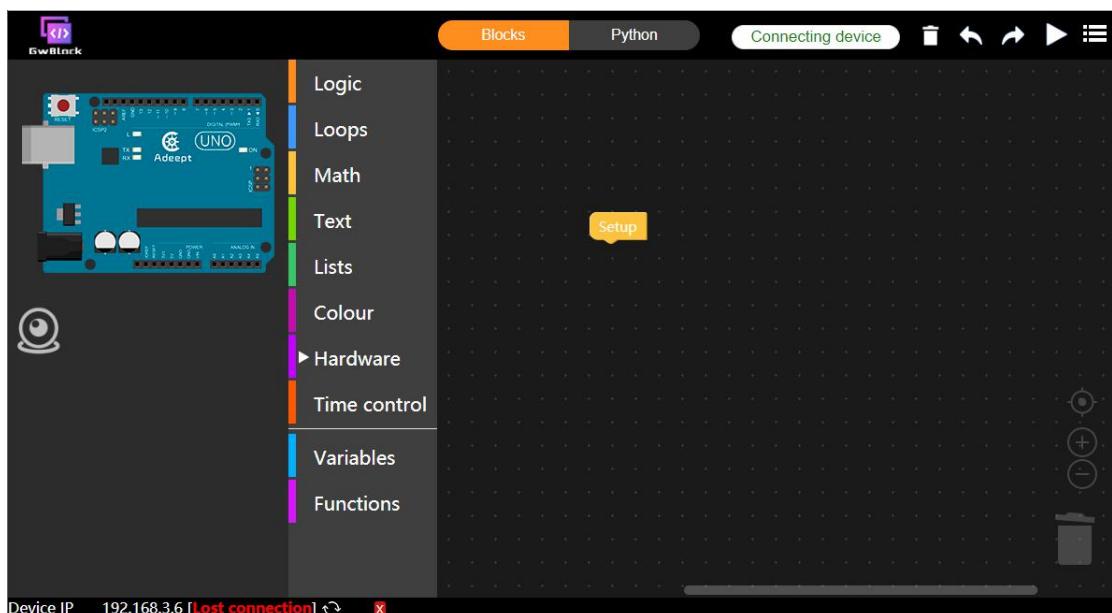
4. Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



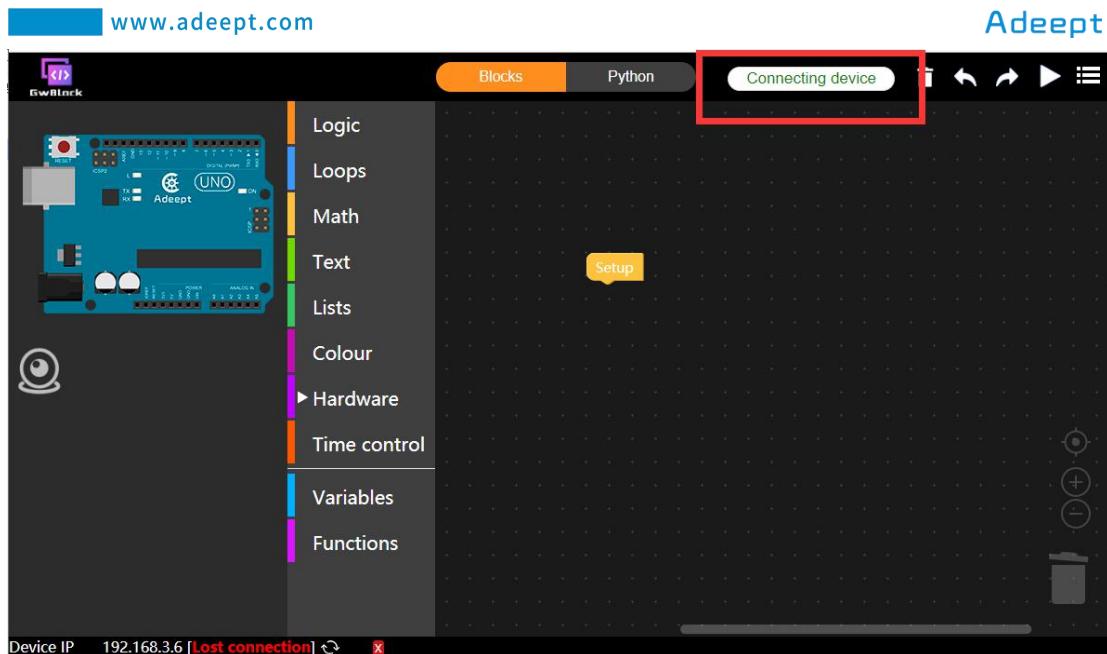
5. Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

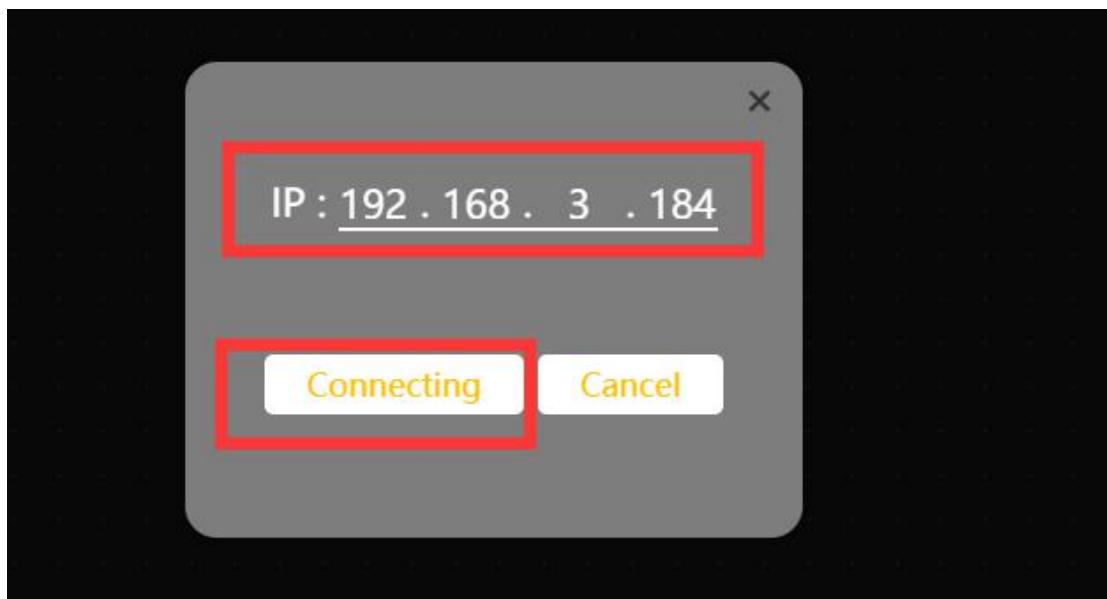
After successfully entering the website, the interface is as follows:



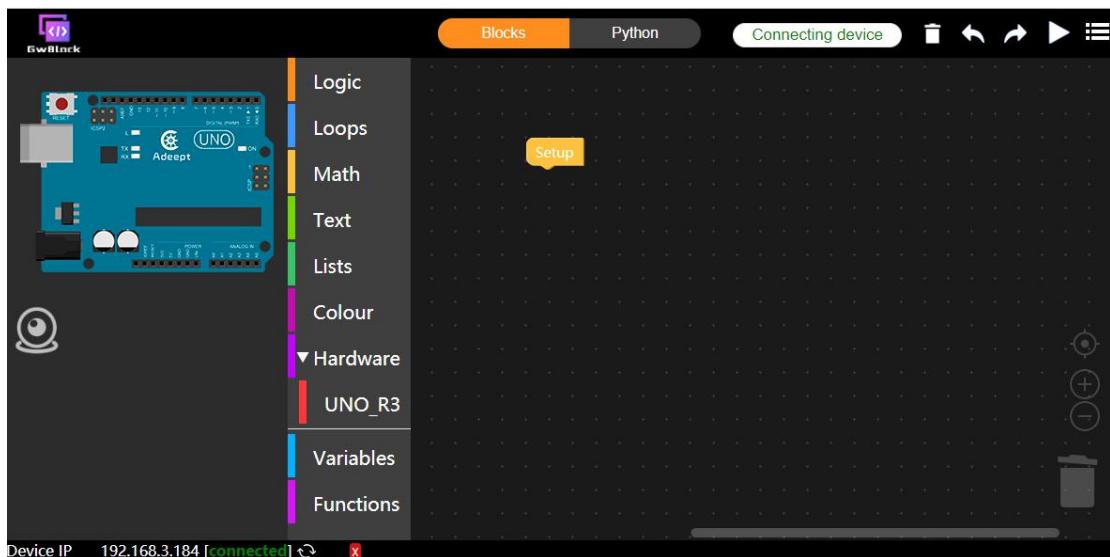
6. Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

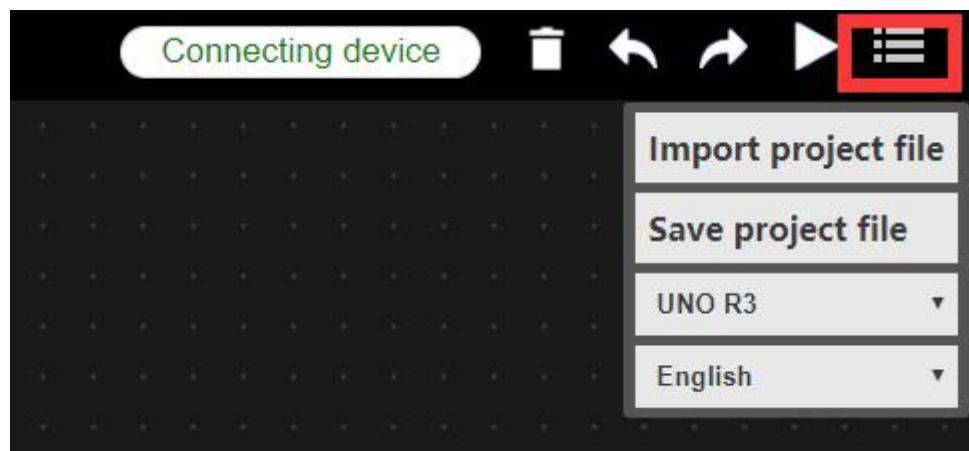


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

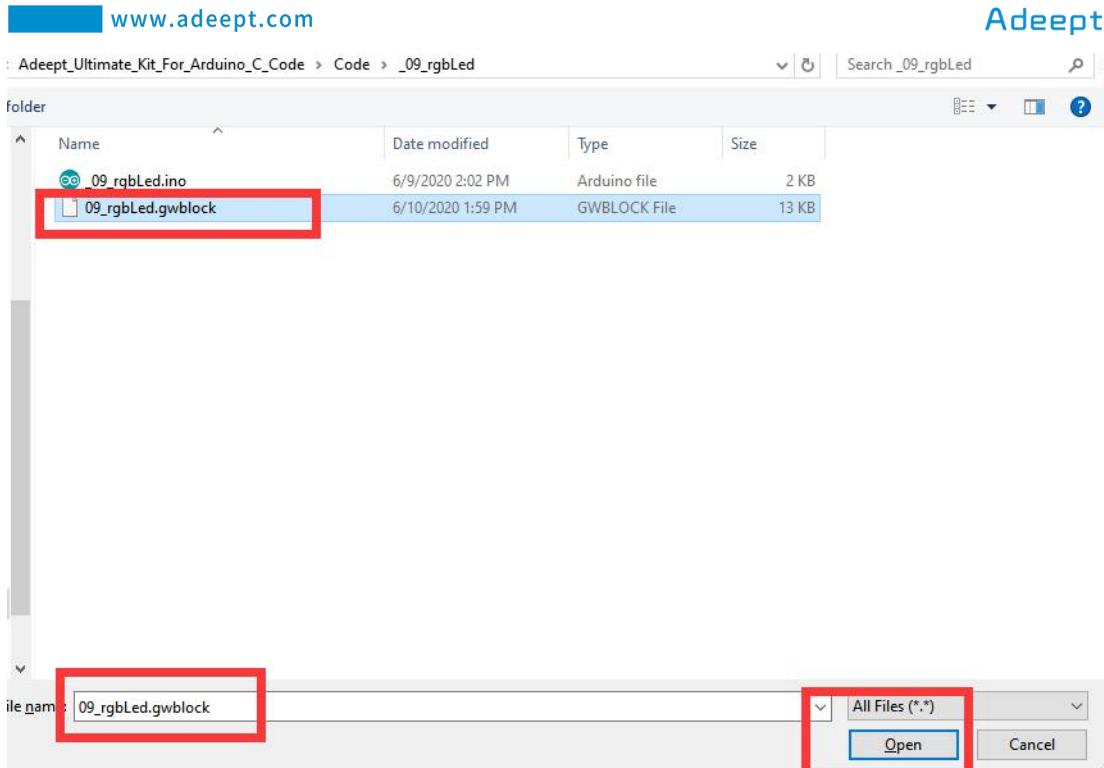
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_09_rgbLed

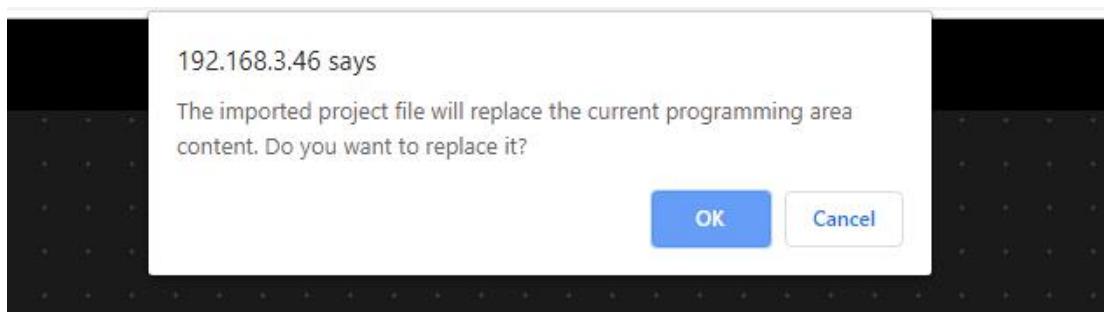
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

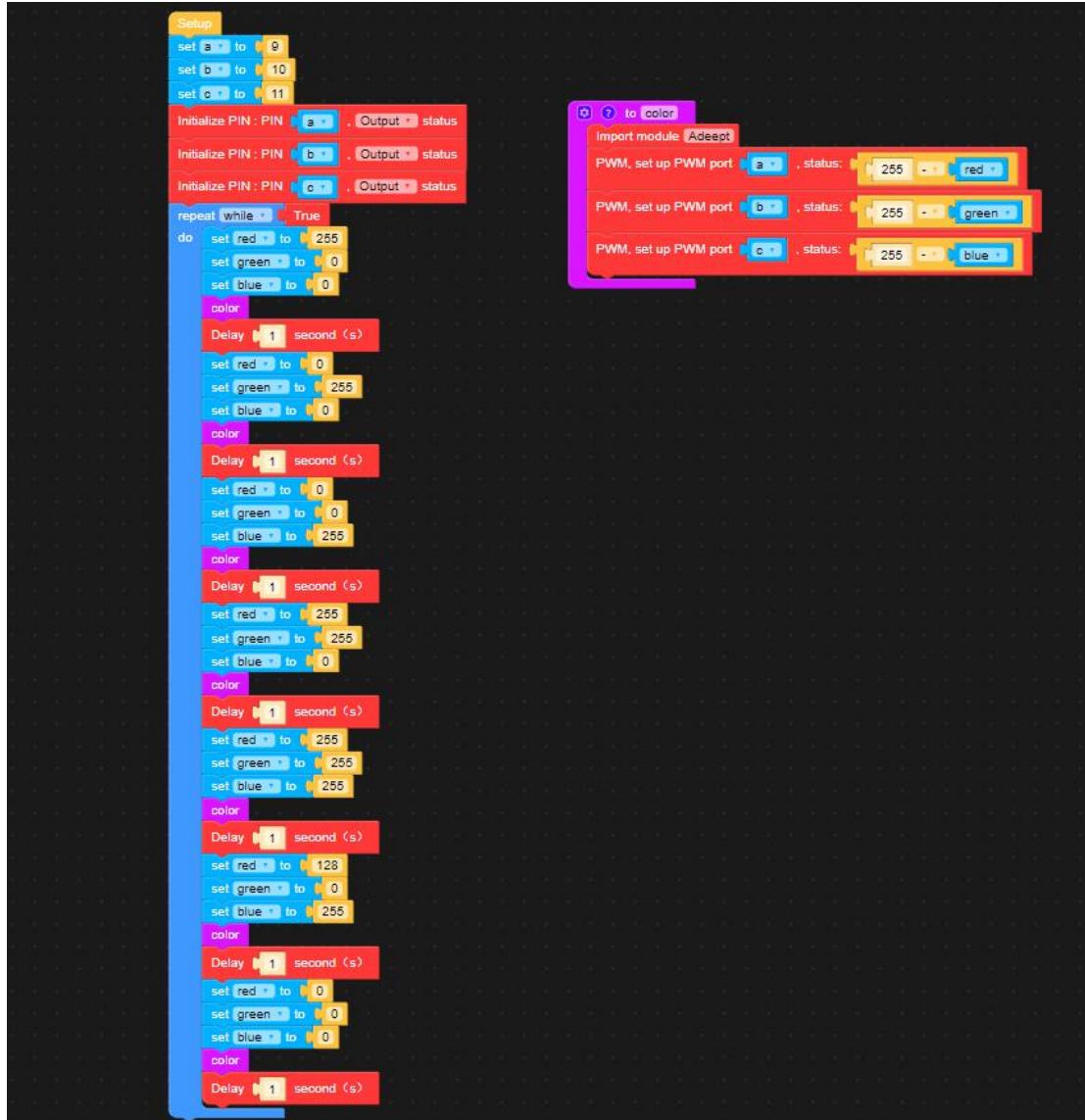
5. Select the "09_rgbLed.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



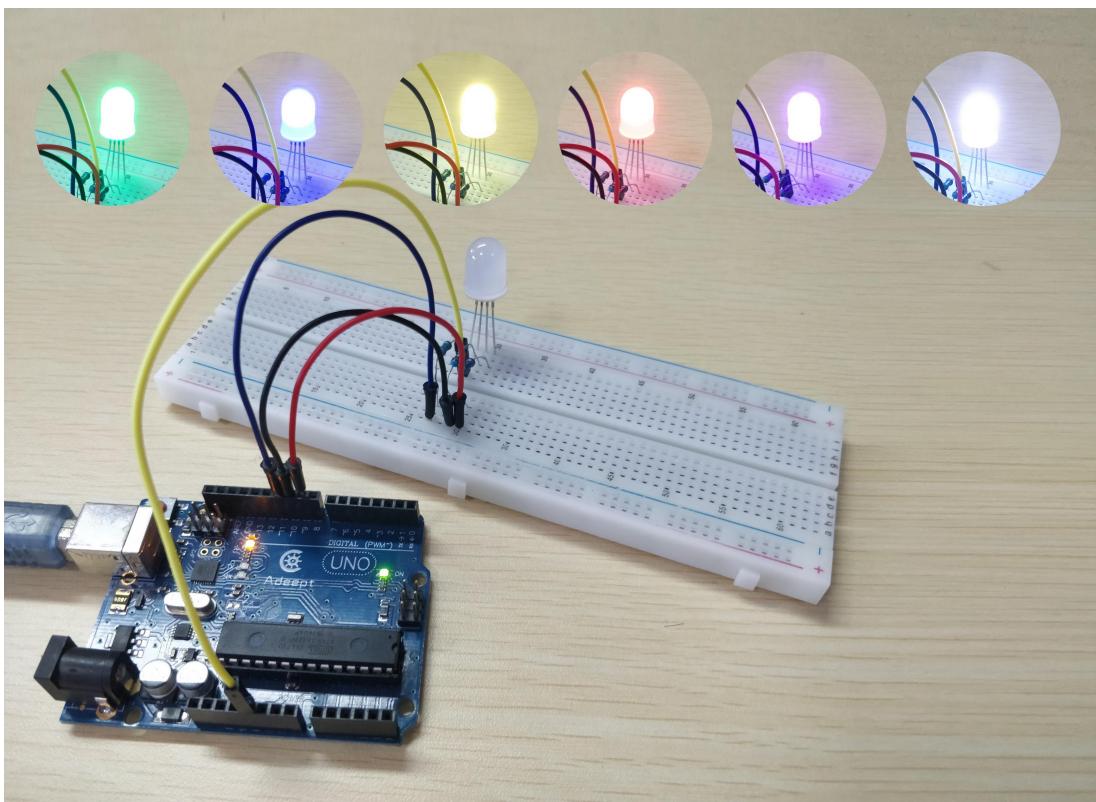
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



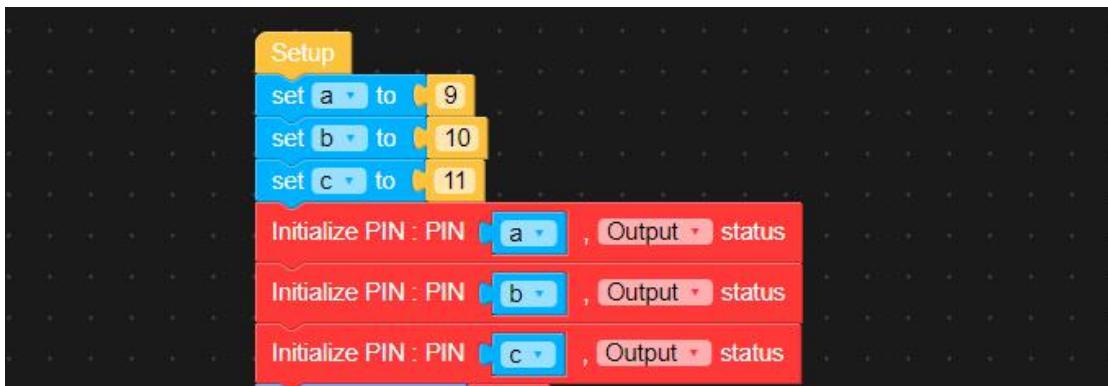
8. After successfully running the program, we will observe that the RGB LED lights up, and the colors of red, green, blue, yellow, white, and purple appear in turn, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



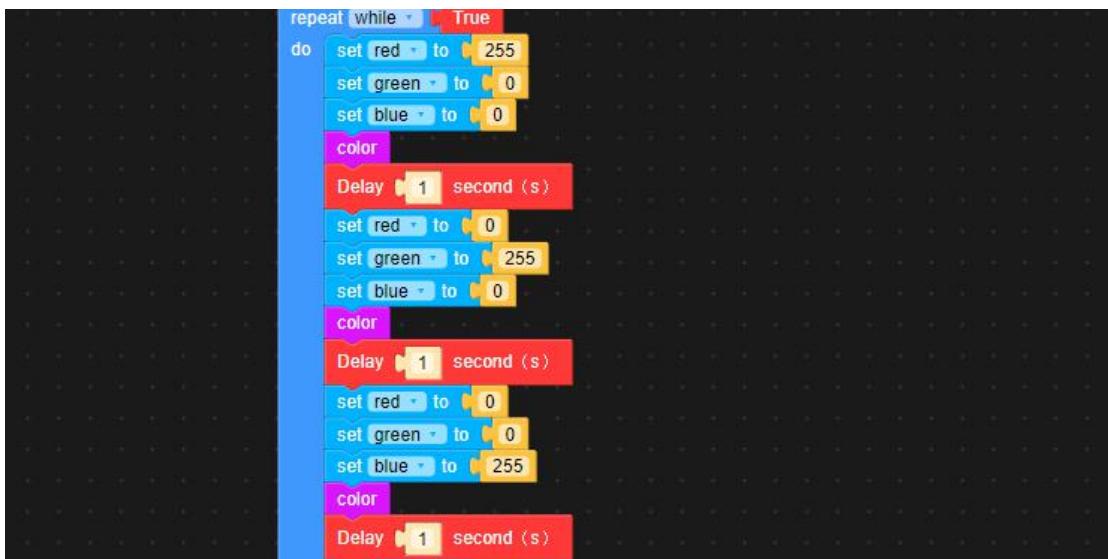
(6) Core code program

After the above hands-on operation, you must be very interested to know how we change the color of RGB LED by programming on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

In the GwBlock graphical editor, all code programs are executed from **Setup**. First define the pins of the RGB LED to connect to the ports of 9, 10, 11 on Arduino UNO through the instruction block . The pin a is set to the output mode through the instruction block . Similarly, the pins of b and c are also set to the output mode.



In the while loop statement, the values of R, G, and B are set by instructions `set [red v] to`, `set [green v] to` and `set [blue v] to` respectively to control the RGB LED to display different colors.



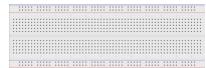
The code to change the RGB LED color is shown below:



Lesson 10 The Application of the Passive Buzzer

In this lesson, we will learn how to control the Passive Buzzer.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	1	
Passive Buzzer	1	
PNP Transistor(8550)	1	

2. The introduction of the Buzzer

(1) Buzzer

The Buzzer is an electronic sounder with an integrated structure. It is powered by DC voltage and is widely used as a sounding device in electronic products such as computers, printers, copiers, alarms, electronic toys, automotive electronic equipment, telephones, timers, and other electronic products. . There are two types of buzzer: active buzzer and passive buzzer. As shown in the figure below, the left is the active buzzer (the two pins have different lengths), and the right is the passive buzzer (the two pins have the same length).



(2)Working principle of the Buzzer

The sounding principle of buzzer is composed of vibration device and resonance device, and buzzer is divided into passive buzzer and active buzzer. The working sounding principle of passive buzzer is: square wave signal input resonant device is converted into sound signal output; the working sounding principle of active buzzer is: DC power input is generated by the amplification sampling circuit of the oscillation system under the action of the resonance device Sound signal. Our course in this section uses an active buzzer. We can program the Raspberry Pi output high and low alternately, so that the active buzzer will sound.

(3) Two kinds of Transistors (S8050 and S8550)

To make the active buzzer sound, a large current is required. However, the output current of Raspberry Pi GPIO is very weak, so we need a transistor S8050 or S8550 to drive the active buzzer. The main function of the transistor S8050 (S8550) is to amplify the voltage or current, and it can also be used to control the conduction or cut-off time of the circuit.

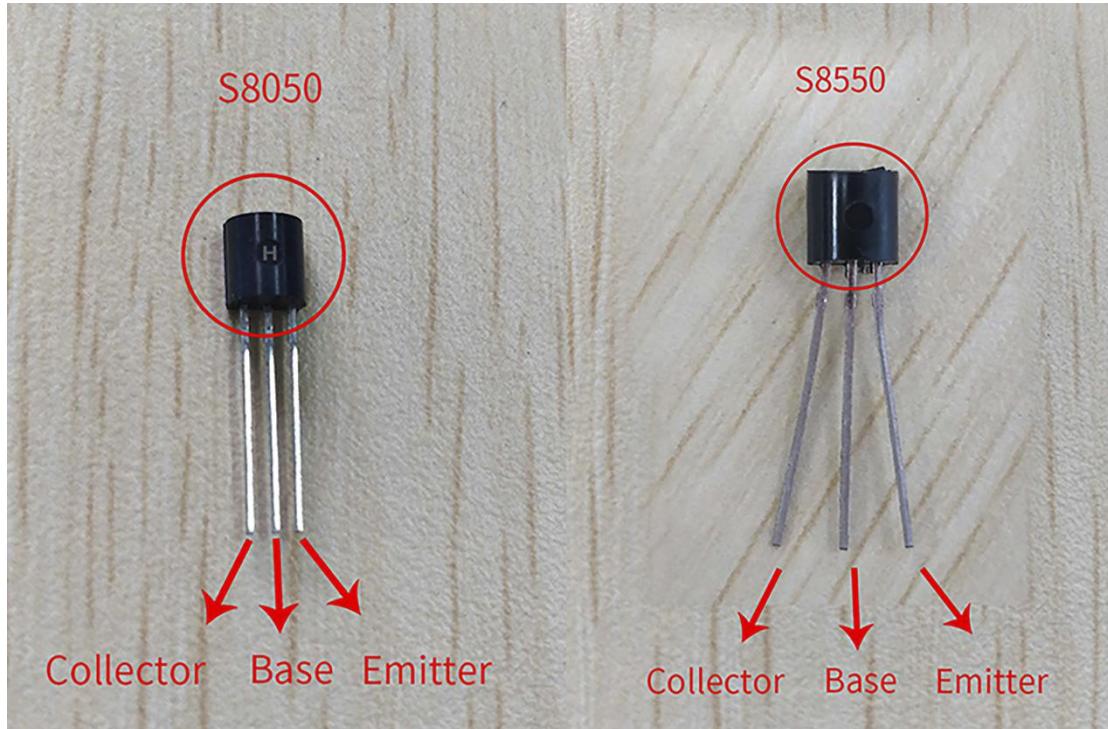
There are two kinds of transistors, one is NPN, such as the transistor S8050 used in our course; the other is a PNP transistor, such as the other S8550 we provide. The pin structure of the two transistors we use is the same. Their pin structure is as shown in the figure below. In the circuit, Emitter is abbreviated as e, Base is abbreviated as b,

and Collector is abbreviated as c.

- 1.Emitter**
- 2.Base**
- 3.Collector**



The S8050 and S8550 transistors provided by our course are as shown below. The one with letter H is S8050, and the other without H is S8550.



The two transistors S8050 and S8550 and the buzzer are connected in the circuit as shown below:

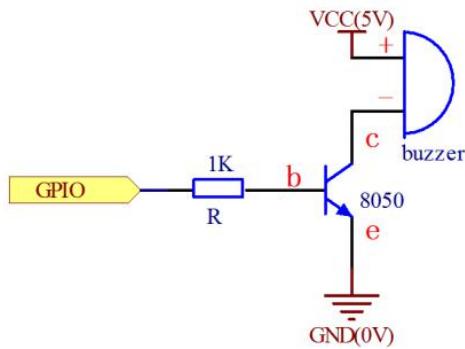


Figure1

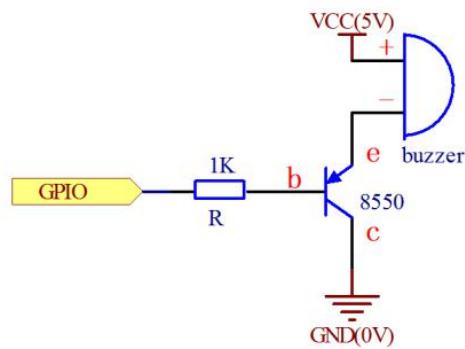


Figure2

Figure1:

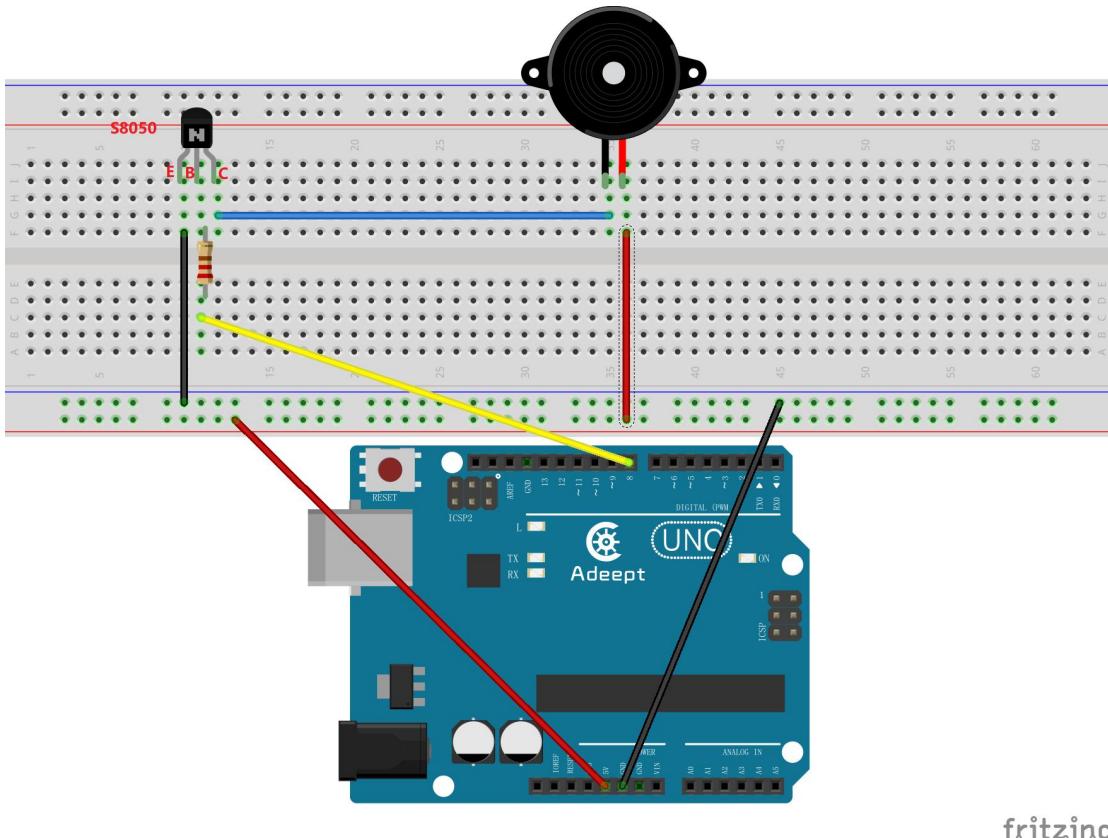
Set the Raspberry Pi GPIO as a high level, the transistor S8050 will conduct, and then the buzzer will sound; set the Raspberry Pi GPIO as low level, the transistor S8050 will cut off, then the buzzer will stop.

Figure2:

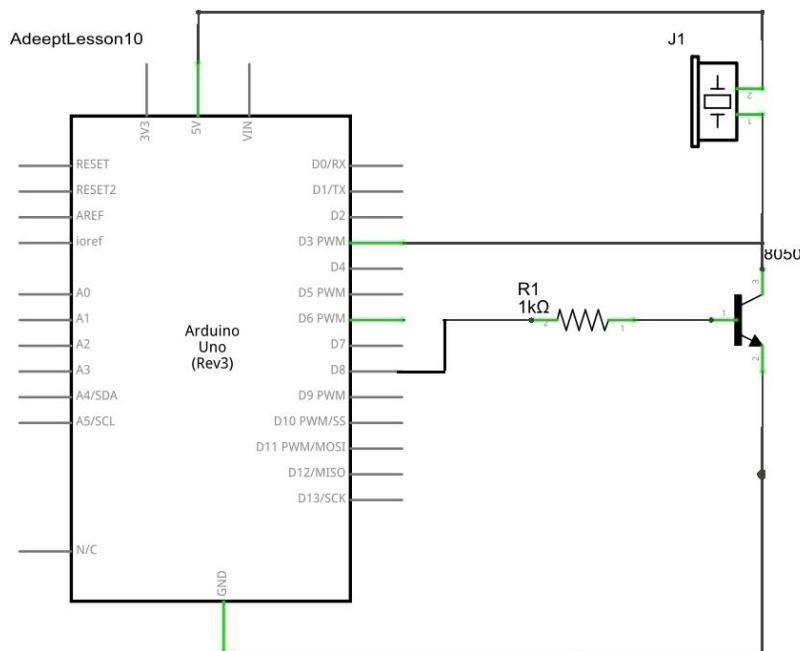
Set the Raspberry Pi GPIO as low level, the transistor S8550 will conduct, and the buzzer will sound; set the Raspberry Pi GPIO as a high level, the transistor S8550 will cut off, then the buzzer will stop.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connected them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. Use 220Ω resistor. As shown in the following figure:



fritzing



4. The Application of the Passive Buzzer

We provide two different methods to control the Passive Buzzer. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to

master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

1.Using C language to program to control the Passive Buzzer on Arduino UNO

(1)Compile and run the code program of this course

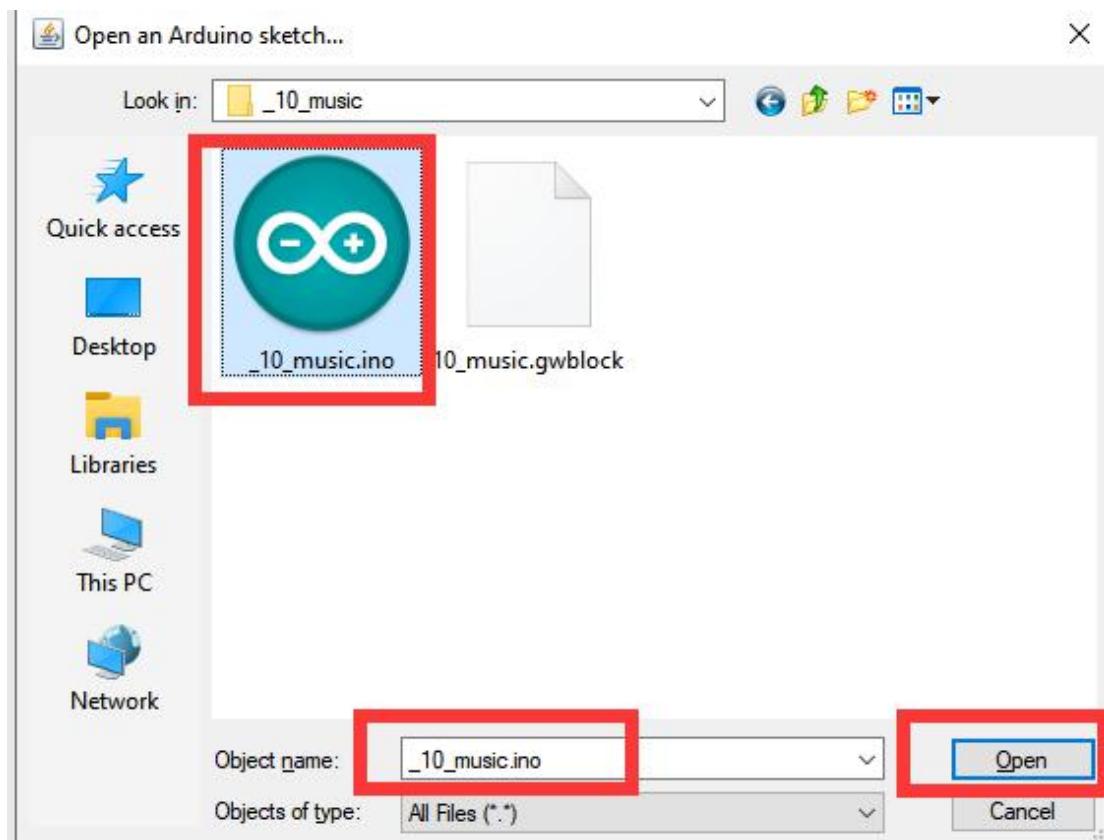
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\10_music directory. Select _10_music.ino. This file is the code program we need in this course. Then click Open.



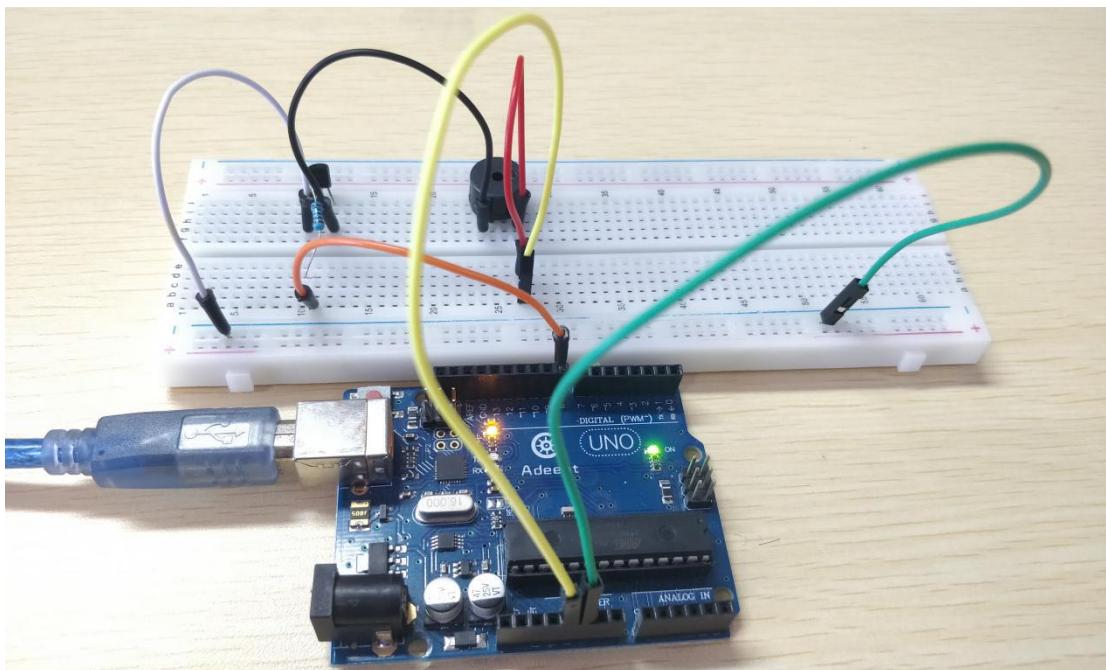
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                                         Arduino Uno on COM4
```

5. After successfully running the program, the passive buzzer will sound, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we control the Passive Buzzer on Arduino UNO. We will introduce how our core code can be achieved:

Define the pin number of passive buzzer as 8 and represent it with tonepin. MUSIC represents sound frequency. In the setup() function, set the tonepin to OUTPUT mode through the pinMode(tonepin, OUTPUT) function. Drive the buzzer to sound at 500Hz through the tone(tonepin, MUSIC) function. Delay 1000ms of time with delay (1000). Turn off the buzzer through noTone (tonepin) function.

```

int tonepin=8; //Buzzer connected digital pin 8
int MUSIC=500; //frequency

void setup()
{
    pinMode(tonepin,OUTPUT); //Digital pin 8 output mode
    tone(tonepin,MUSIC);
    delay(1000);
    noTone(tonepin);
}

```

2. Using graphical code blocks to control the Passive Buzzer on

Arduino UNO

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

```
/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <dht11.h>
#include <Keypad.h>
```

2. Due to the use of special components in this course: Passive Buzzer, we need to modify the block_py.ino file to upload the program. First, we need to modify the code

file `#define BUZZER 0` into `#define BUZZER 1`. At the same time, we need to modify `#define MPU6050 1` into `#define MPU6050 0`. It will show as below:

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

```

3. After the modification is completed. It will show as below:

```

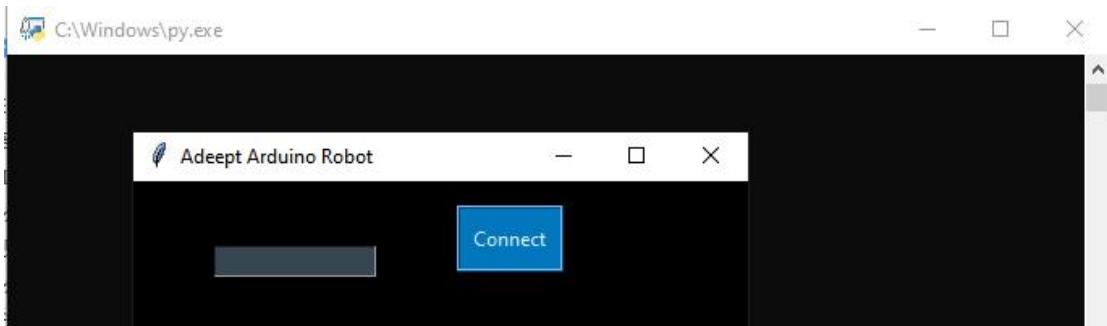
/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 0
#define BUZZER 1

```

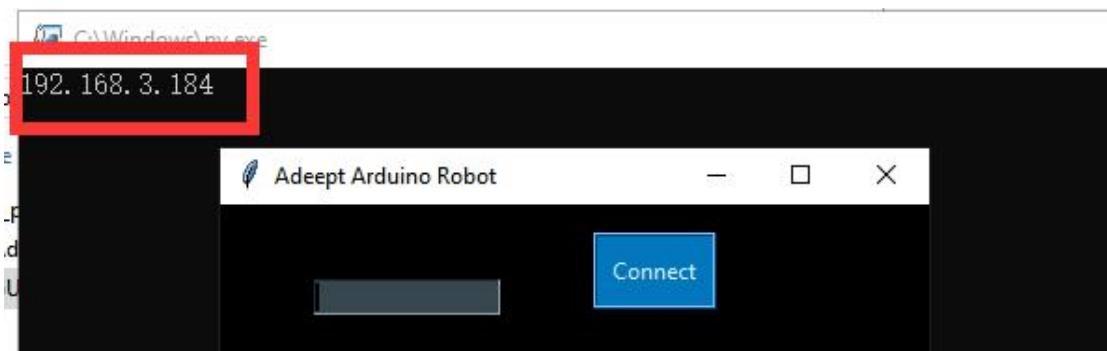
4. After opening, click  to upload the code program to the Arduino UNO. It will show as below:



5. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



6. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



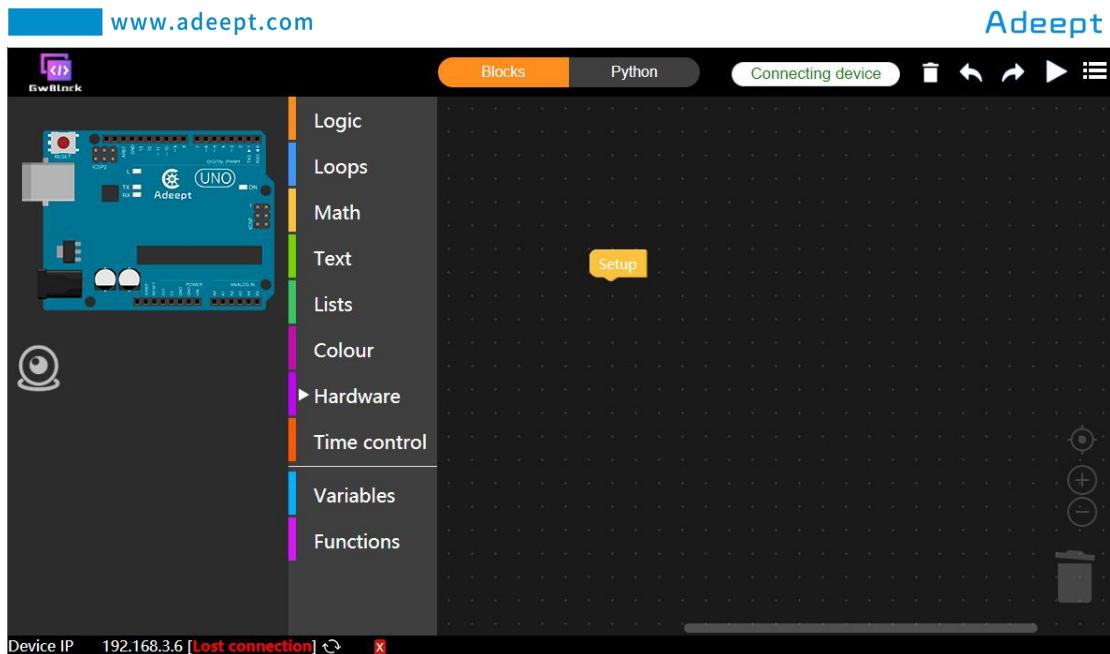
7. Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



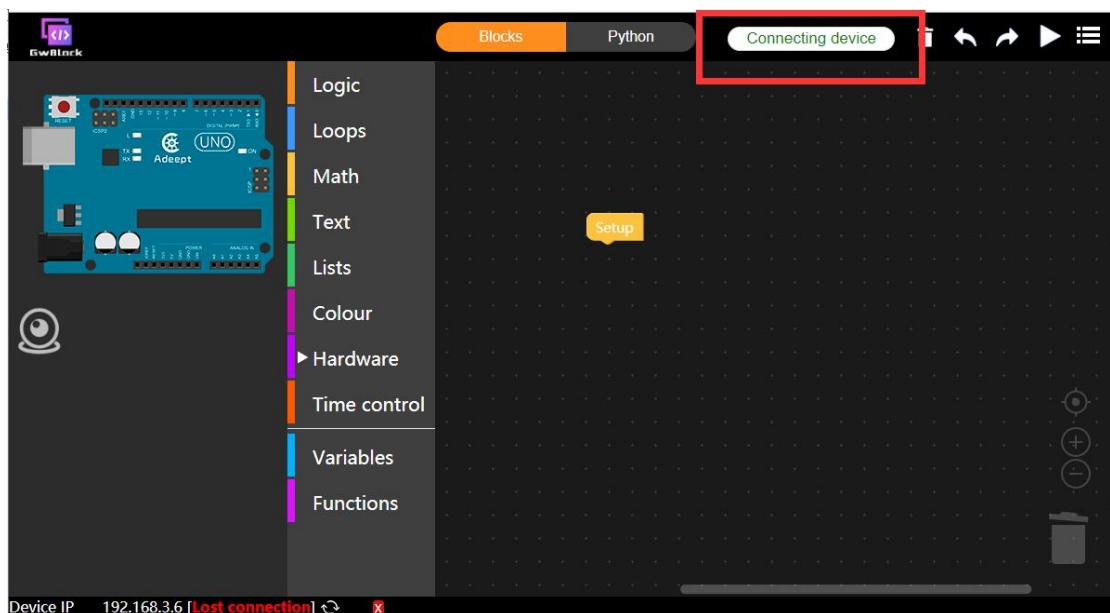
8. Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

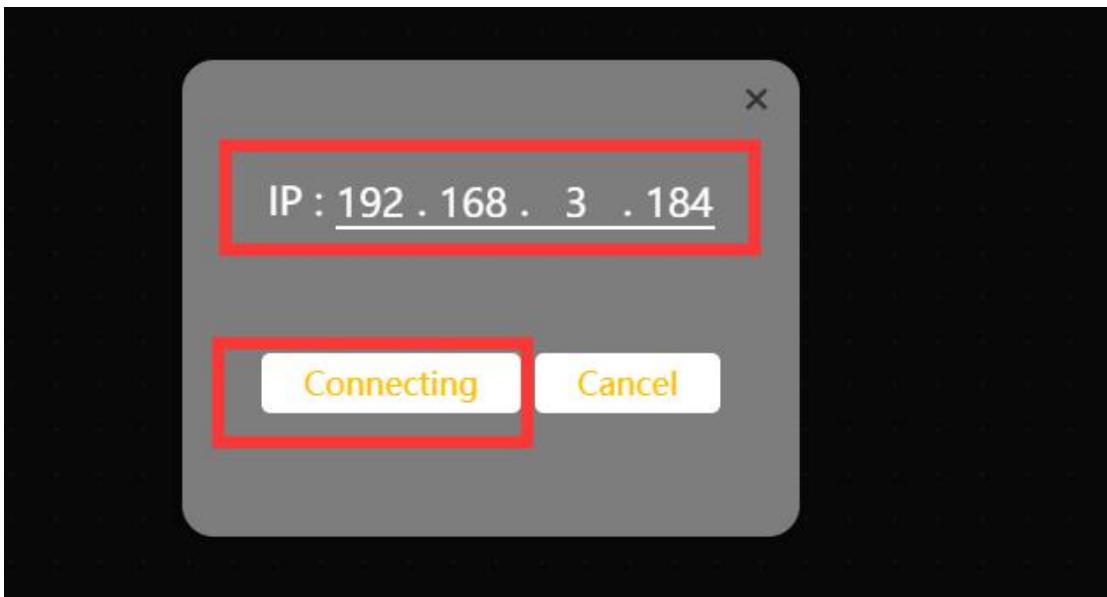
After successfully entering the website, the interface is as follows:



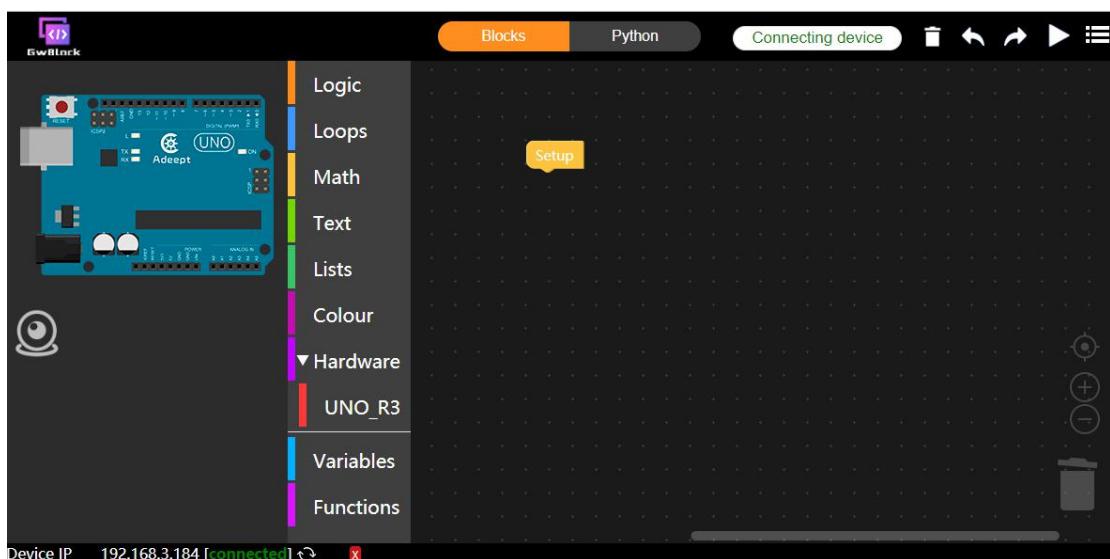
9. Click the "Connecting device" button in the upper right corner. It will show as below:



10. In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184. Then click the Connecting . It will show as below:



11. After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

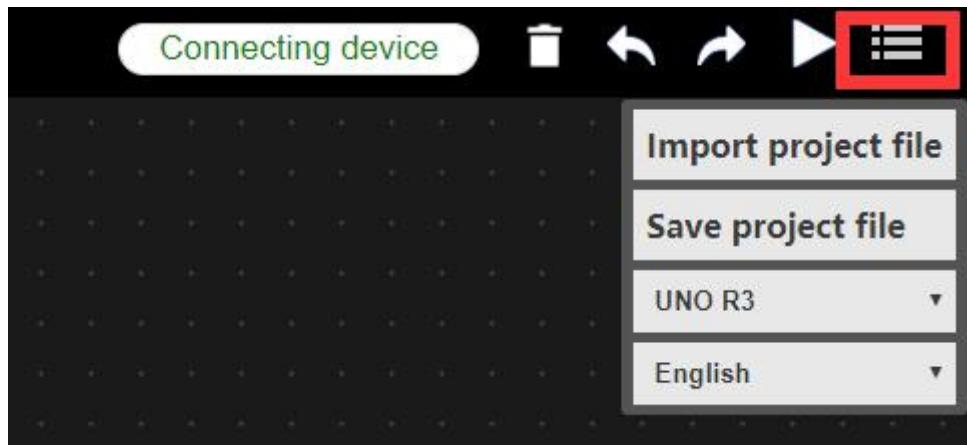


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

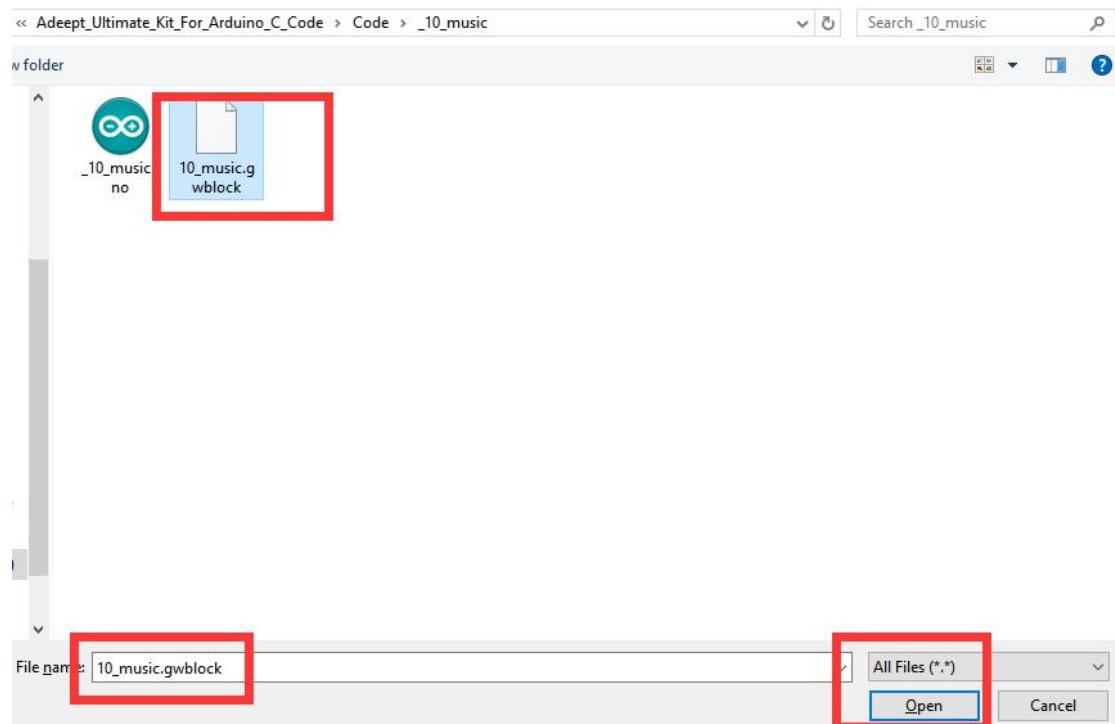
Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_10_music

It will show as below:

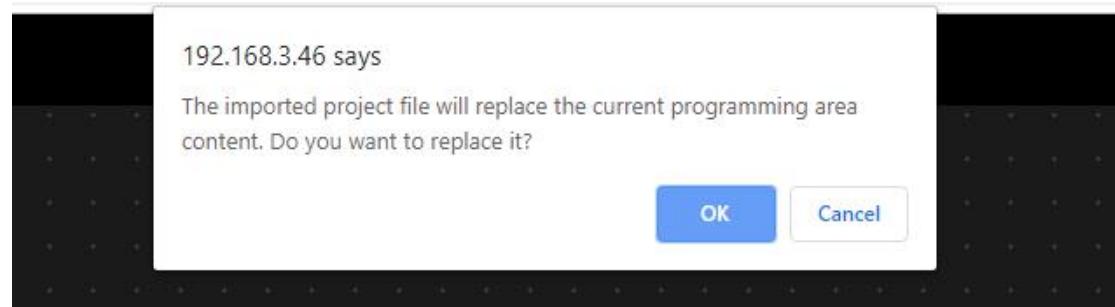
Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

5. Select the "10_music.gwblock" file. This file is the graphical code program for our

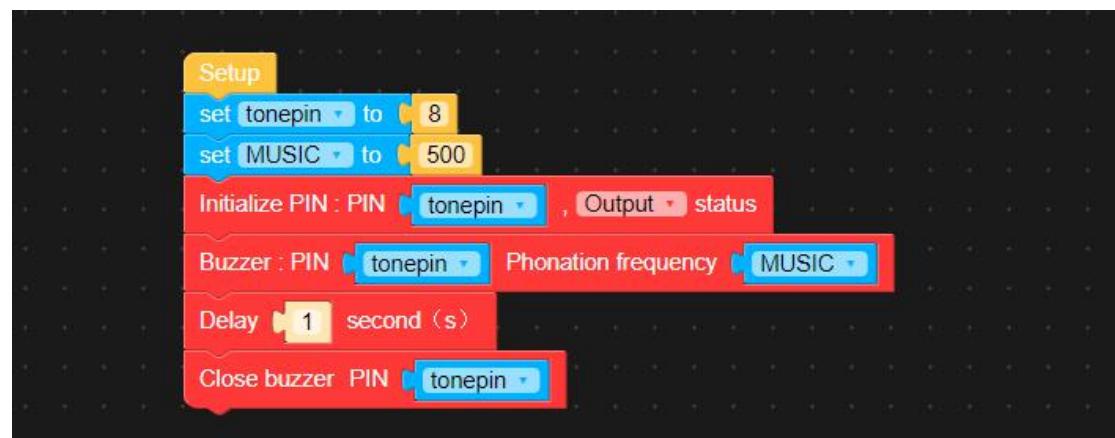
lesson. Click "Open" in the lower right corner. It will show as below:



6. Click OK. It will show as below:

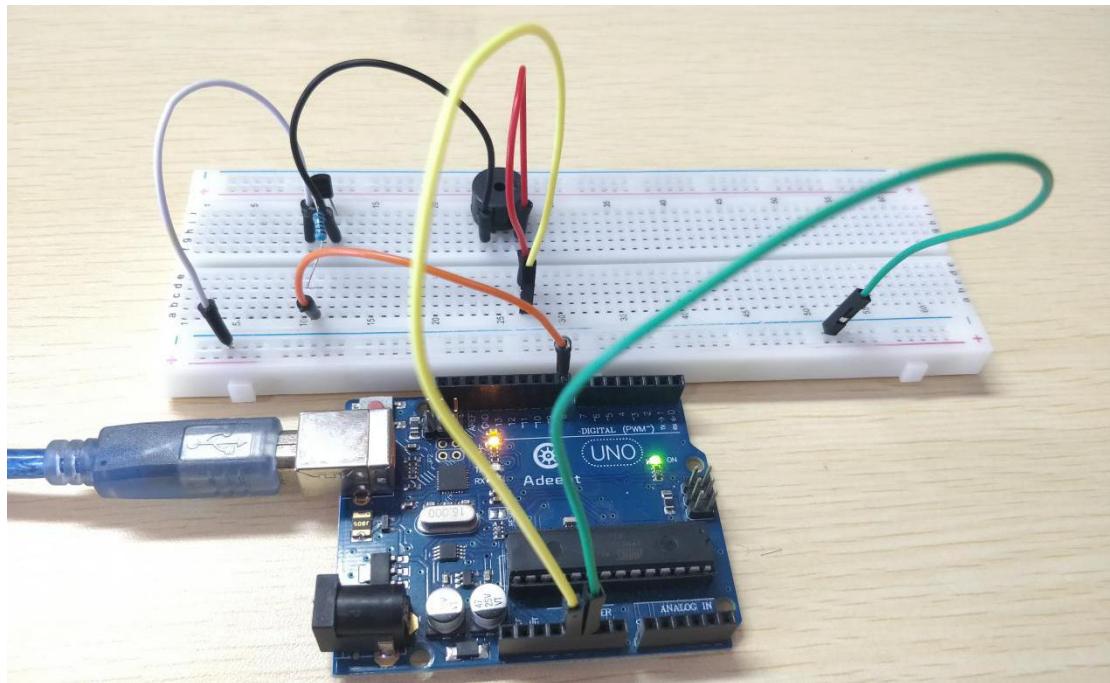


7. It will show as below after successfully opening:



8. After successfully running the program, the passive buzzer will sound, indicating

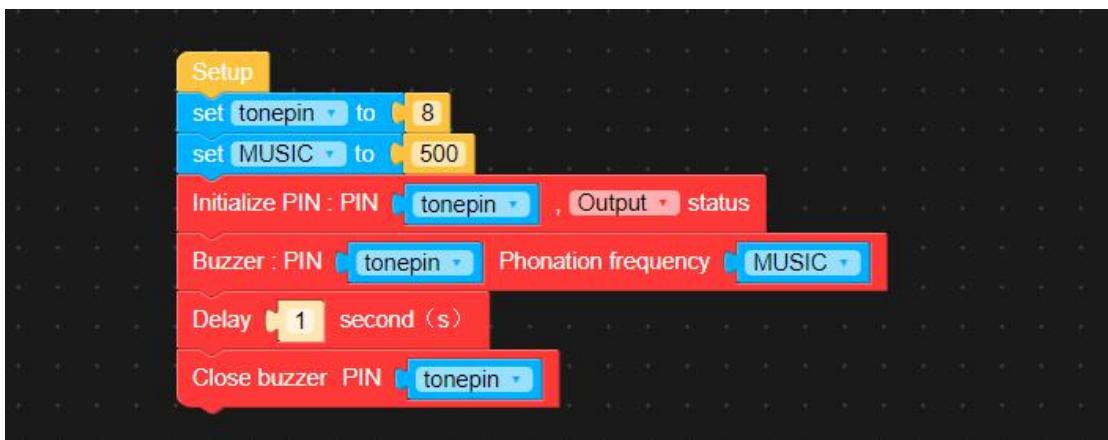
that our experimental test is successful. The physical connection diagram of the experiment is as below:



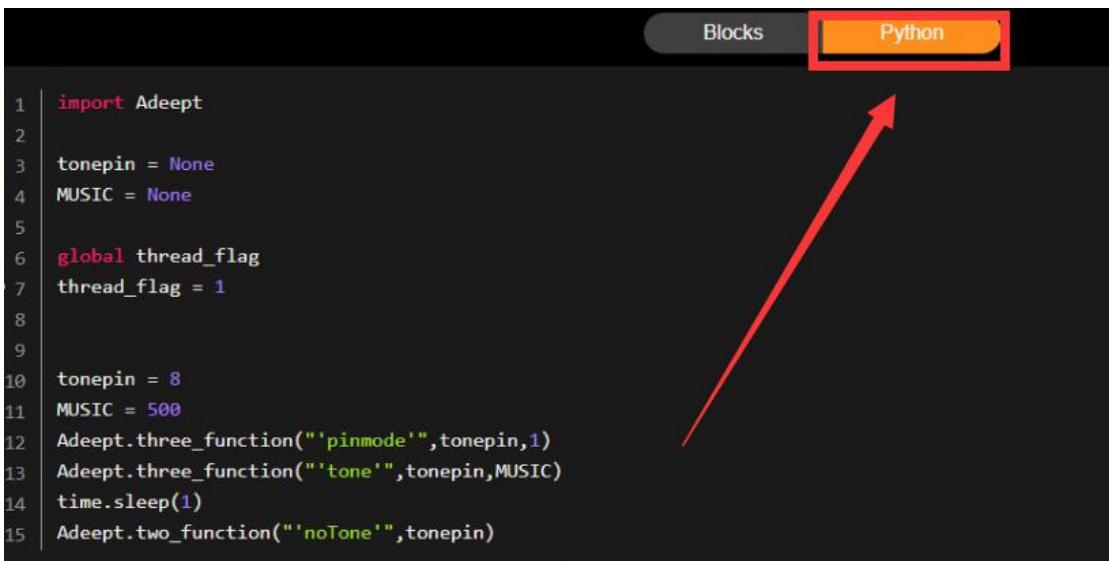
(3)Core code program

After the above hands-on operation, you must be very interested to know how to control the Passive Buzzer by programming on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

In the GwBlock graphical editor, all code programs are executed from **Setup**. First set the pin number of tonepin of Buzzer to 8 with the instruction block **set tonepin to 8**. Then set tonepin to Output mode with the instruction block **Initialize PIN : PIN tonepin , Output status**. Set the Buzzer to sound at 500 Hz with the instruction block **Buzzer : PIN tonepin Phonation frequency MUSIC**. The instruction **Delay 1 second (s)** can delay the sound for 1s. Finally, close Buzzer through command **Close buzzer PIN tonepin**.



If you click "Python", you will find that the graphical code instruction block will be converted into Python language. If you have studied Python, you will easily understand it.



```

import Adeept
tonepin = None
MUSIC = None
global thread_flag
thread_flag = 1

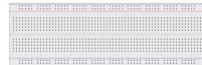
tonepin = 8
MUSIC = 500
Adeept.three_function("pinmode",tonepin,1)
Adeept.three_function("tone",tonepin,MUSIC)
time.sleep(1)
Adeept.two_function("noTone",tonepin)

```

Lesson 11 The Application of the LCD1602 and the IIC Interface

In this lesson, we will learn the application of the LCD1602 and the IIC Interface.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
LCD1602	1	
Adeept IIC Module	1	
Male to Female Jumper Wires	Several	

2. The introduction of the LCD1602 and the IIC Interface

(1)The LCD1602 and the IIC Interface

1. The LCD1602

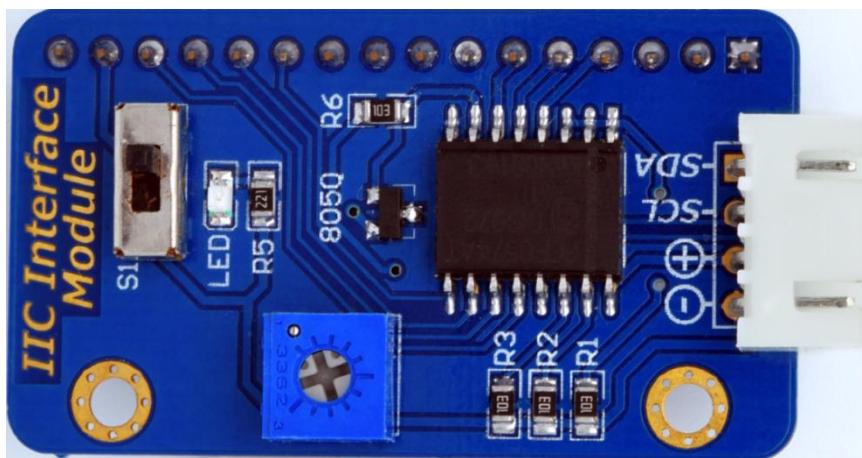
LCD1602 liquid crystal display is a character liquid crystal display module widely used. It is composed of a character liquid crystal display (LCD), a main control circuit HD44780 and its extended drive circuit HD44100, and a small number

of resistors, capacitors, and structural parts, etc., which are assembled on a PCB.



2. The IIC Interface

IIC "Inter-Integrated Circuit" is a serial communication bus used for communication between low-speed devices on board. The purpose of this communication protocol developed by Philips is to simplify the system hardware design and reduce the connections between devices. IIC can reduce the number of pins occupied by LCD1602, and reduce the number of pins to two.



(2)Working principle of the LCD1602

The LCD1602 character liquid crystal display module is a dot-matrix LCD specifically for displaying letters, numbers, and symbols. Commonly used modules are 16×1, 16×2, 20×2, and 40×2. The internal controller of the general LCD1602 character liquid crystal display is mostly HD44780, which can display English letters, Arabic numerals, Japanese Katakana and general symbols.

LCD1602 is a kind of character LCD display. The LCD has a parallel interface, meaning that the microcontroller has to manipulate several interface pins at once to control the display. The interface consists of the following pins:

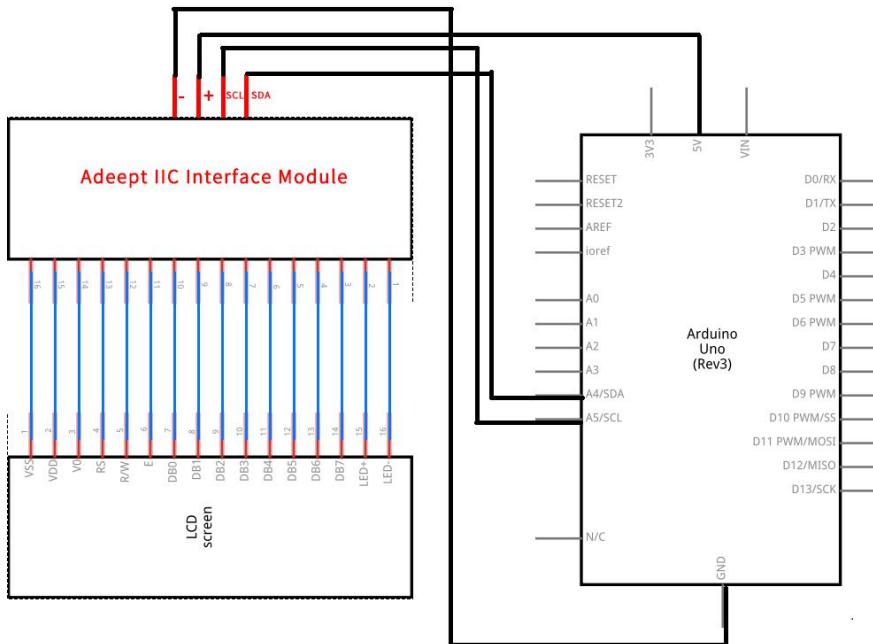
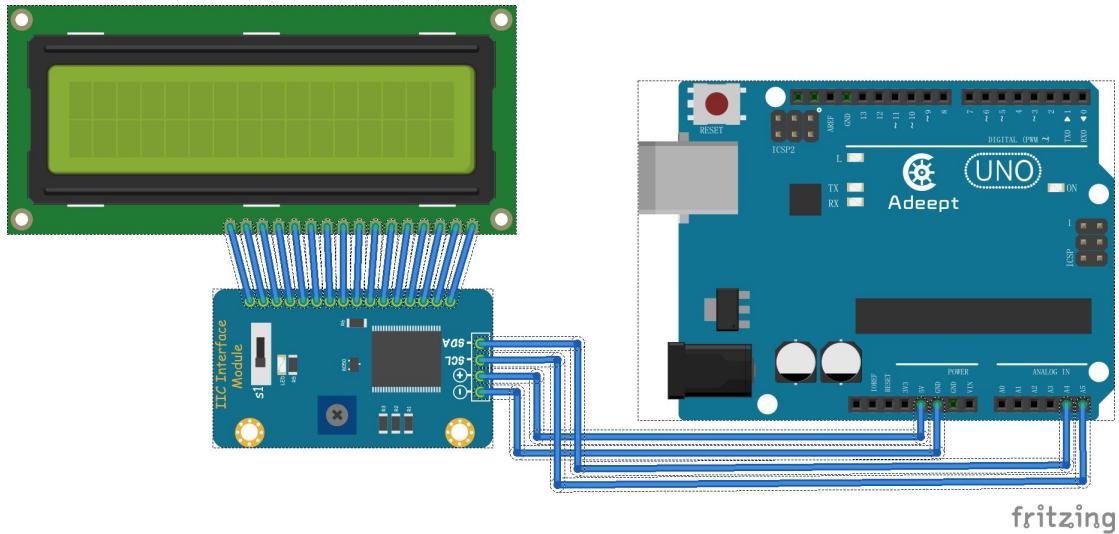
- A register select (RS) pin that controls where in the LCD's memory you're writing data to. You can select either the data register, which holds what goes on the screen, or an instruction register, which is where the LCD's controller looks for instructions on what to do next.
- A Read/Write (R/W) pin that selects reading mode or writing mode
- An Enable pin that enables writing to the registers
- 8 data pins (D0-D7). The state of these pins (high or low) is the bits that you're writing to a register when you write, or the values when you read.
- There are also a display contrast pin (Vo), power supply pins (+5V and Gnd) and LED Backlight (Bklt+ and BKlt-) pins that you can use to power the LCD, control the display contrast, and turn on or off the LED backlight respectively.

The process of controlling the display involves putting the data that form the image of what you want to display into the data registers, then putting instructions in the instruction register. The LiquidCrystal Library simplifies this for you so you don't need to know the low-level instructions.

The Hitachi-compatible LED can be controlled in two modes: 4-bit or 8-bit. The 4-bit mode requires seven I/O pins from the Arduino, while the 8-bit mode requires 11 pins. For displaying text on the screen, you can do most everything in 4-bit mode, so example shows how to control a 2x16 LCD in 4-bit mode.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. Use $220\ \Omega$ resistor. As shown in the following figure:



4.Application of the LCD1602

We provide two different methods to display characters on the LCD1602 screen. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

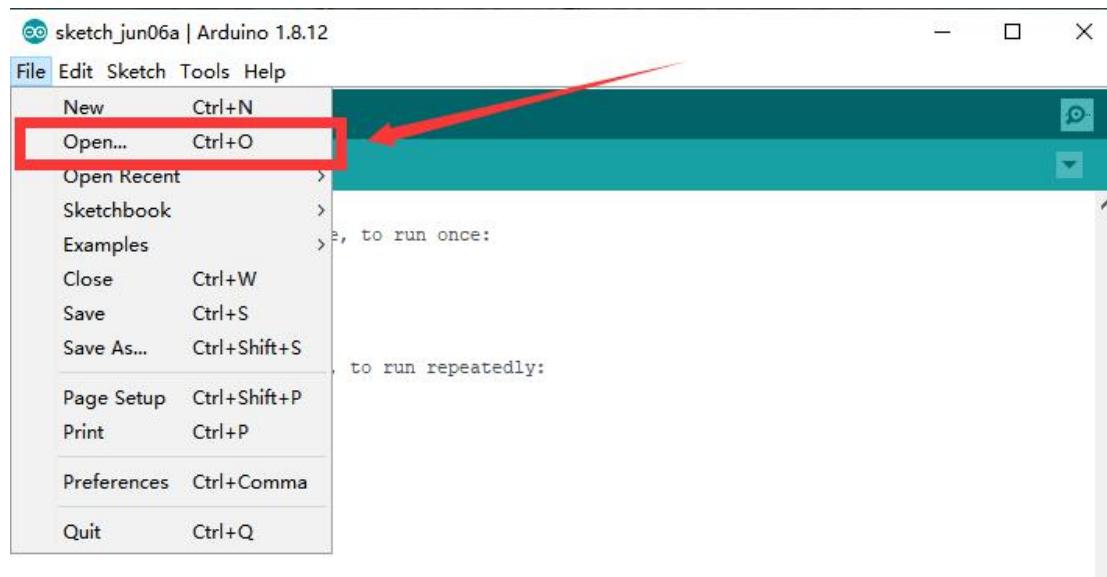
1.Using C language to program to display characters on LCD1602 on Arduino UNO

(1)Compile and run the code program of this course

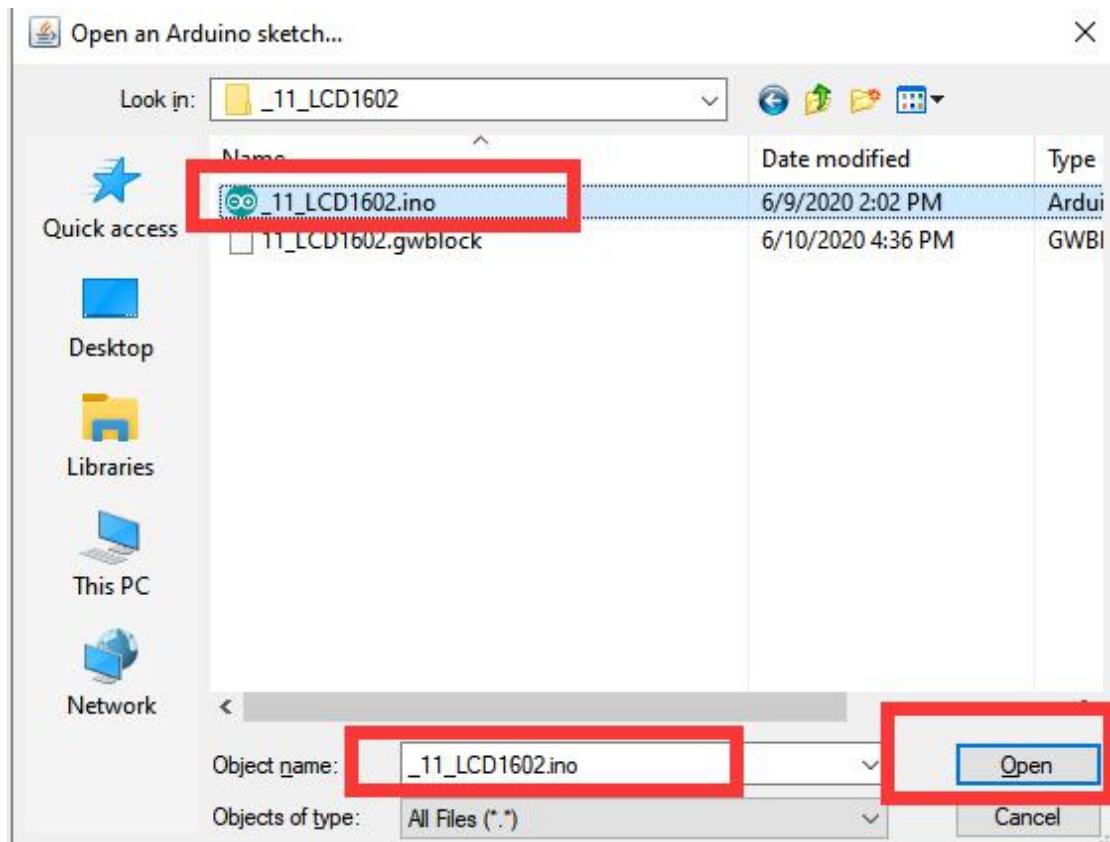
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\11_LCD1602 directory. Select _11_LCD1602.ino. This file is the code program we need in this course. Then click Open.



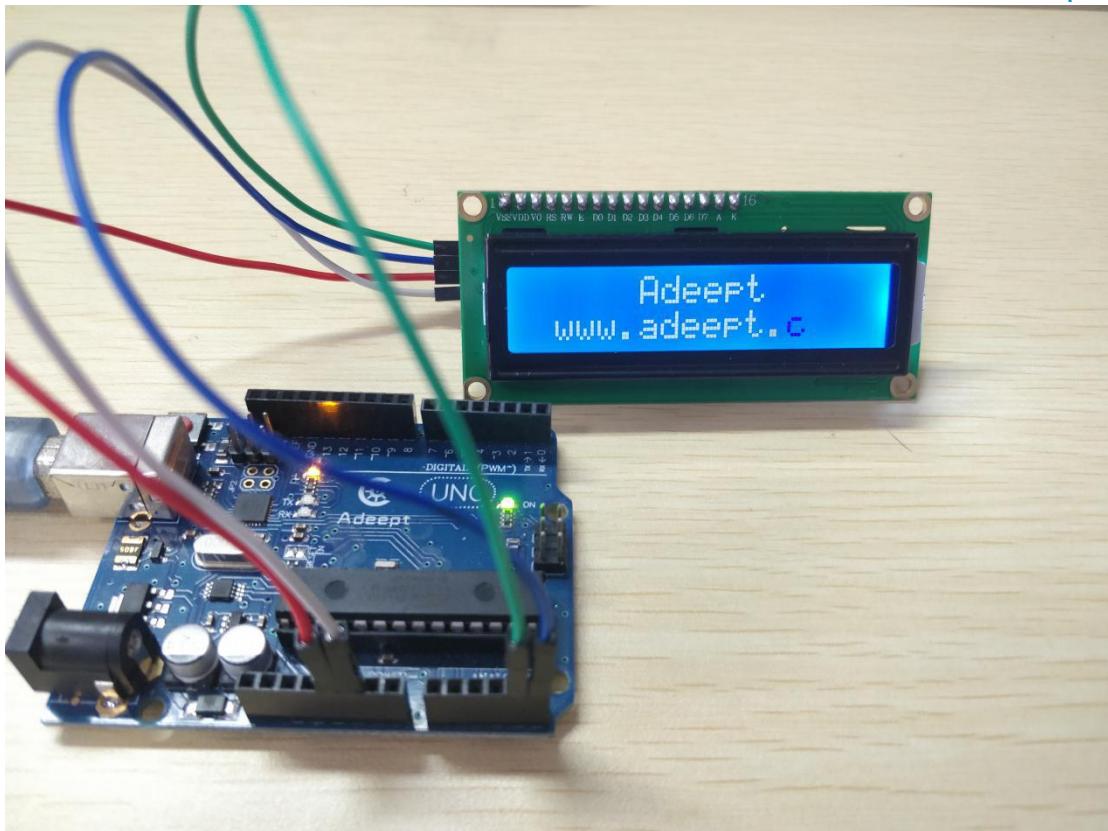
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After the program runs successfully, hello geeks will be displayed on the LCD1602 screen first, and then www.adeept.com is displayed, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we control the LCD1602 on Arduino UNO. We will introduce how our core code can be achieved:

1. Define the text information to be displayed on the LCD1602 screen through the array char array[]. In the setup() function, initialize the LCD through lcd.init(), and use lcd.backlight() to open the backlight.

```

char array1[]=" Adeept      ";           //the string to print on the LCD
char array2[]=" hello geeks!           "; //the string to print on the LCD
char array3[]=" www.adeept.com ";       //the string to print on the LCD

int tim = 250; //the value of delay time

// initialize the library with the numbers of the interface pins
LiquidCrystal_I2C lcd(0x27,16,2);

void setup()
{
    lcd.init(); //initialize the lcd
    lcd.backlight(); //turn on the backlight
}

```

2. In the loop() function, print out the corresponding text information on the LCD1602

screen through lcd.print(array[]) in the for loop statement

```

void loop()
{
    lcd.clear(); //Clears the LCD screen and positions the cursor in the upper-left corner
    lcd.setCursor(15,0); // set the cursor to column 15, line 1
    for (int positionCounter2 = 0; positionCounter2 < 30; positionCounter2++)
    {
        lcd.scrollDisplayLeft(); //Scrolls the contents of the display one space to the left.
        lcd.print(array2[positionCounter2]); // Print a message to the LCD.
        delay(tim); //wait for 250 microseconds
    }

    lcd.clear(); //Clears the LCD screen and positions the cursor in the upper-left corner.

    lcd.setCursor(0,0); // set the cursor to column 15, line 0
    for (int positionCounter1 = 0; positionCounter1 < 16; positionCounter1++)
    {
        lcd.print(array1[positionCounter1]); // Print a message to the LCD.
        delay(tim); //wait for 250 microseconds
    }

    lcd.setCursor(0,1); // set the cursor to column 15, line 1
    for (int positionCounter3 = 0; positionCounter3 < 16; positionCounter3++)
    {
        lcd.print(array3[positionCounter3]); // Print a message to the LCD.
        delay(tim); //wait for 250 microseconds
    }
}

```

2. Using graphical code blocks to program and display

characters on LCD1602 on Arduino UNO

(1) Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

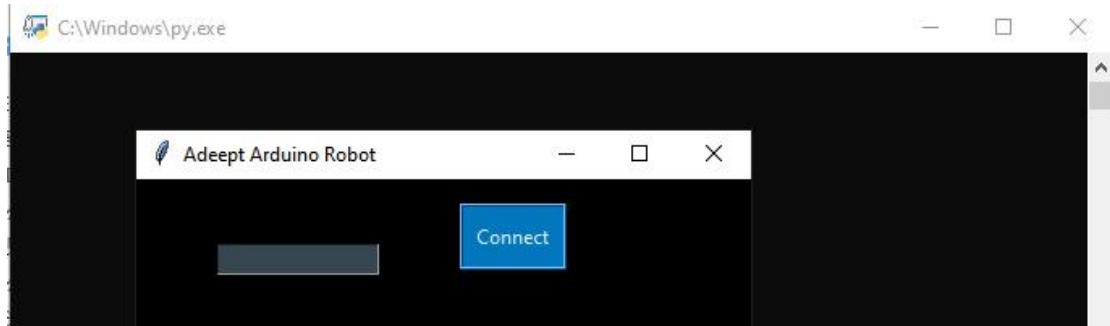
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

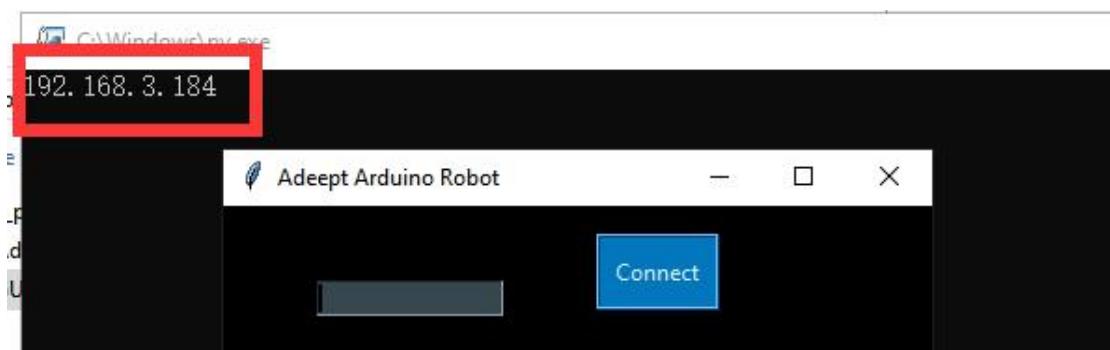
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

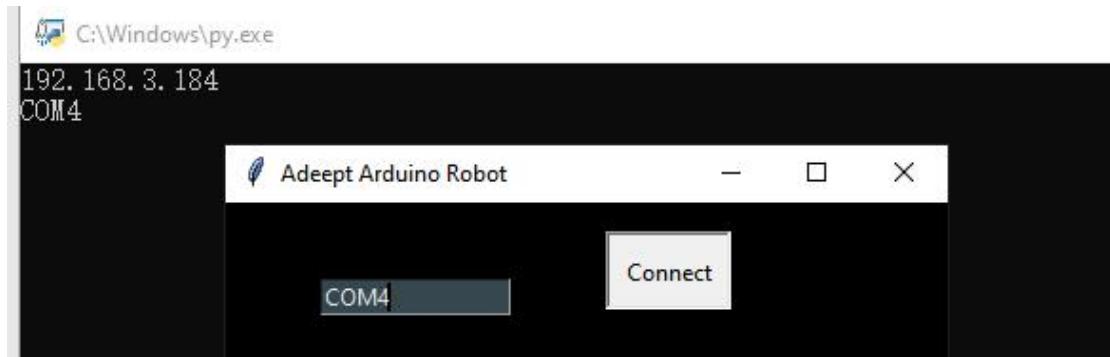
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



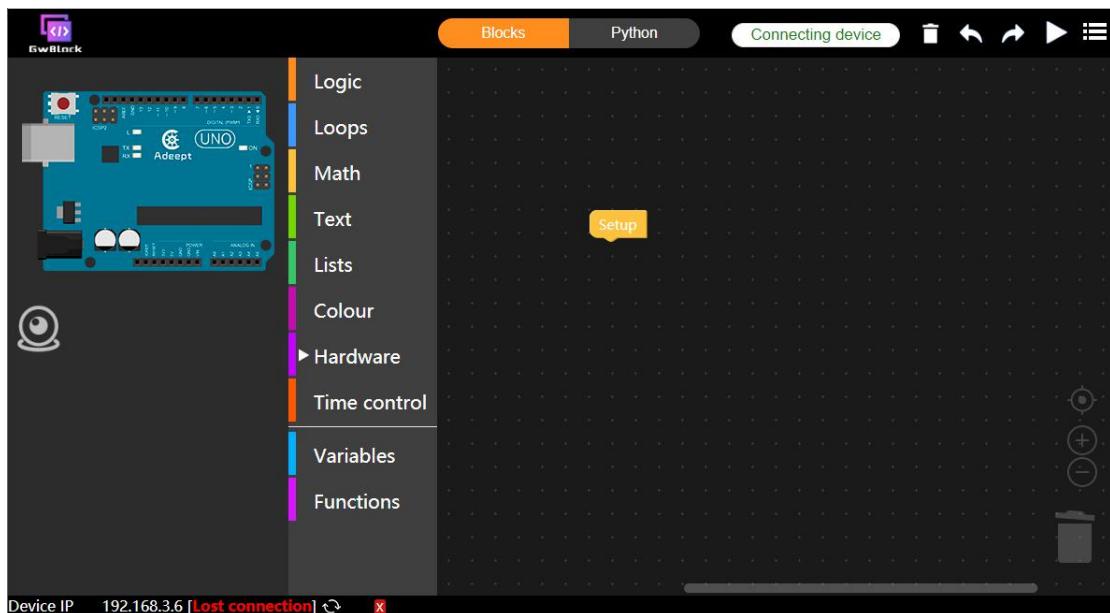
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



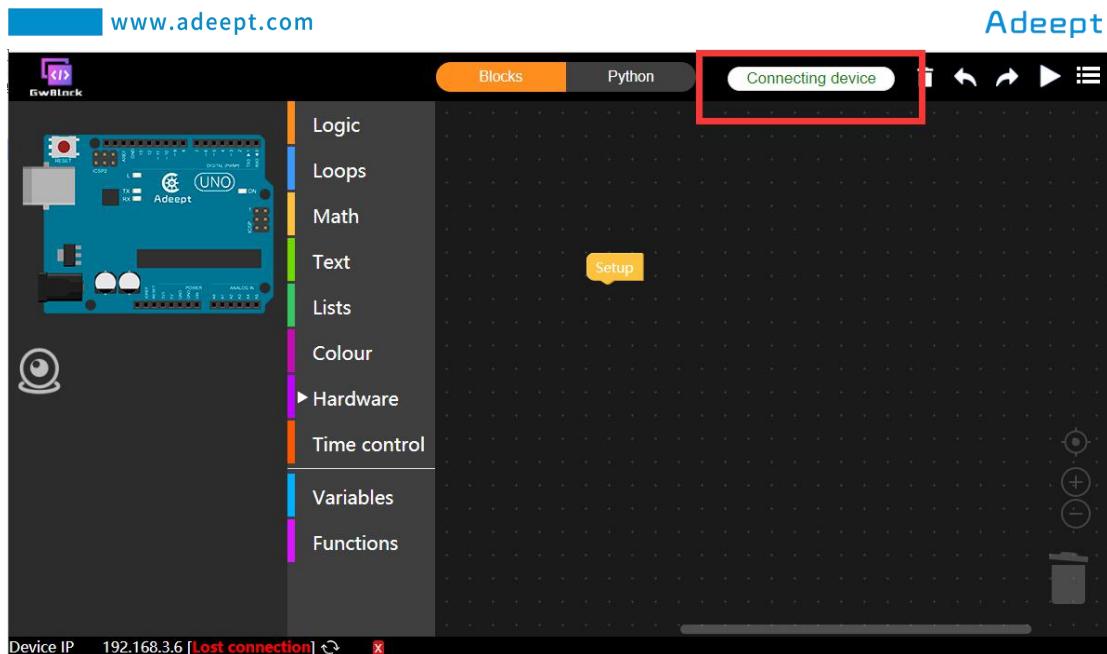
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

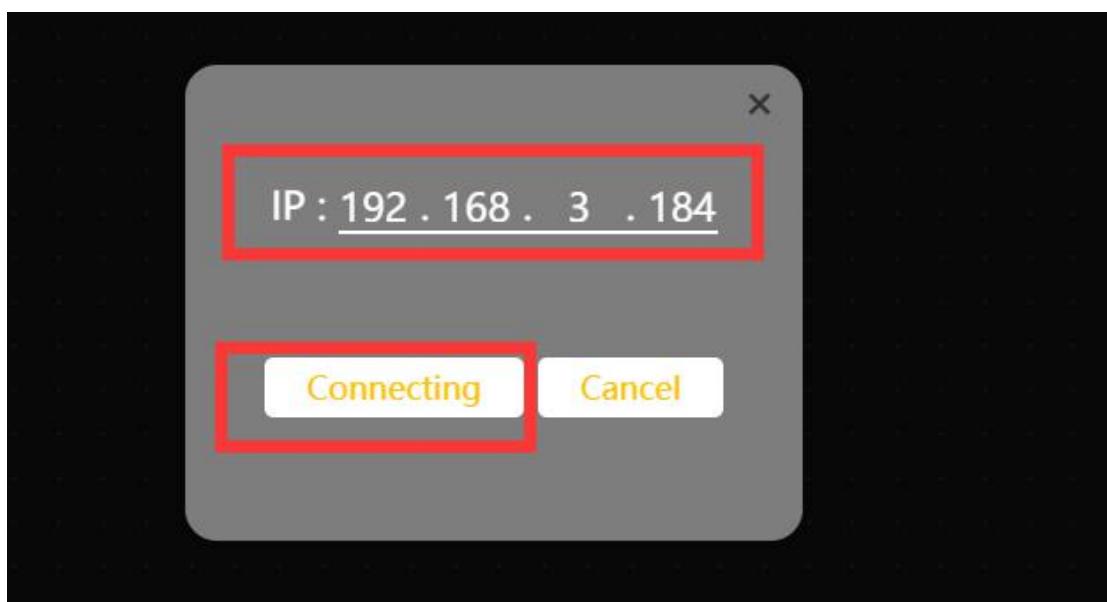
After successfully entering the website, the interface is as follows:



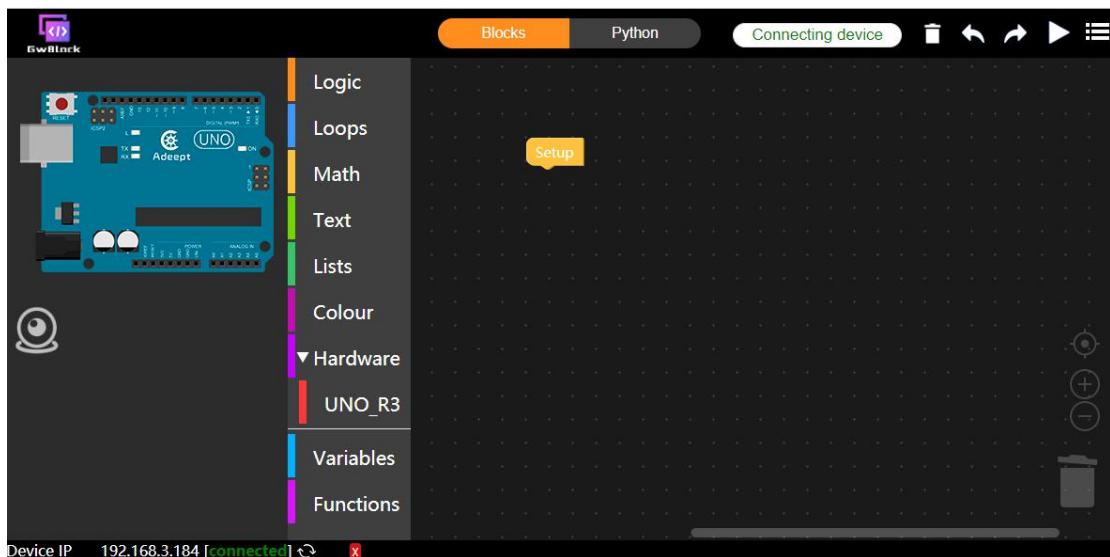
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

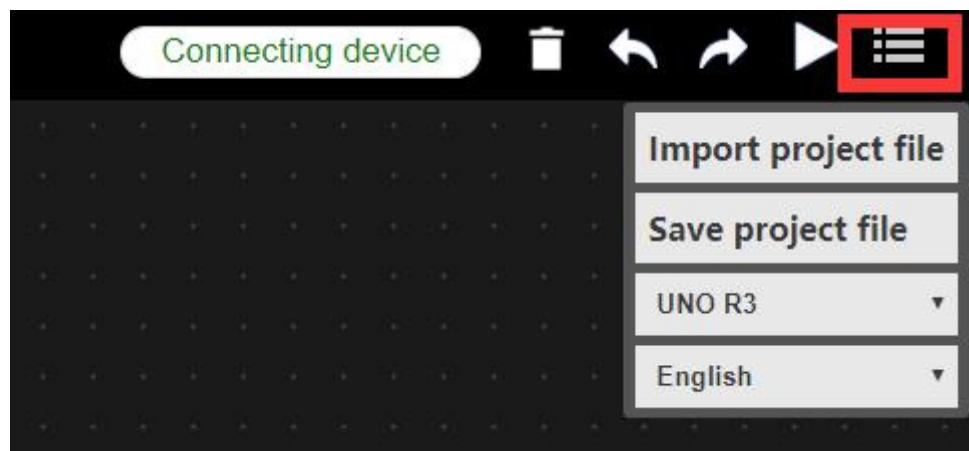


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

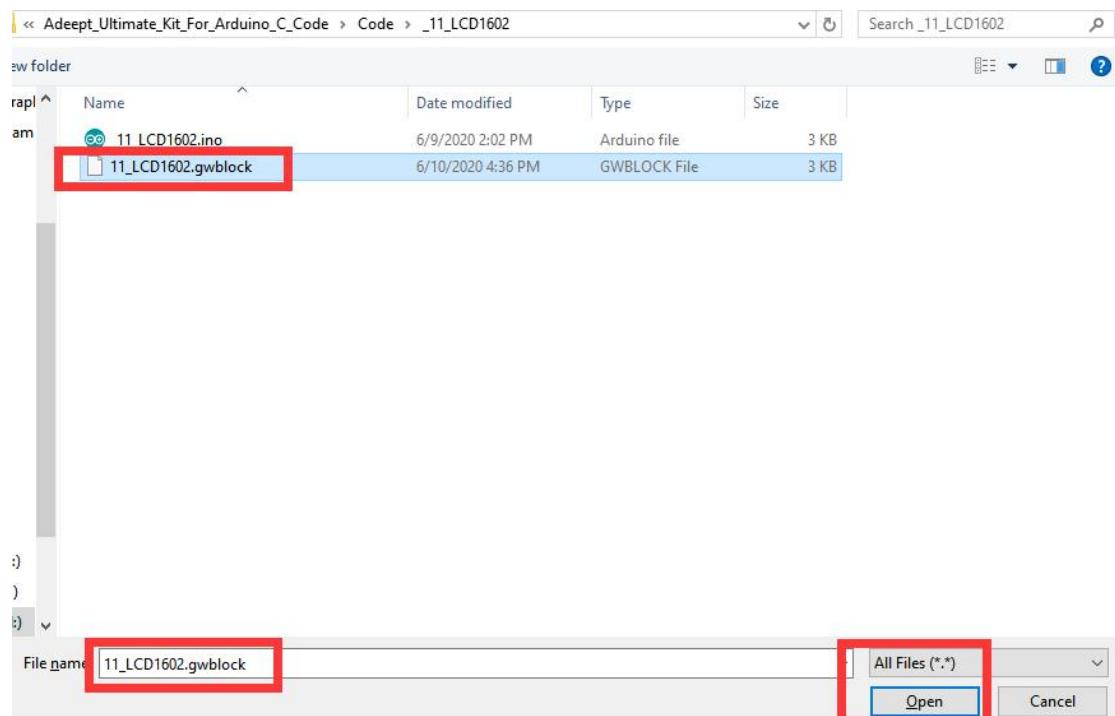
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_11_LCD1602

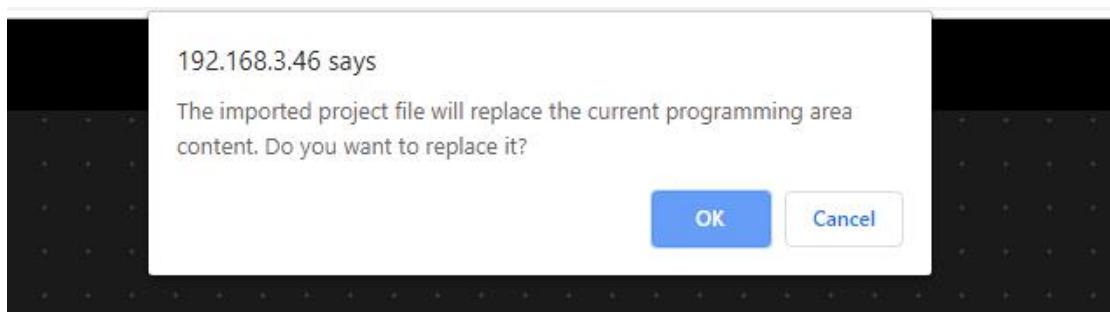
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

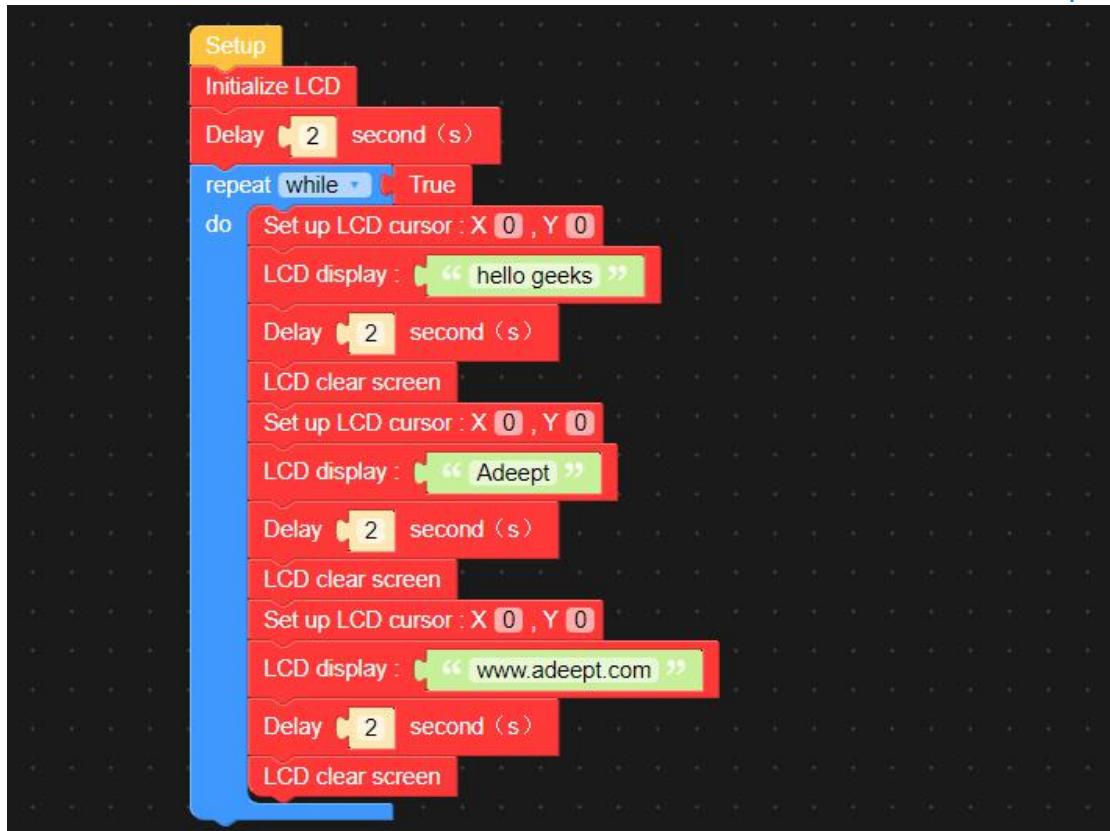
5. Select the "11_LCD1602.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



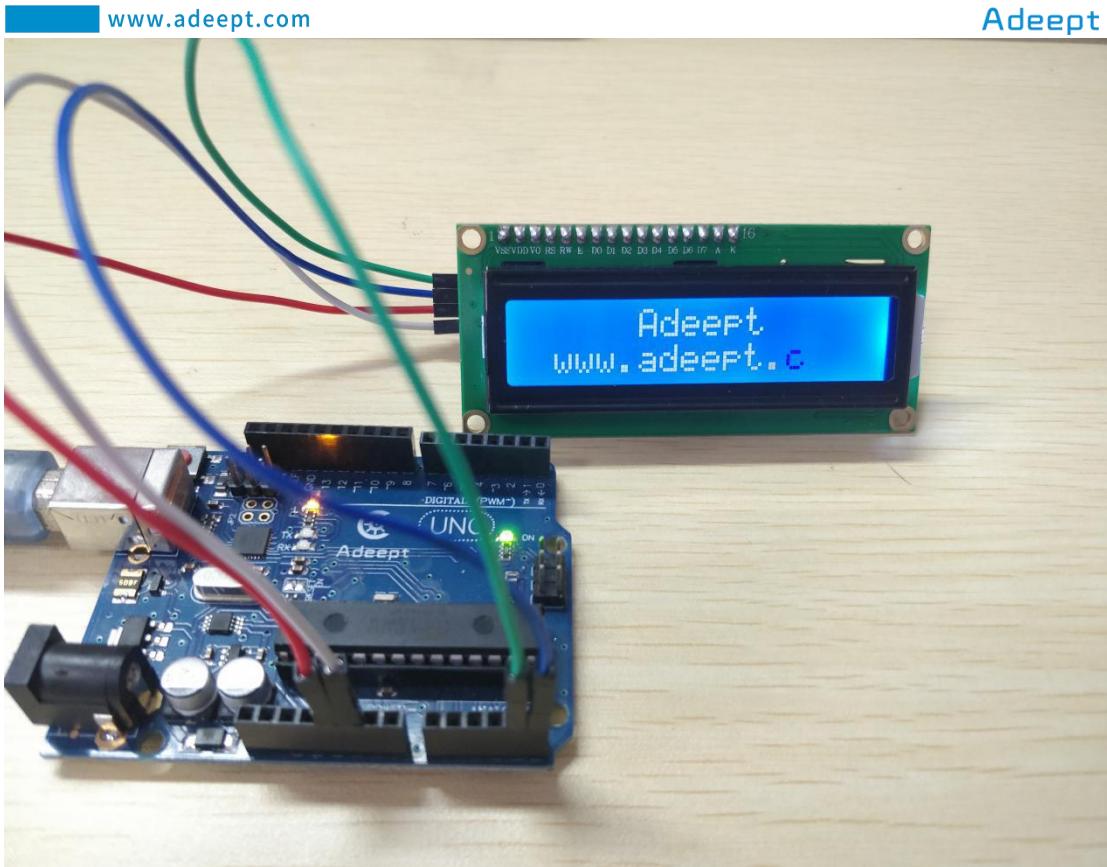
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



8. After the program runs successfully, hello geeks will be displayed on the LCD1602 screen first, and then www.adeept.com is displayed, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



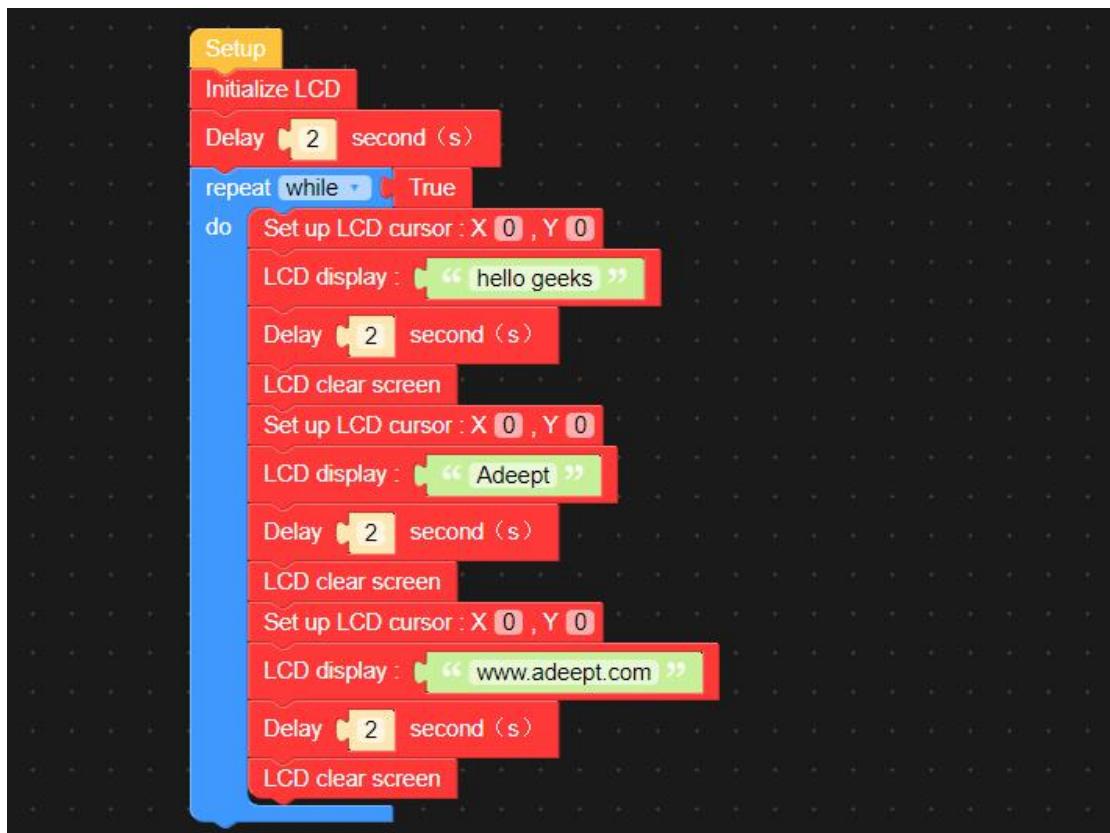
(3)Core code program

After the above hands-on operation, you must be very interested to know how we display characters on LCD1602 on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

In the GwBlock graphical editor, all code programs are executed from **Setup**. First initialize LCD through the instruction block **Initialize LCD**. In the while loop statement, the position of the text information to be displayed on the LCD1602 screen can be set through the instruction block **Set up LCD cursor : X 0 , Y 0**.

Using the instruction block **LCD display : "hello geeks"**, the text information "hello geeks" can be displayed on the LCD1602 screen. The text information in this instruction can be directly modified. You only need to enter the text information you need to display. The instruction block **Delay 2 second (s)** can be used to control the text information displayed on the LCD screen for 2s. Finally, the instruction block **LCD clear screen** needs to be used to clear the LCD1602, so that it can

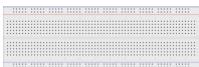
display “ Adeept ” at last.



Lesson 12 Making Voltmeter

In this lesson, we will learn how to make voltmeter with LCD1602, I2C and potentiometer.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
LCD1602	1	
Adeept IIC Module	1	
Male to Female Jumper Wires	Several	
Potentiometer	1	

2. The introduction of Voltmeter

In this lesson, we will make a simple voltmeter (0~5V) with Arduino UNO and LCD1602. Then, we will measure the voltage of the potentiometer(when adjusting the knob) with the simple voltmeter and display the voltage detected on the LCD1602.

(1)LCD1602

Please refer to Lesson 11, for we have introduced the LCD1602 in detail in Lesson

11.

(2)Potentiometer

Please refer to Lesson 5, for we have introduced the potentiometer in detail in Lesson 5.

(3)Experimental principle of making Voltmeter

The basic principle of this experiment: Convert the analog voltage collected from Arduino to digital quantity by the ADC (analog-to-digital converter) through programming, and then display the voltage on the LCD1602.

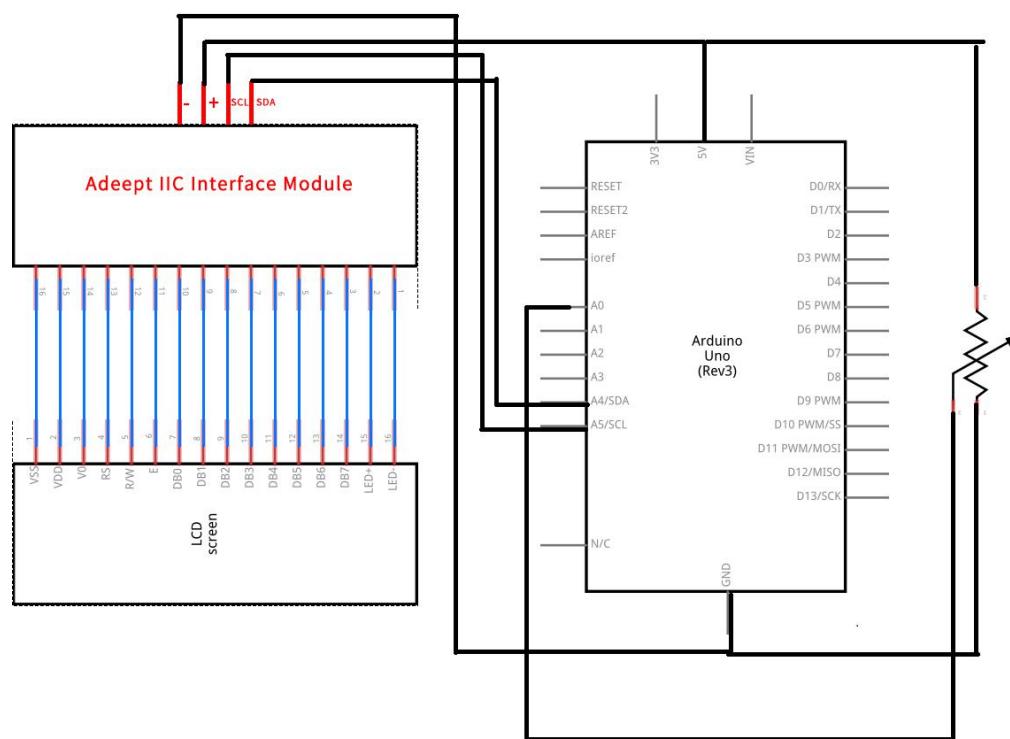
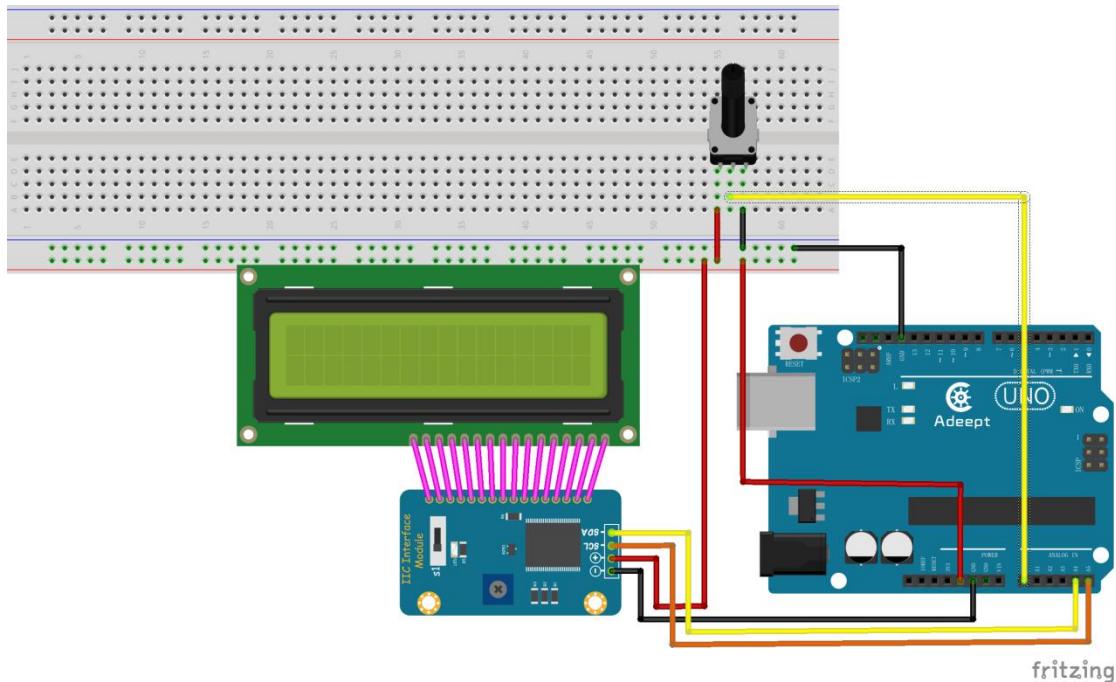
Connect the three wires from the potentiometer to your Arduino board. The first goes to ground from one of the outer pins of the potentiometer. The second goes from analog input 0 to the middle pin of the potentiometer. The third goes from 5V to the other outer pin of the potentiometer.

By turning the shaft of the potentiometer, you change the resistance on either side of the wiper which is connected to the center pin of the potentiometer. This changes the voltage at the center pin. When the resistance between the middle and the side one (connected to 5V) is close to zero (and the resistance on the other side is close to 10k Ohm), the voltage at the middle pin is close to 5V. When the resistances are reversed, the voltage at the center pin changes to about 0V, or ground. This voltage is the analog voltage that you're reading as an input.

The Arduino UNO board has a circuit inside called an analog-to-digital converter that reads this changing voltage and converts it to a number between 0 and 1023. When the shaft is turned all the way in one direction, there are 0 volts going to the pin, and the input value is 0. When the shaft is done in the opposite direction, it's 5 volts going to the pin and the input value is 1023. In between, `analogRead()` returns a number between 0 and 1023 that is proportional to the amount of voltage being applied to the pin.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:



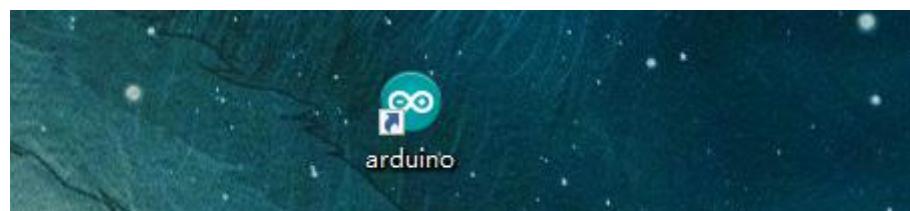
4.How to make Voltmeter

We provide two different methods to make Voltmeter. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

1.Using C language to program voltage measurement on Arduino UNO

(1)Compile and run the code program of this course

1.Open the Arduino IDE software, as shown below:

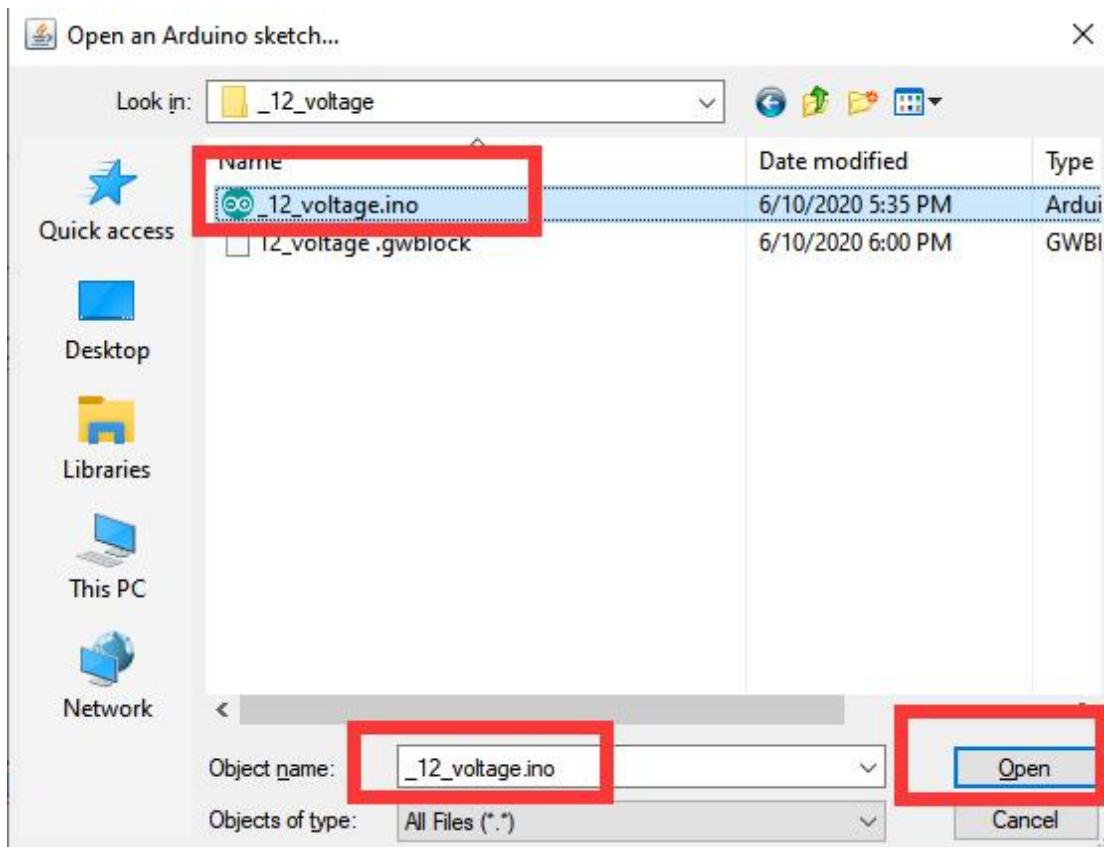


2.Click Open in the File drop-down menu:

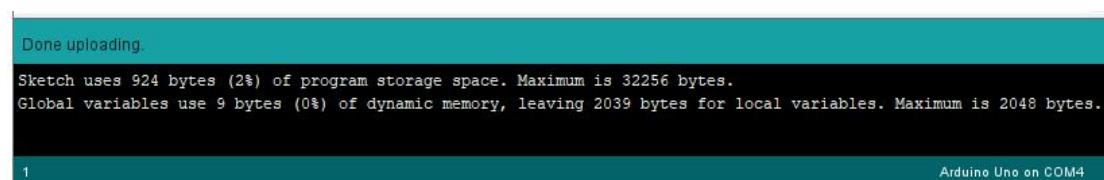


3.Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the

Code\12_voltage directory. Select 12_voltage.ino. This file is the code program we need in this course. Then click Open.

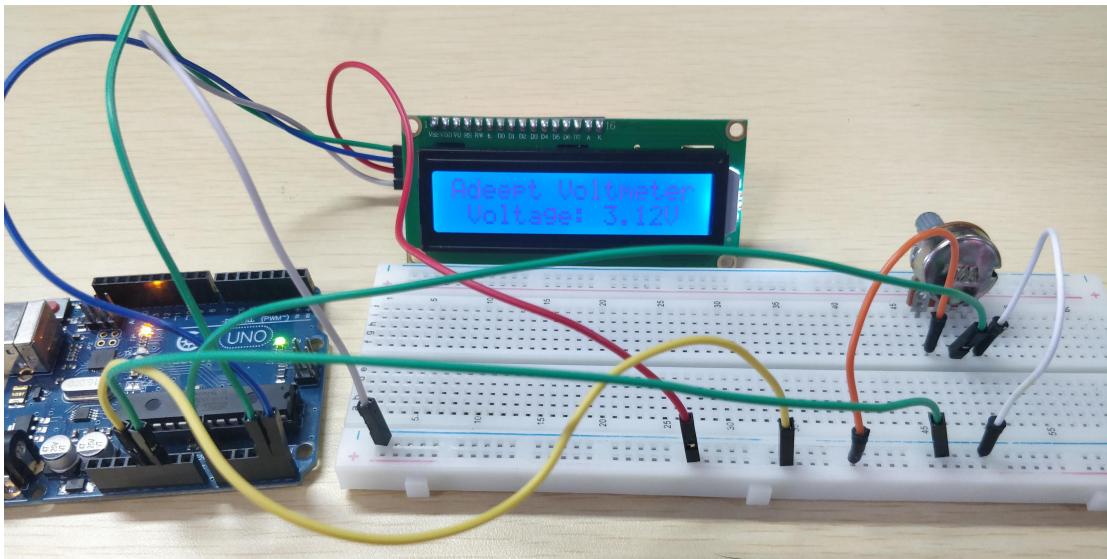


4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.



5. After running the program, rotate the potentiometer clockwise to increase the voltage to a maximum of about 5V; turn the potentiometer counterclockwise to decrease the voltage. The value of Voltage will be displayed on the LCD1602 screen.

The physical connection diagram of the experiment is as below:



【Pay Attention】

If the voltage number displayed on LCD1602 keeps changing during the experiment, it may be that the pin of the potentiometer connected to the breadboard is loose, you can have a try to reconnect it at other locations.

(2)The core code program

After the above hands-on operation, you must be very interested to know how we program to display the voltage value on LCD1602 on Arduino UNO. We will introduce how our core code can be achieved:

1.In the setup() method, the text information in the corresponding array is displayed on the LCD1602 screen through the lcd.print() function in the for loop statement.

```
void setup()
{
    Serial.begin(115200);
    lcd.init(); //initialize the lcd
    lcd.backlight(); //turn on the backlight
    lcd.setCursor(0,0); // set the cursor to column 15, line 0
    for (int positionCounter1 = 0; positionCounter1 < 16; positionCounter1++)
    {
        lcd.print(array1[positionCounter1]); // Print a message to the LCD.
        delay(250); //wait for 250 microseconds
    }
}
```

2.In the loop() function, read the value of the potentiometer through the analogRead(photoresistorPin) function. In the for loop statement, display the voltage value on the LCD1602 screen through the lcd.print() function.

```

void loop()
{
    val = analogRead(photoresistorPin)*(5.0 / 1023.0)*100;//Data collection
    array2[10] = val/100%10+0x30;//Take one hundred - bit data
    array2[12] = val/10%10+0x30; //Take ten-bit data
    array2[13] = val%10+0x30; //Take a bit of data
    lcd.setCursor(0,1); //Set the cursor to column 15, line 1
    for (int positionCounter3 = 0; positionCounter3 < 26; positionCounter3++)
    {
        lcd.print(array2[positionCounter3]); // Print a message to the LCD.
        delay(250); //Wait for 250 microseconds
    }
}

```

2. Using graphical code blocks to program voltage measurement on Arduino UNO

(1) Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

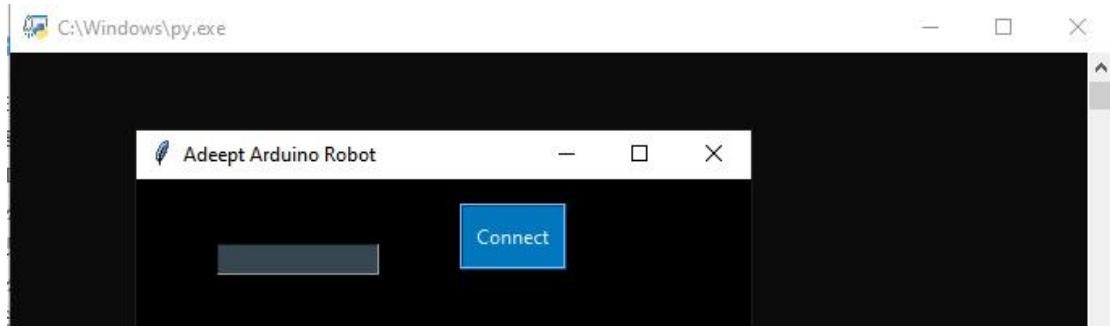
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

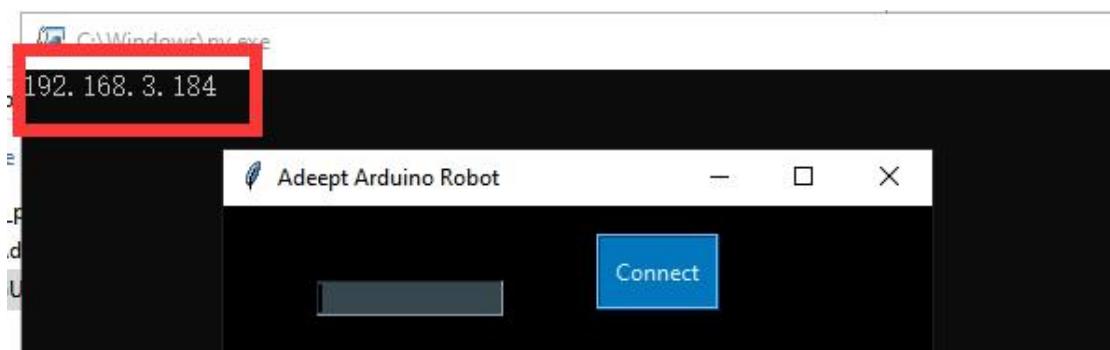
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

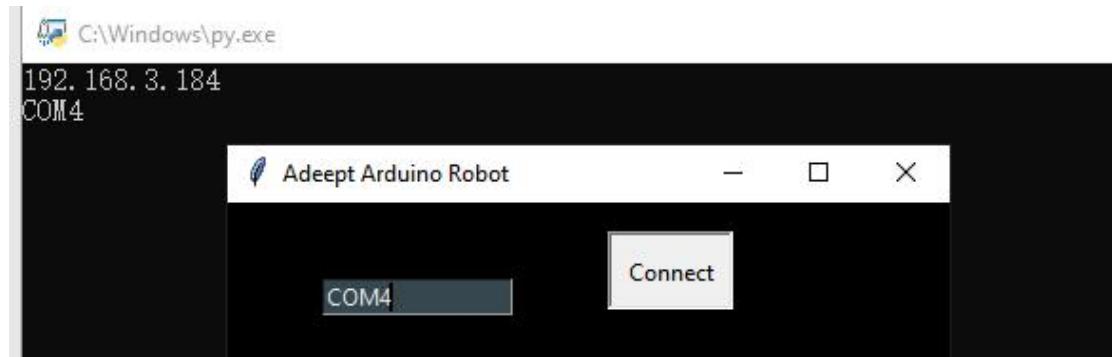
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



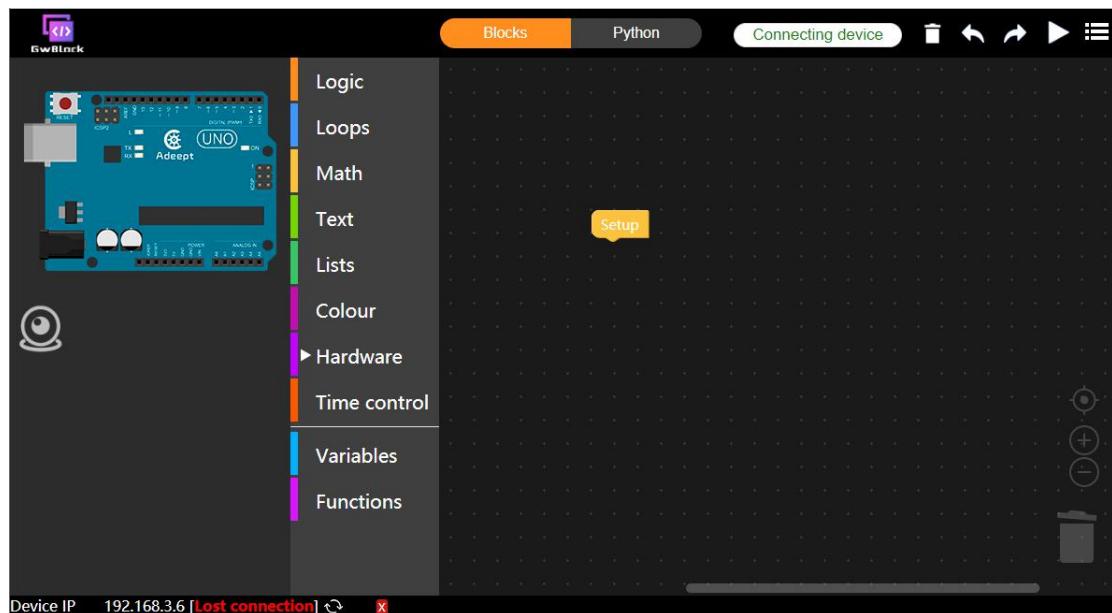
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



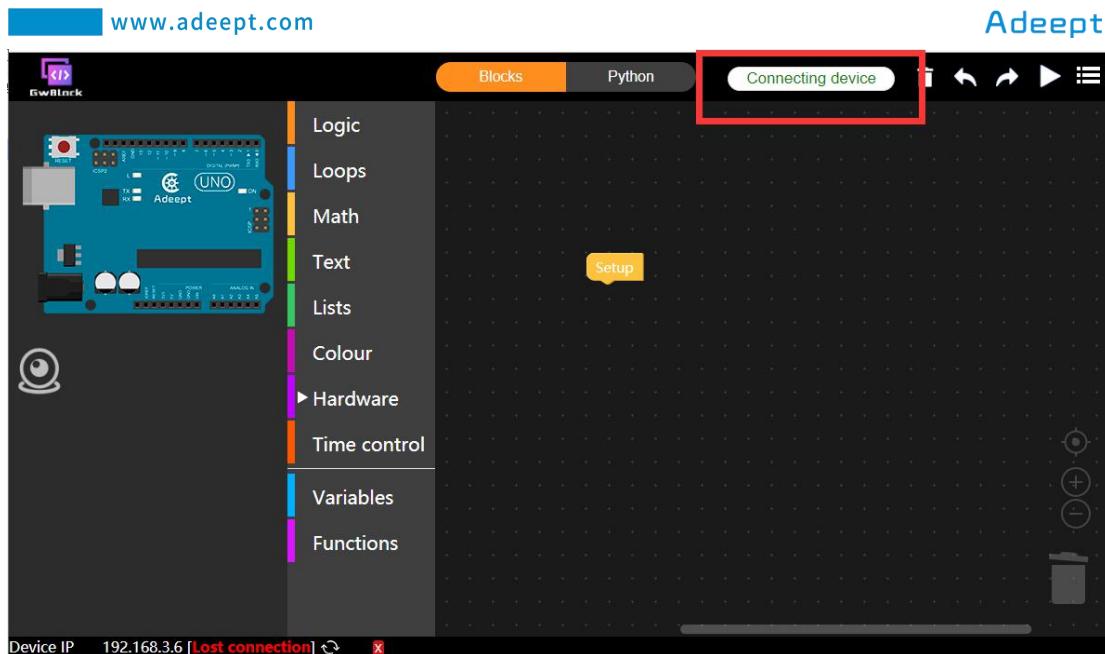
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

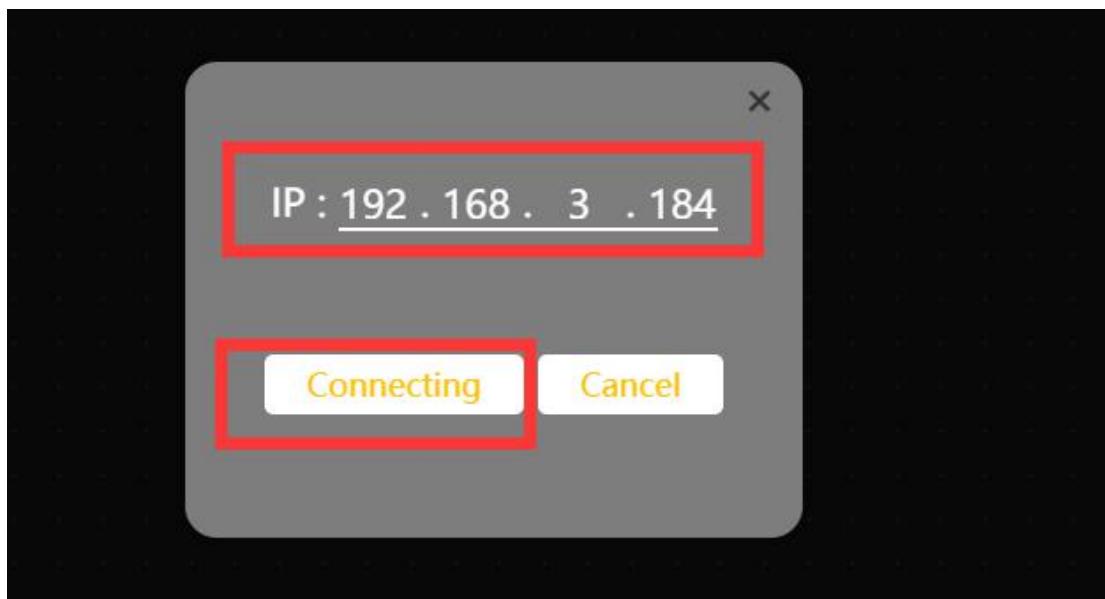
After successfully entering the website, the interface is as follows:



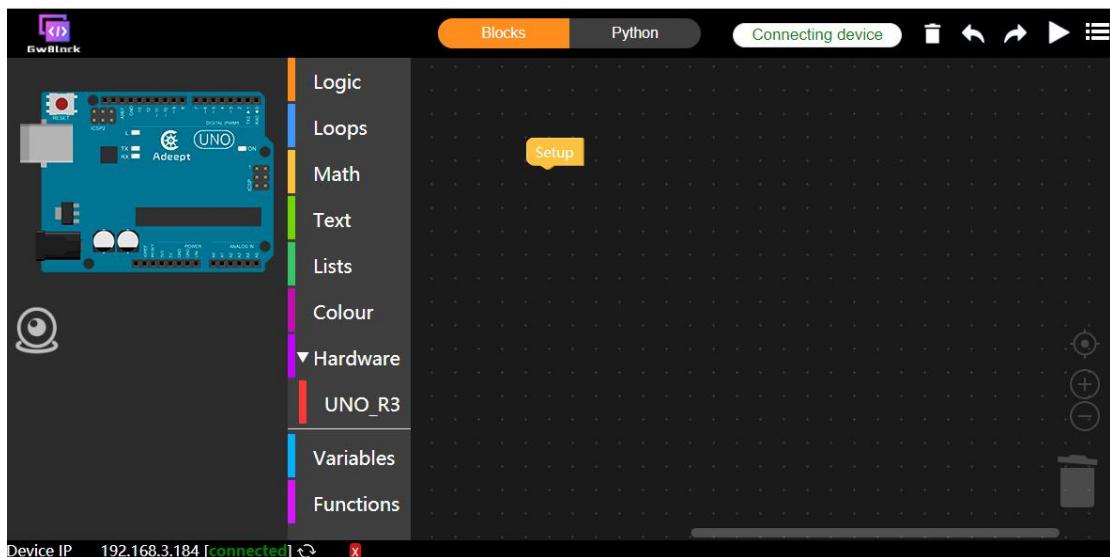
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

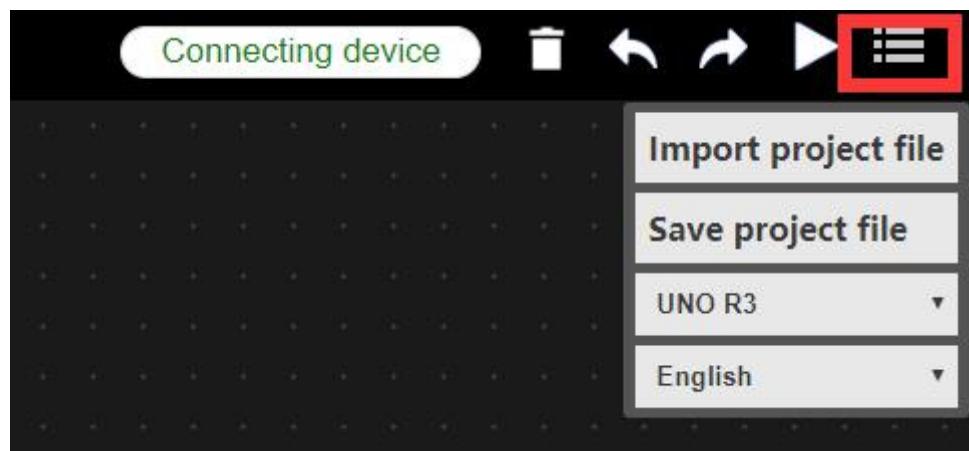


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3.The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

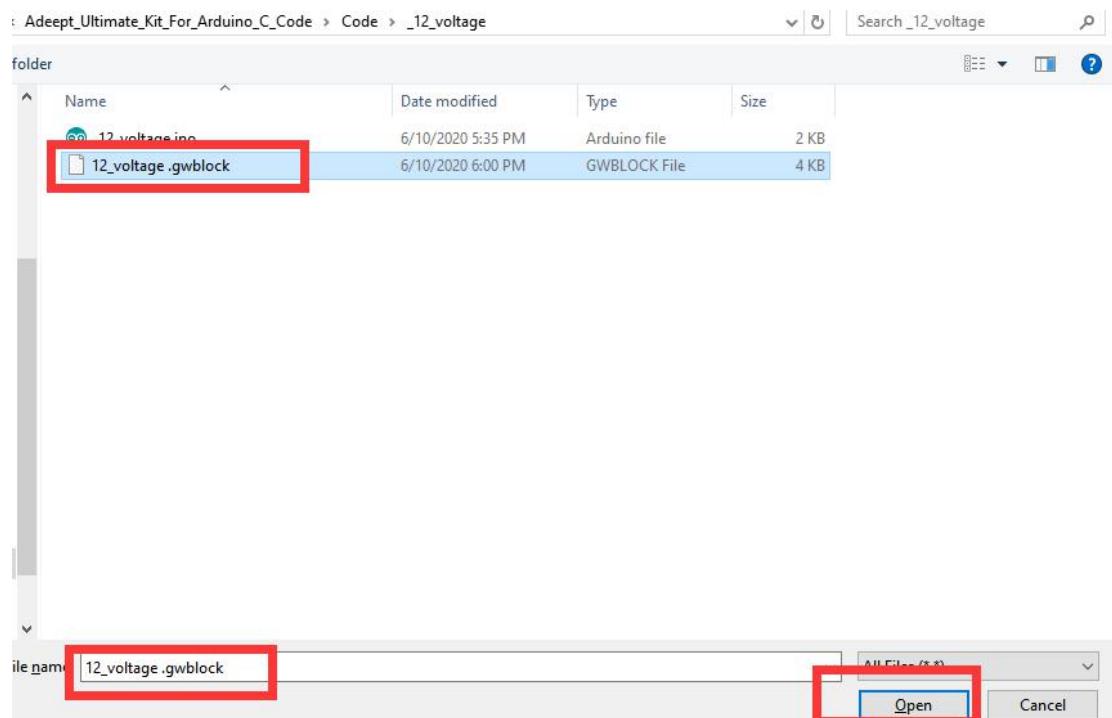
4.Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_12_voltage

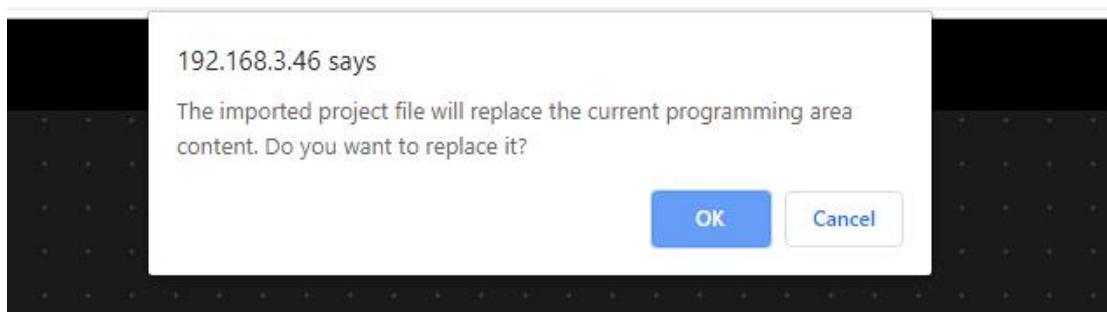
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

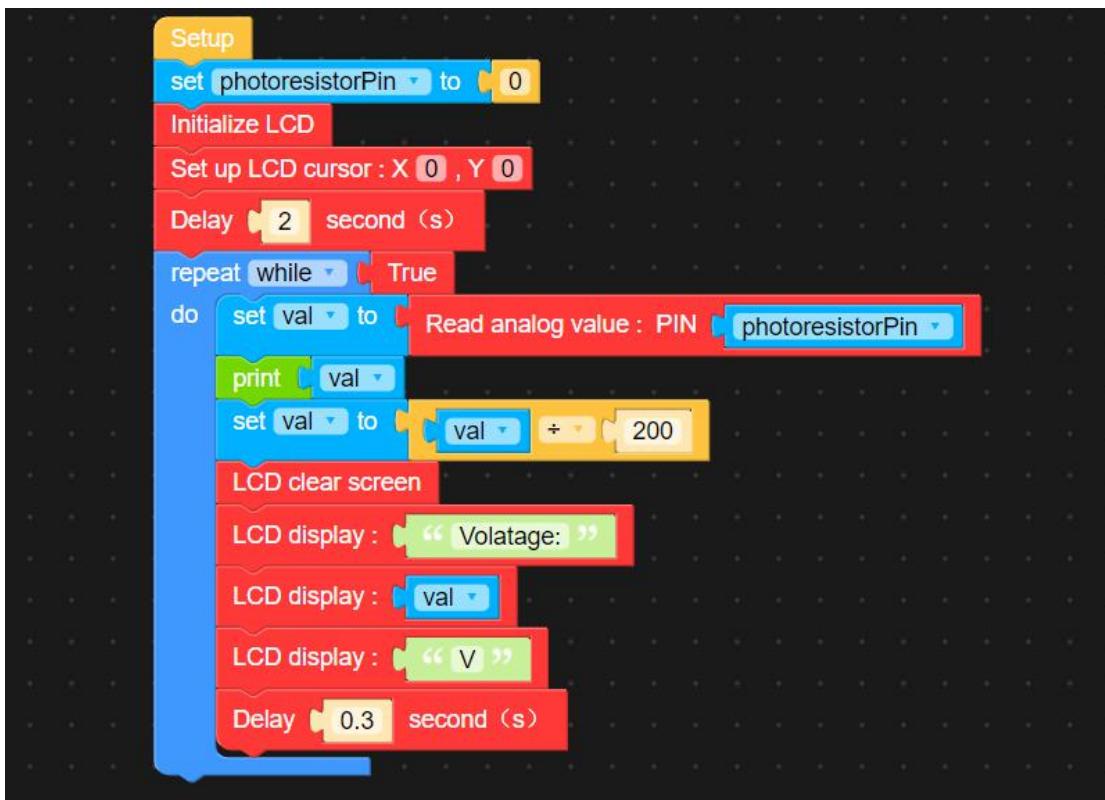
5. Select the "12_voltage.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



6.Click OK.It will show as below:

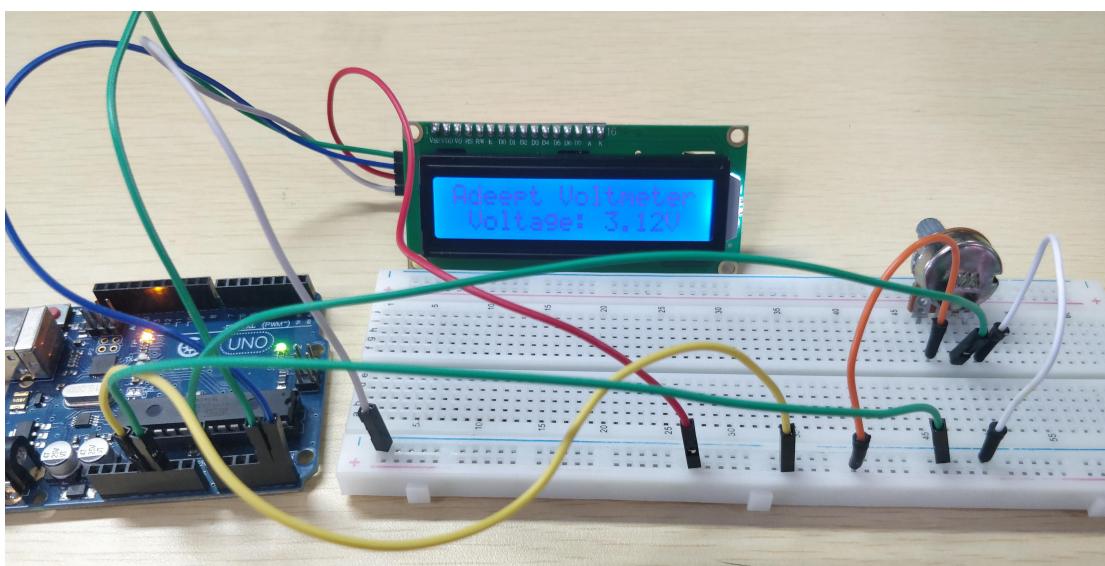


7.It will show as below after successfully opening:



8. After running the program, rotate the potentiometer clockwise to increase the voltage to a maximum of about 5V; turn the potentiometer counterclockwise to decrease the voltage. The value of Voltage will be displayed on the LCD1602 screen.

The physical connection diagram of the experiment is as below:



【Pay Attention】

If the voltage number displayed on LCD1602 keeps changing during the

experiment, it may be that the pin of the potentiometer connected to the breadboard is loose, you can have a try to reconnect it at other locations.

(3)Core code program

After the above hands-on operation, you must be very interested to know how we program voltage measurement on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

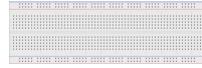
In the GwBlock graphical editor, all code programs are executed from **Setup**. First initialize the pin number of photoresistorPin of potentiometer to 0 through the instruction block **set [photoresistorPin] to 0**. In the while loop statement, the value of the potentiometer is read through the instruction block **Read analog value : PIN [photoresistorPin]**. Convert the read value through instruction block **set [val] to [val] + [200]**. Finally display the voltage value on the LCD1602 screen through the instruction block **LCD display : [val]**.



Lesson 13 The Application of the 7-segment Display

In this lesson, we will learn the application of the 7-segment Display.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	1	
7-segment Display	1	

2. The introduction of the 7-segment Display

(1)The 7-segment Display

The 7-segment display is also called the 7-segment LED display. It is a type of LED display, using 7 LED lights to form the font "8", and another dot LED to display the decimal point, which means that there are 8 LED lights in total to form the font of "8."

According to the connection form, it can be divided into common anode display and common cathode display. By controlling the display mode of semiconductor light emitting diodes, LED display panel (LED panel) is used to display information such as text, graphics, images, animations, quotes, videos, video signals, etc. The 7-segment display module is used to display numbers from decimal 0 to 9 and

decimal point, and can also display English letters, including English A to F in hexadecimal (b and d are lowercase, others are uppercase)

(2)Working principle of the 7-segment Display

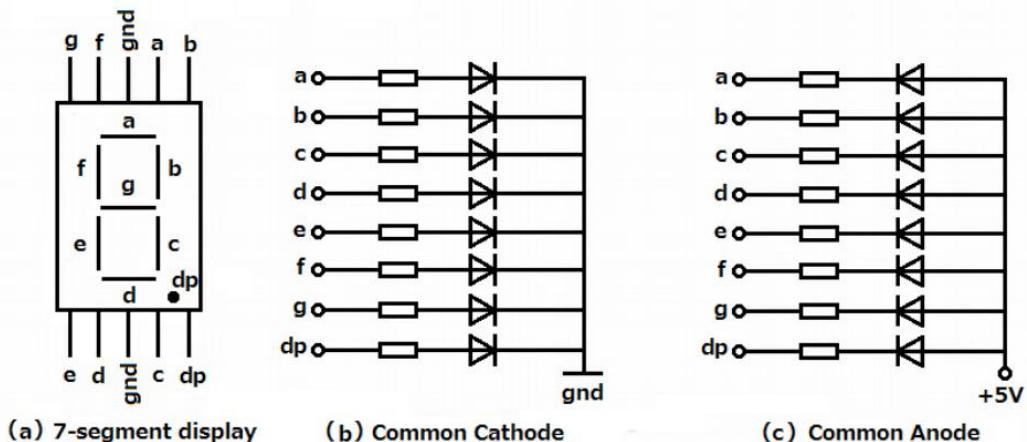
The 7-segment display module is packaged by multiple LED light-emitting diodes. Each LED is called a segment. In addition to the seven-segment strokes necessary for displaying numbers, a decimal point is also provided in the display. It has 8 pins with numbers from a to g. By forward installing the appropriate pins of the LED segments in a specific order, some segments will be bright and others will be dark, allowing the desired character pattern of the numbers generated on the display. Then each of the ten decimal digits 0 to 9 can be displayed on the same 7-segment display.

The seven-segment display is an 8-shaped LED display device composed of eight LEDs (including a decimal point). The segments respectively named a, b, c, d, e, f, g, and dp.

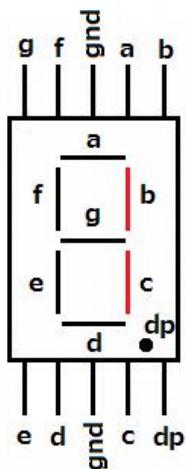
The segment display can be divided into common anode and common cathode segment display by internal connections.

When using a common anode LED, the common anode should be connected to the power supply (VCC); when using a common cathode LED, the common cathode should be connected to the ground (GND).

Each segment of a segment display is composed of LED, so a resistor is needed for protecting the LED.

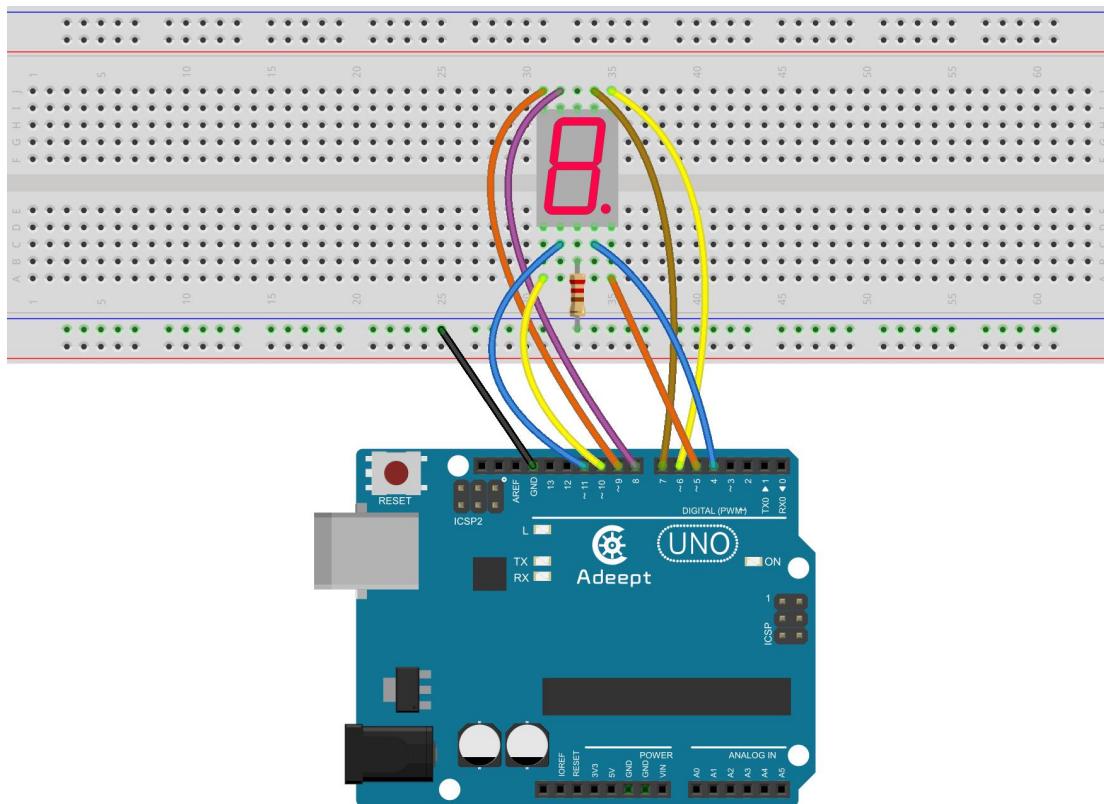
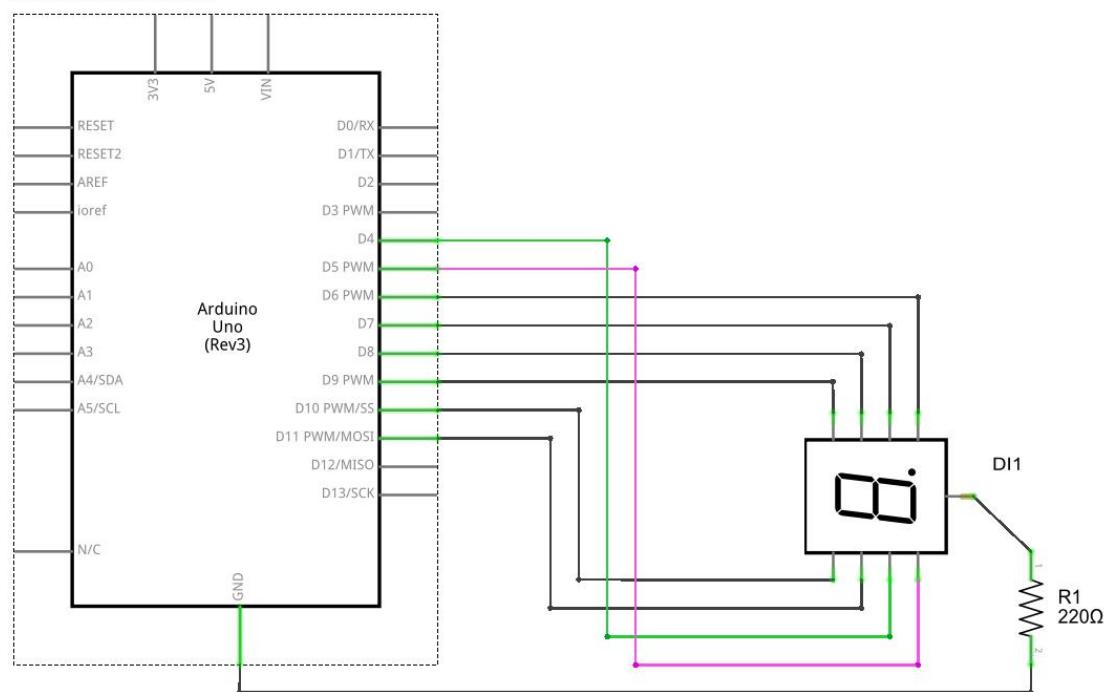


A 7-segment display has seven segments for displaying a figure and one more for displaying a decimal point. For example, if you want to display a number '1', you should only light the segment b and c, as shown below.



3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:


AdeeptLesson13


4. The Application of the 7-segment Display

We provide two different methods to control the 7-segment Display. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

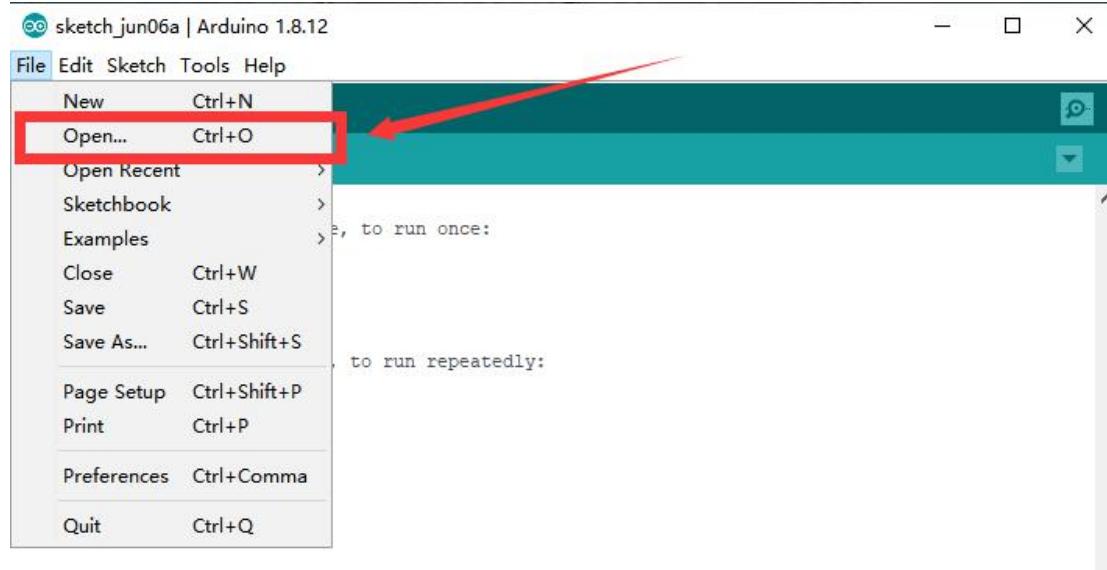
1.Using C language to program to control the 7-segment Display on Arduino UNO

(1)Compile and run the code program of this course

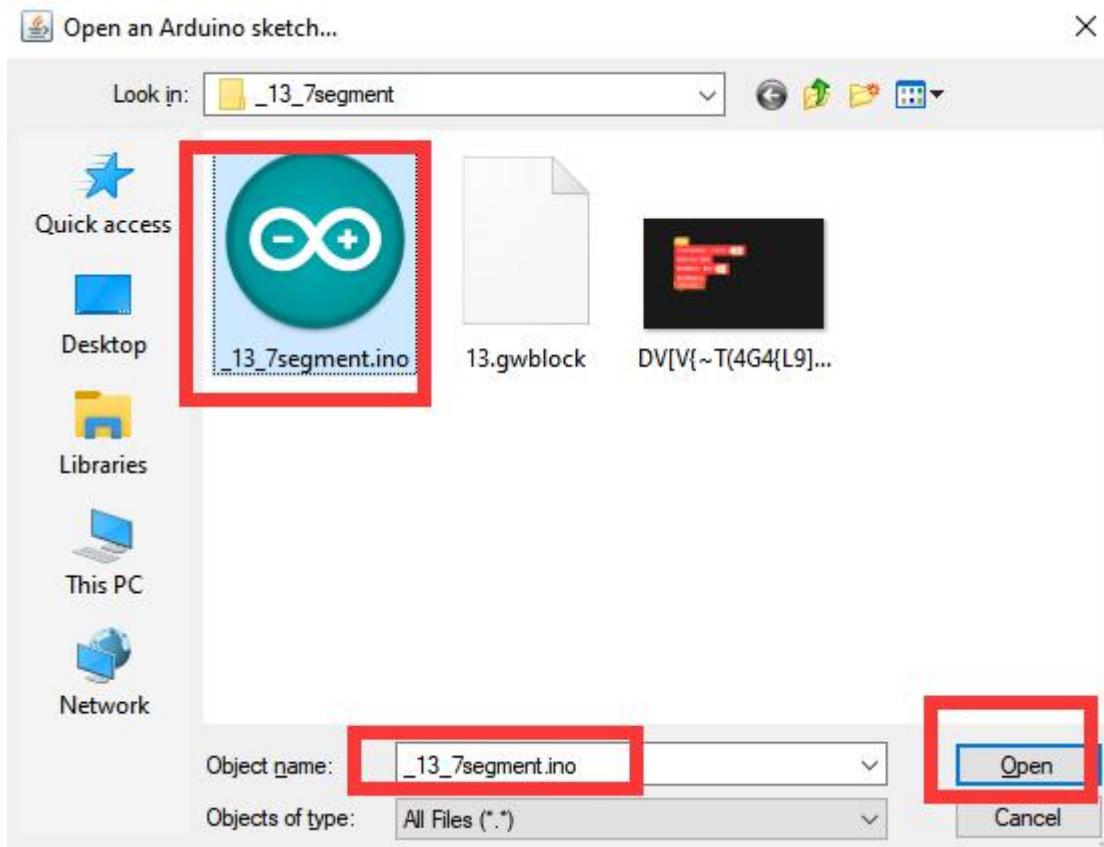
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\13_7segment directory. Select _13_7segment.ino. This file is the code program we need in this course. Then click Open.



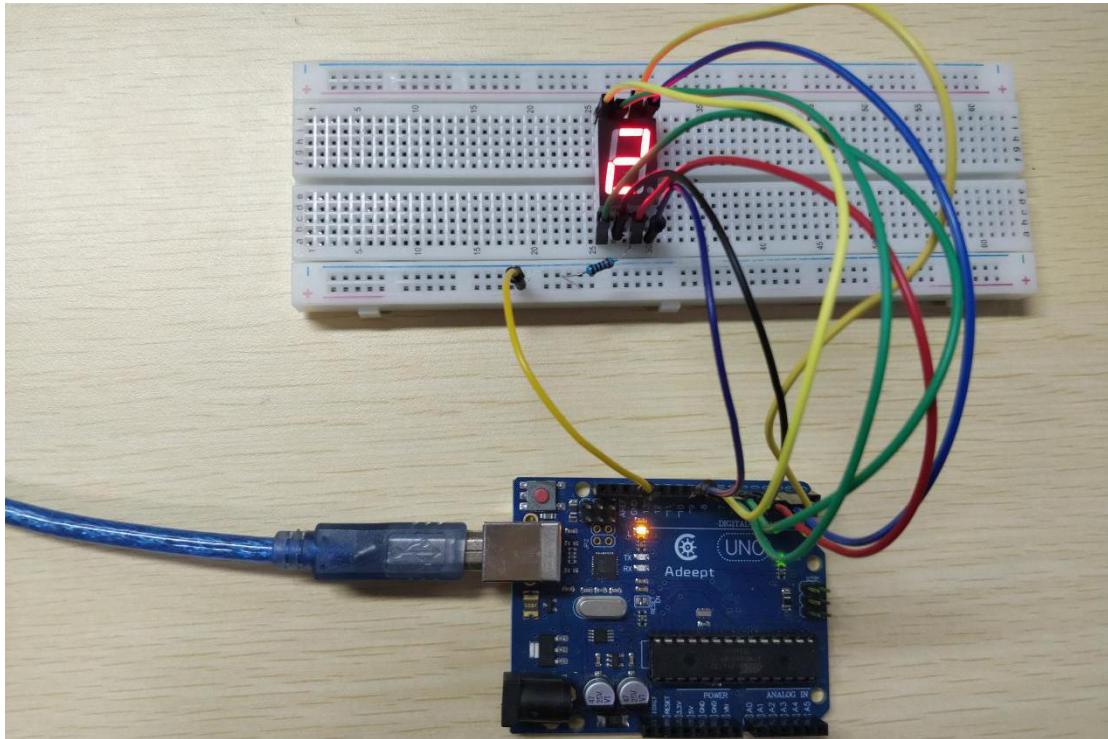
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After successfully running the program, we observe the digital tube and will find that the number changes from 0 to 9, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



(2)The core code program

After the above hands-on operation, you must be very interested to know how we program to control the the 7-segment Display on Arduino UNO.We will introduce how our core code can be achieved:

1.In the setup() function, set the digital pins of the 7-segment digital tube connected to the Arduino UNO to OUTPUT mode in turn through pinMode(i, OUTPUT) in the for loop statement.

```
void setup()
{
    int i;           //Define variables
    for(i=2;i<=11;i++)
        pinMode(i,OUTPUT); //Set up 4 ~ 11 pins to output mode
}
```

2.In the loop() function, set the 7-segment digital tube through the digital() function in turn to display the numbers from 0 to 9.

```

void loop()
{
    digital_0(); //Segment display digital 0
    delay(1000); //Delay 1s
    digital_1(); //Segment display digital 1
    delay(1000); //Delay 1s
    digital_2(); //Segment display digital 2
    delay(1000); //Delay 1s
    digital_3(); //Segment display digital 3
    delay(1000); //Delay 1s
    digital_4(); //Segment display digital 4
    delay(1000); //Delay 1s
    digital_5(); //Segment display digital 5
    delay(1000); //Delay 1s
    digital_6(); //Segment display digital 6
    delay(1000); //Delay 1s
    digital_7(); //Segment display digital 7
    delay(1000); //Delay 1s
    digital_8(); //Segment display digital 8
    delay(1000); //Delay 1s
    digital_9(); //Segment display digital 8
    delay(1000); //Delay 1s
}

void digital_0(void) //Segment display digital 0
{
    digitalWrite(a,HIGH);
    digitalWrite(b,HIGH);
    digitalWrite(c,HIGH);
    digitalWrite(d,HIGH);
    digitalWrite(e,HIGH);
    digitalWrite(f,HIGH);
    digitalWrite(g, LOW);
    digitalWrite(dp,LOW);
}
void digital_1(void) //Segment display digital 1
{
    digitalWrite(a,LOW);
    digitalWrite(b,HIGH);
    digitalWrite(c,HIGH);
    digitalWrite(d,LOW);
    digitalWrite(e,LOW);
    digitalWrite(f,LOW);
    digitalWrite(g,LOW);
    digitalWrite(dp,LOW);
}

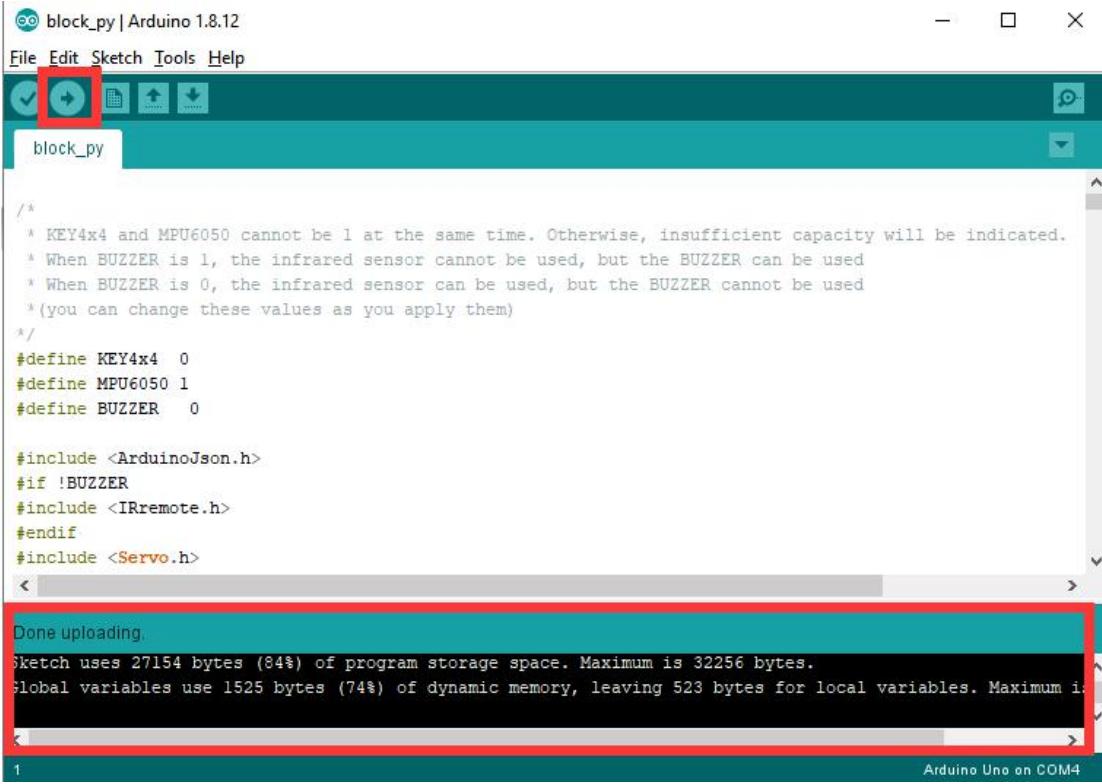
```

2. Using graphical code blocks to program to control the 7-segment Display on Arduino UNO

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1.Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:



```

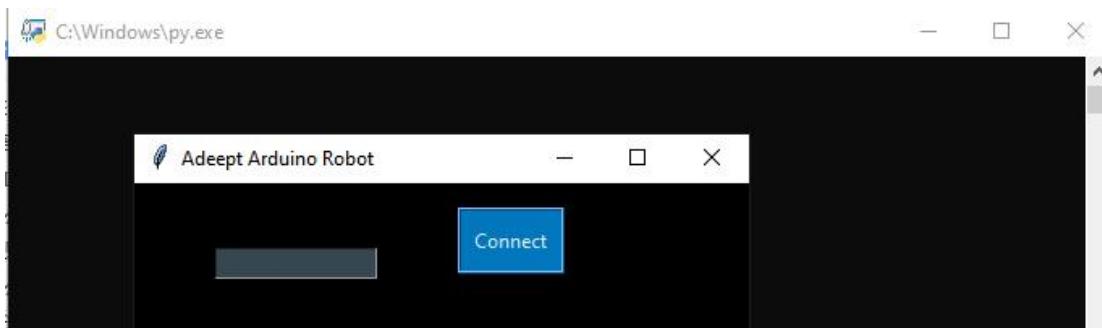
block_py | Arduino 1.8.12
File Edit Sketch Tools Help
block_py
/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<

Done uploading.
Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum i:
<
1
Arduino Uno on COM4

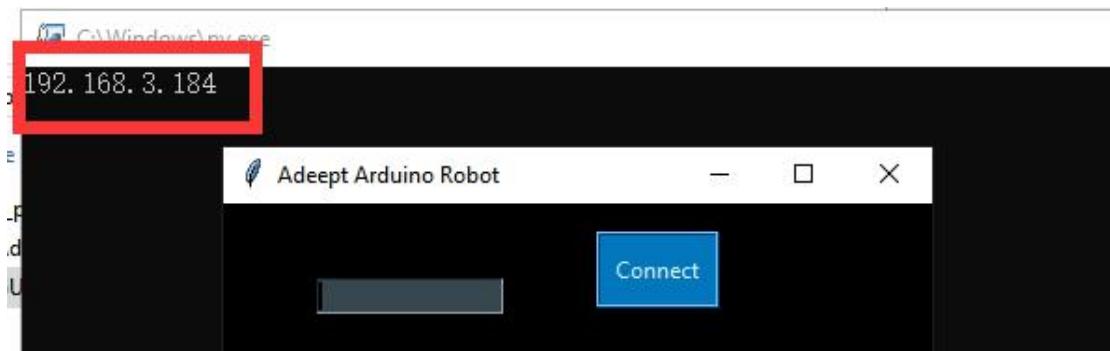
```

2.Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again.Then open the websocket folder. Double-click to open the GUI info v1.0.py file.It will show as below:



3.Remember the IP address: 192.168.3.184 (each user's IP is different) which will be

used later. It will show as below:



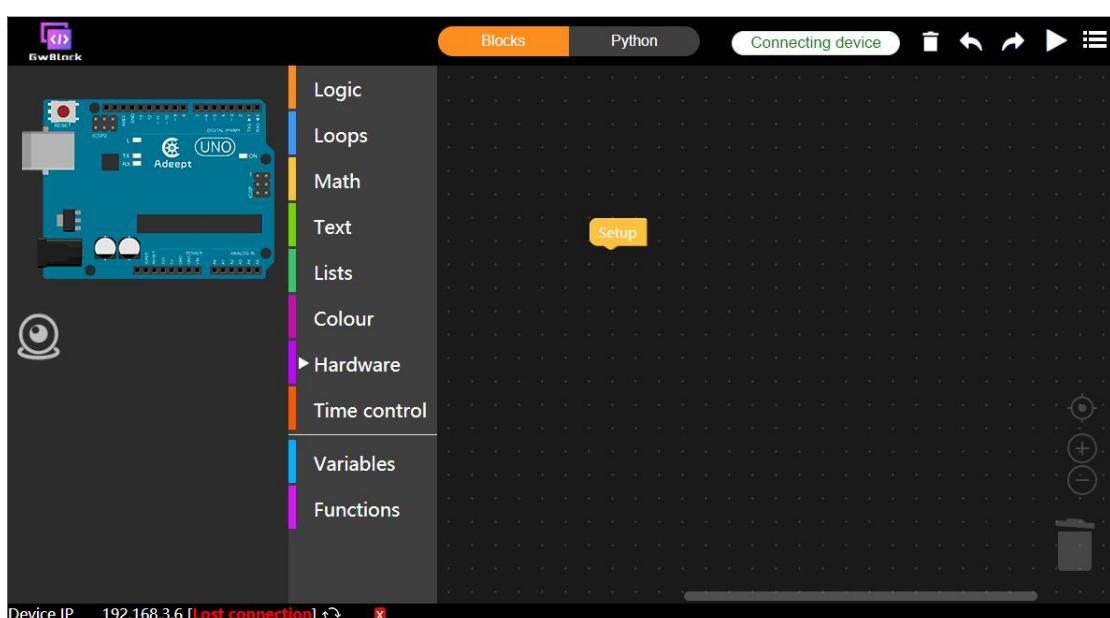
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



5.Enter the URL of the GwBlock graphical editor in the browser:

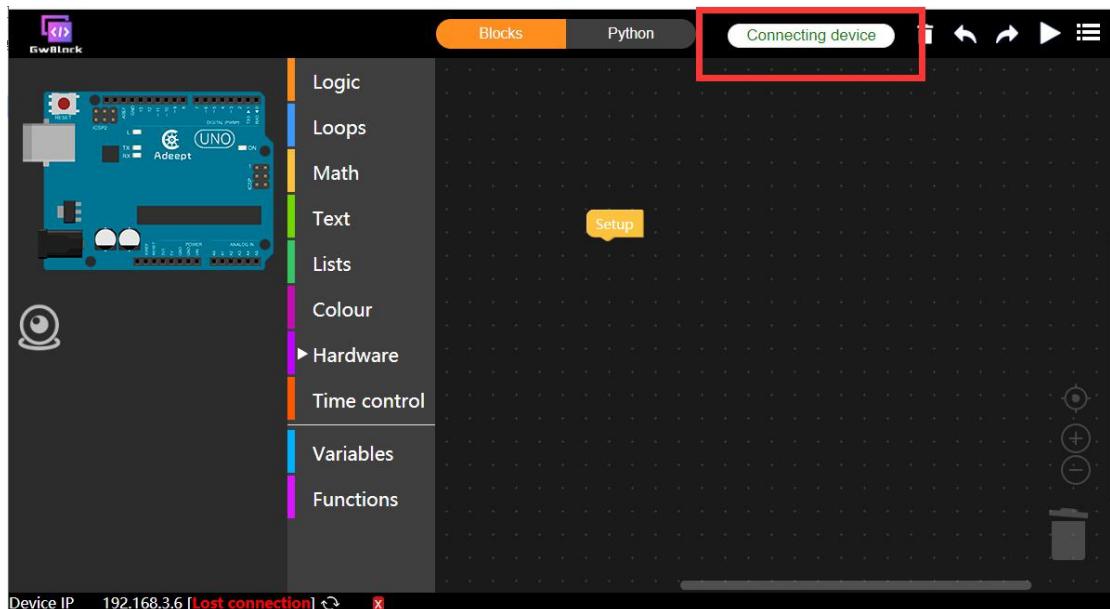
http://www.adeept.com/gwblock/?hd_mo=uno_r3.

After successfully entering the website, the interface is as follows:

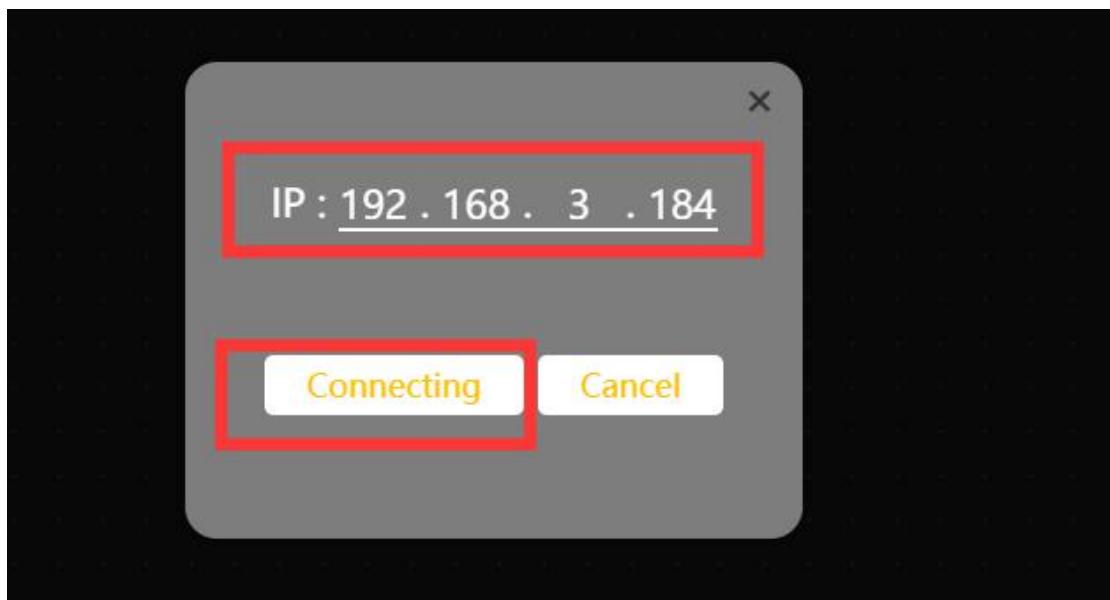


6.Click the "Connecting device" button in the upper right corner. It will show as

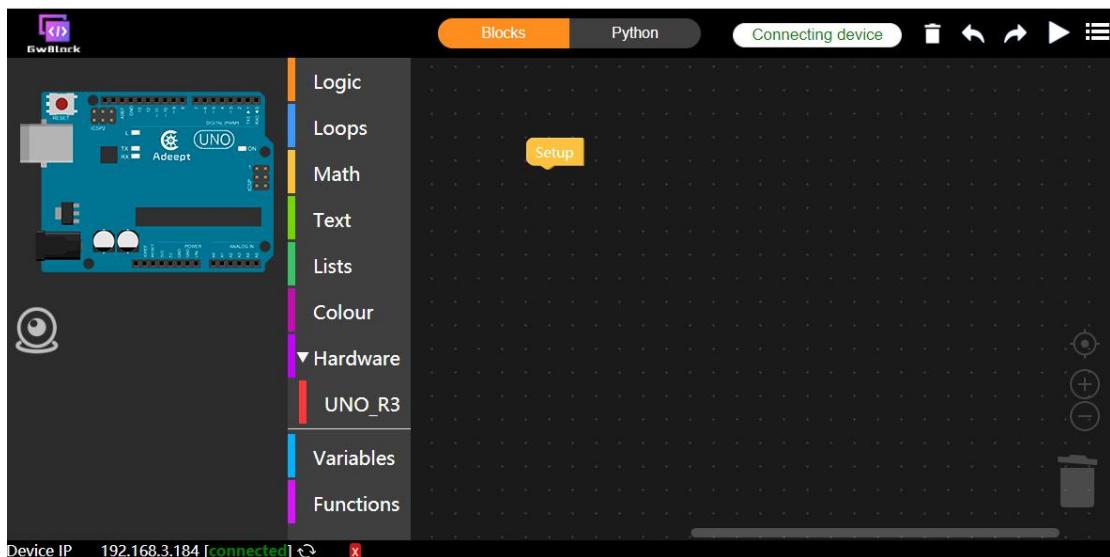
below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

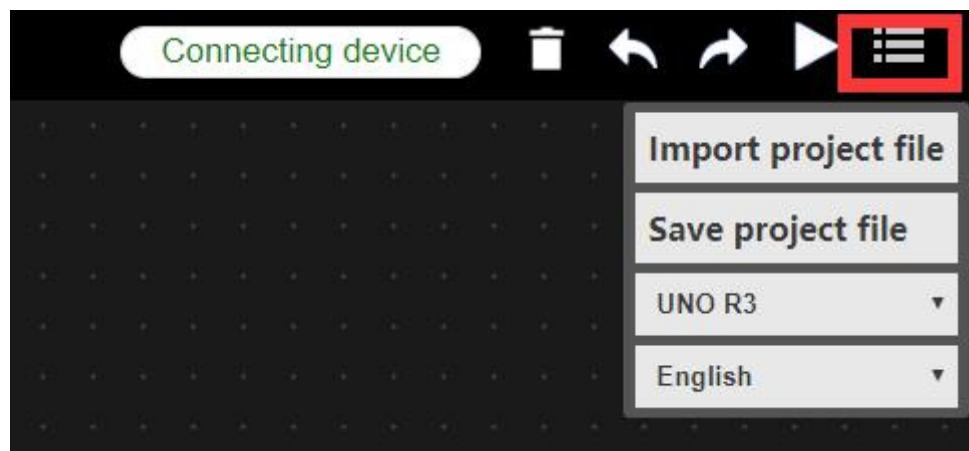


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

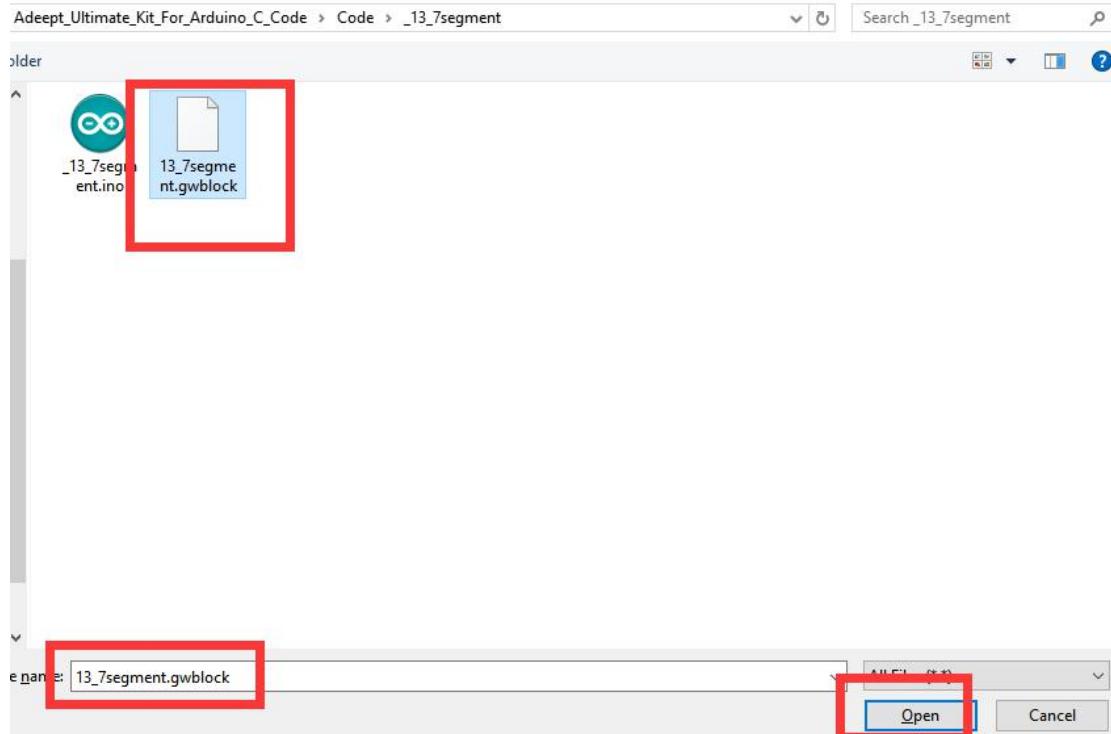
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_13_7segment

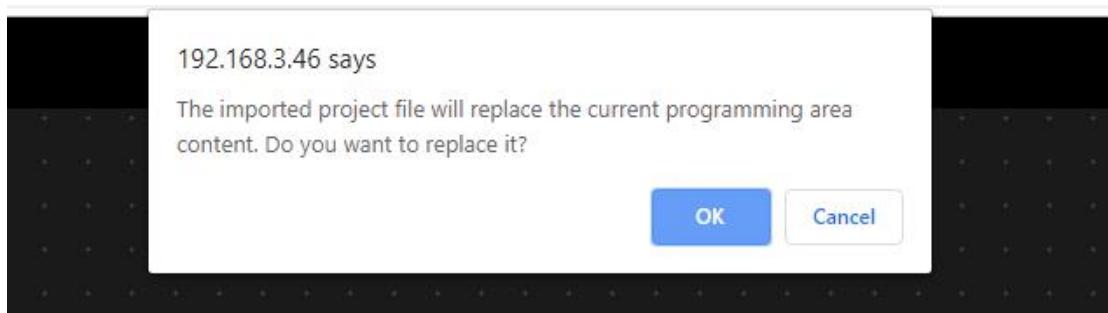
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

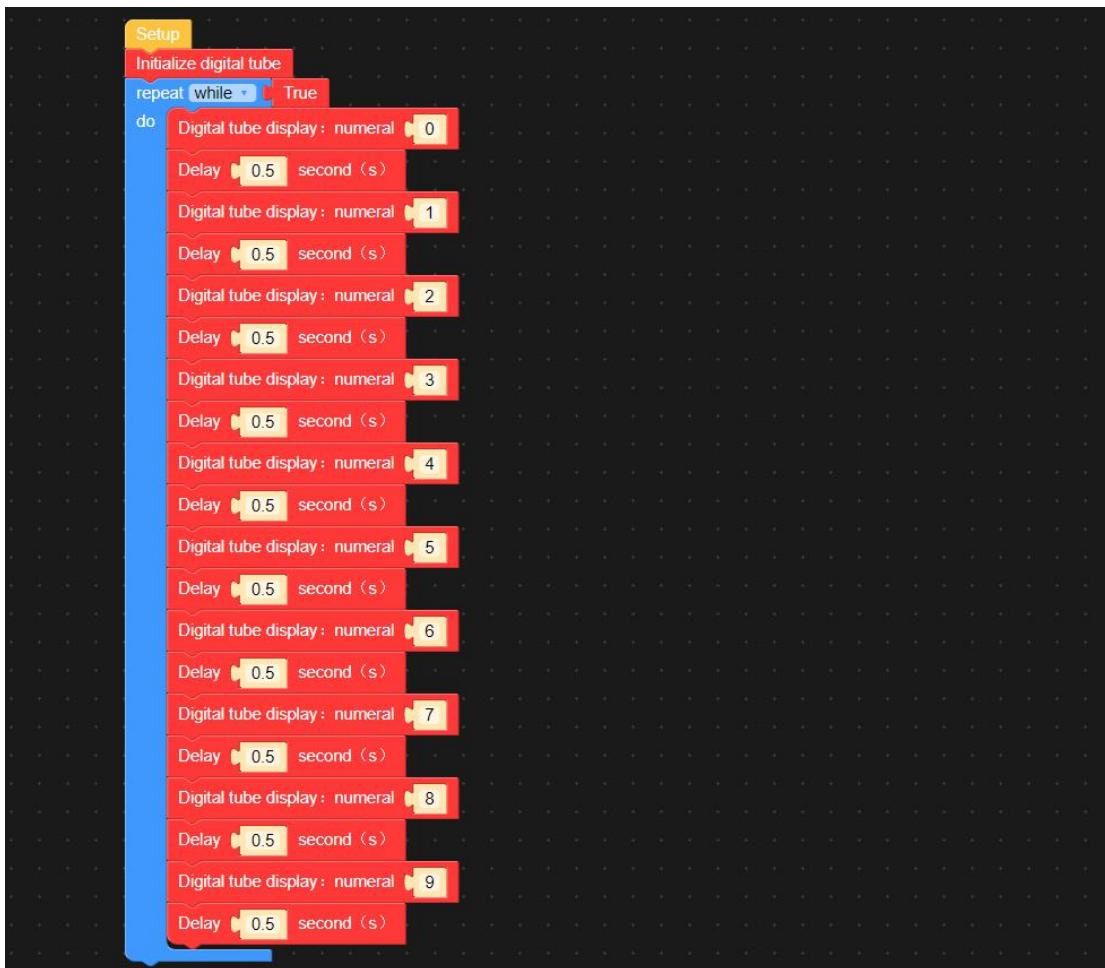
5. Select the "13_7segment.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



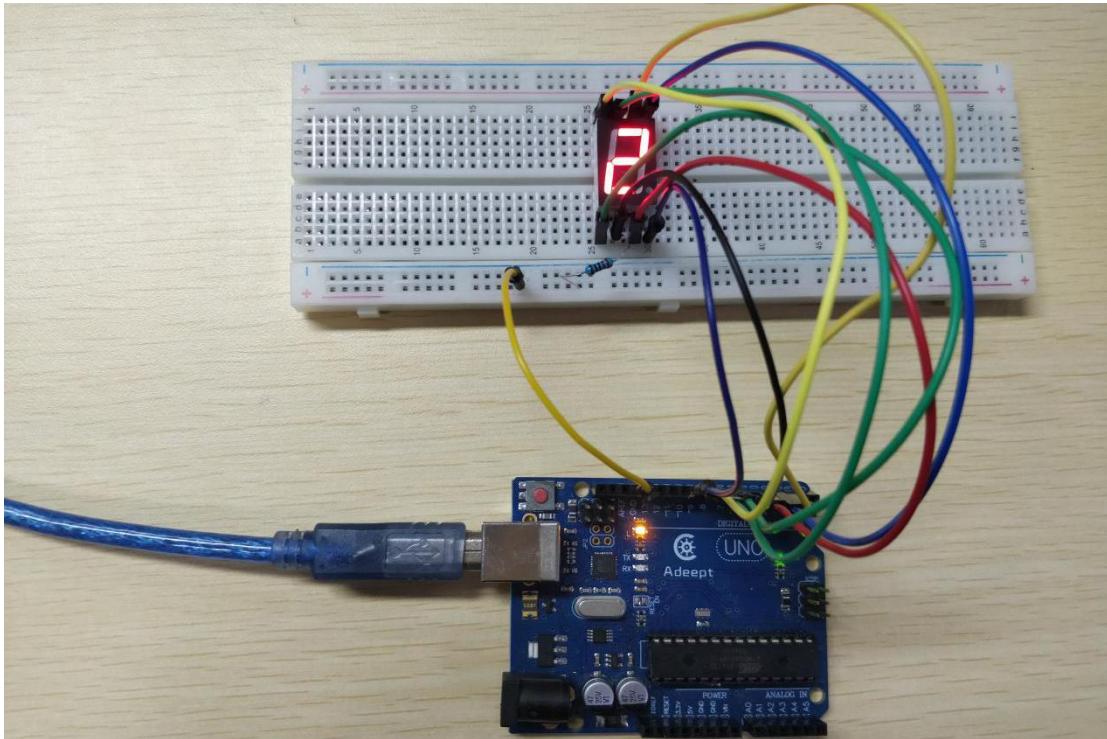
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



8. After successfully running the program, we observe the digital tube and will find that the number changes from 0 to 9, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



(3) Core code program

After the above hands-on operation, you must be very interested to know how we program to control the the 7-segment Display with potentiometer on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

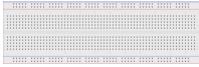
In the GwBlock graphical editor, all code programs are executed from **Setup** .Initialize the 7-segment Display through the instruction block **Initialize digital tube** .Set the 7-segment Display through the instruction block **Digital tube display: numeral** in turn to display the numbers 0 to 9.



Lesson 14 Making A Simple Counter

In this lesson, we will learn how to make a simple counter with 4-digit 7-segment Display.

1. Components used in this course

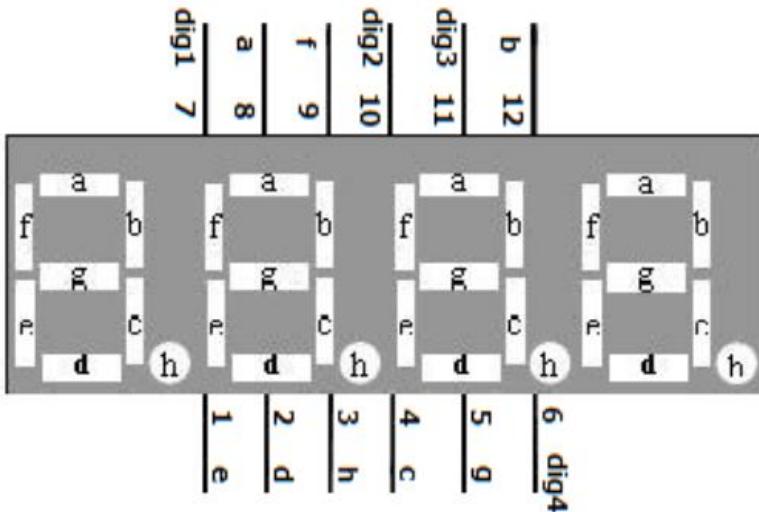
Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	1	
4-digit 7-segment Display	1	

2. The introduction of the 4-digit 7-segment Display

(1)The 4-digit 7-segment Display

The 4-digit 7-segment display module is also a kind of LED display screen. It can be divided into a common positive display and a common negative display according to its connection form. LED display panel (LED panel) is a display screen used to display information by controlling the display mode of semiconductor light emitting diodes. Such as text, graphics, images, animations, quotes, videos, video signals, etc. It consists of a 12-pin 4-digit 7-segment common anode digital tube and a control chip TM1637. A 4 * 8 shape LED display device composed of 32 LEDs (including four decimal points), these segments are named a, b, c, d, e, f, g, h, dig1,

dig2, dig3, dig4.

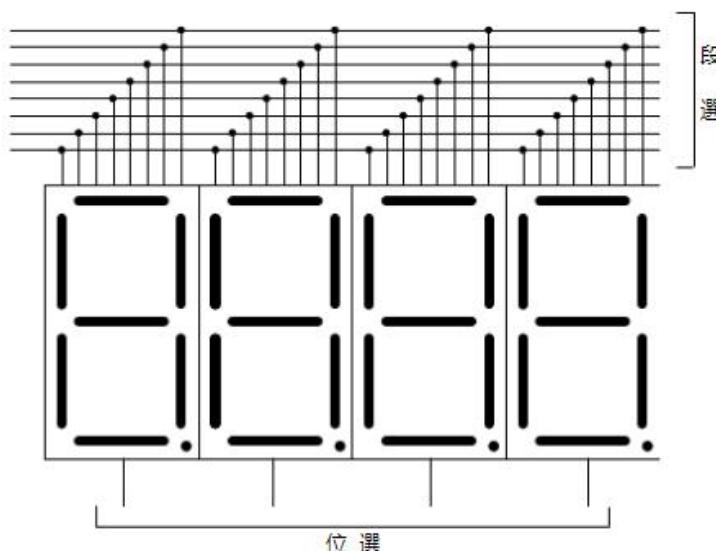


(2)Working principle of the 4-digit 7-segment Display

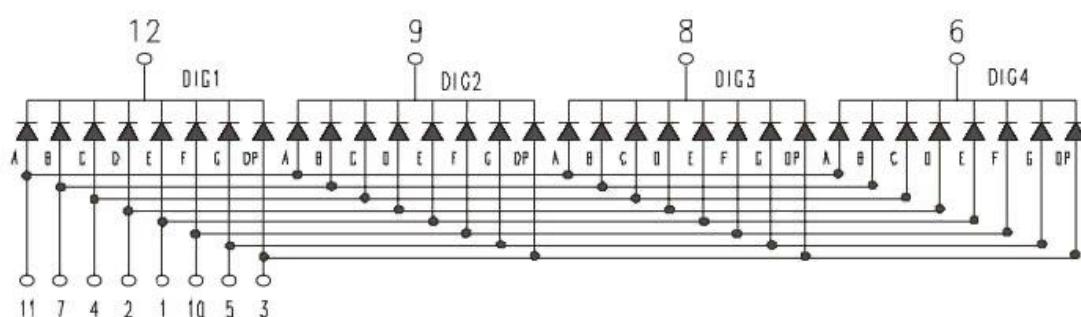
Since all the segment selection lines are connected to the same I / O in parallel, it is controlled by this I / O port. Therefore, if all 4-digit 7-segment LEDs are selected, the 4-digit 7-segment LED will display the same characters. To make the 7-segment LED of each bit display different characters, you must use dynamic scanning method to turn on each 7-segment LED in turn, that is, only one 7-segment LED is selected to display individual characters at each instant. During this lighting period, the segment selection control I / O port outputs the segment selection code of the corresponding character to be displayed, and the bit selection control I / O port outputs the bit selection signal, sending the strobe level to the bit to be displayed (The common cathode sends a low level, and the common anode sends a high level), so that the corresponding character is displayed. In this way, the four-digit 7-segment LEDs are turned on in turn, so that each digit displays the character that the digit should display. Since the visual retention time of the human eye is 0.1 seconds, when the interval of each display does not exceed 33ms, and it is maintained until the next display during the display, the eye looks like 4 due to the visual retention effect of the human eye Bit 7 segment LED lights are all lit. When designing, pay attention to the interval time of each display. Because the extinguishing time of one 7-segment LED cannot exceed

100ms, that is to say, the time used to light up other bits cannot exceed 100ms. When displaying, the time t of each bit interval must meet the following formula: $t \leq 100ms / (N-1)$

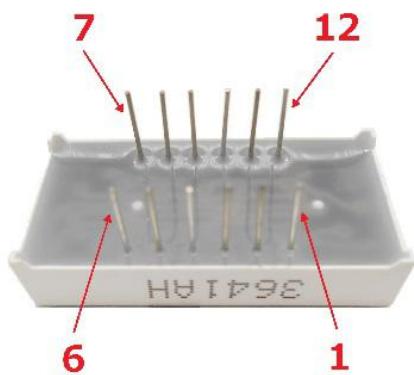
For example, if 4 bits are used now, that is, $N = 4$, then $t \leq 33ms$ can be calculated from the formula, that is, the interval between each bit cannot exceed 33ms. Of course, the time can also be set shorter, such as 5ms or 1ms. As shown in the following figure:



What we use in this experiment is a common cathode 4-digit 7-segment display. Its internal structure is as shown below:

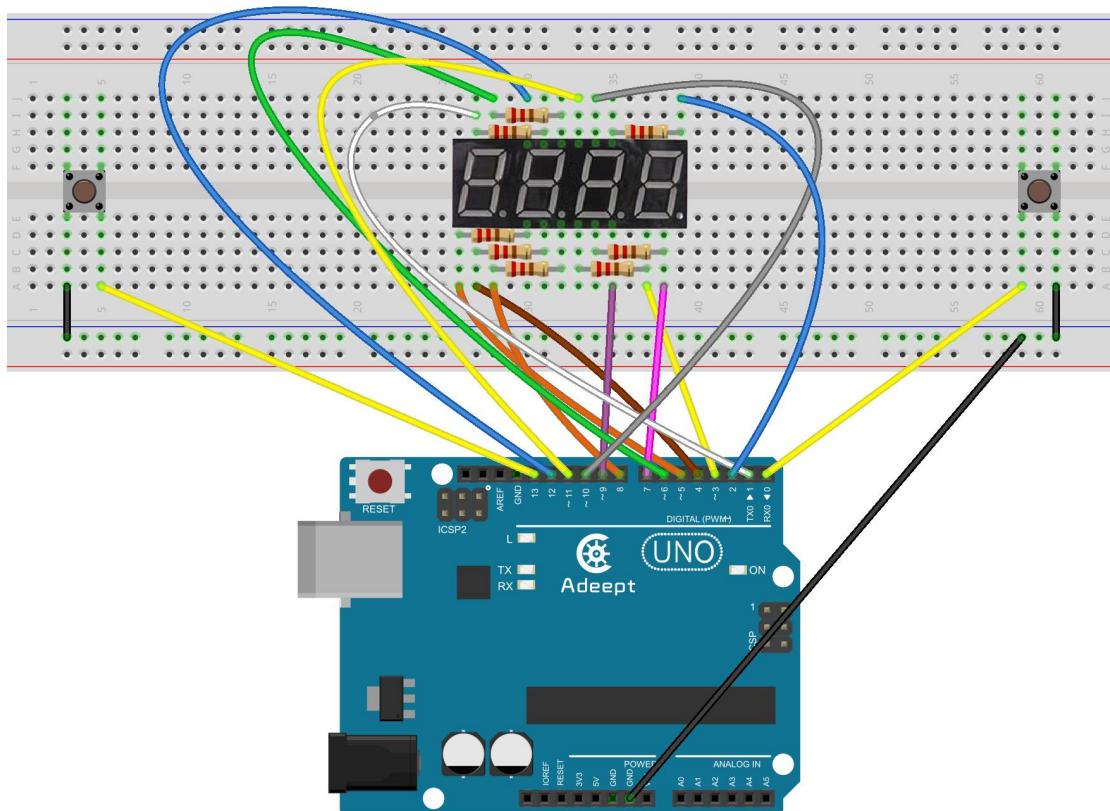


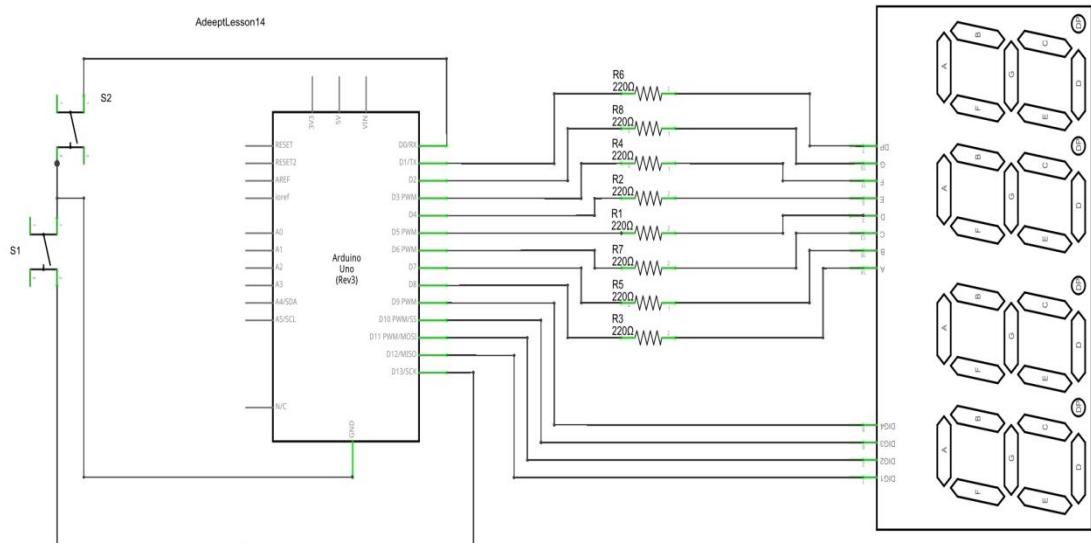
The pin number is as follows:



3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. Use $220\ \Omega$ resistor. As shown in the following figure:





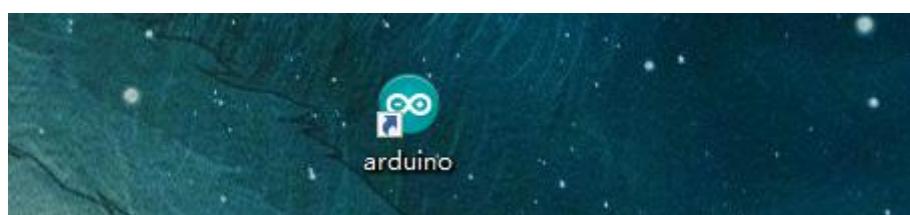
4. Making a simple counter

We provide two different methods to make a simple counter. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

1. Using C language to program to make a simple counter on Arduino UNO

(1) Compile and run the code program of this course

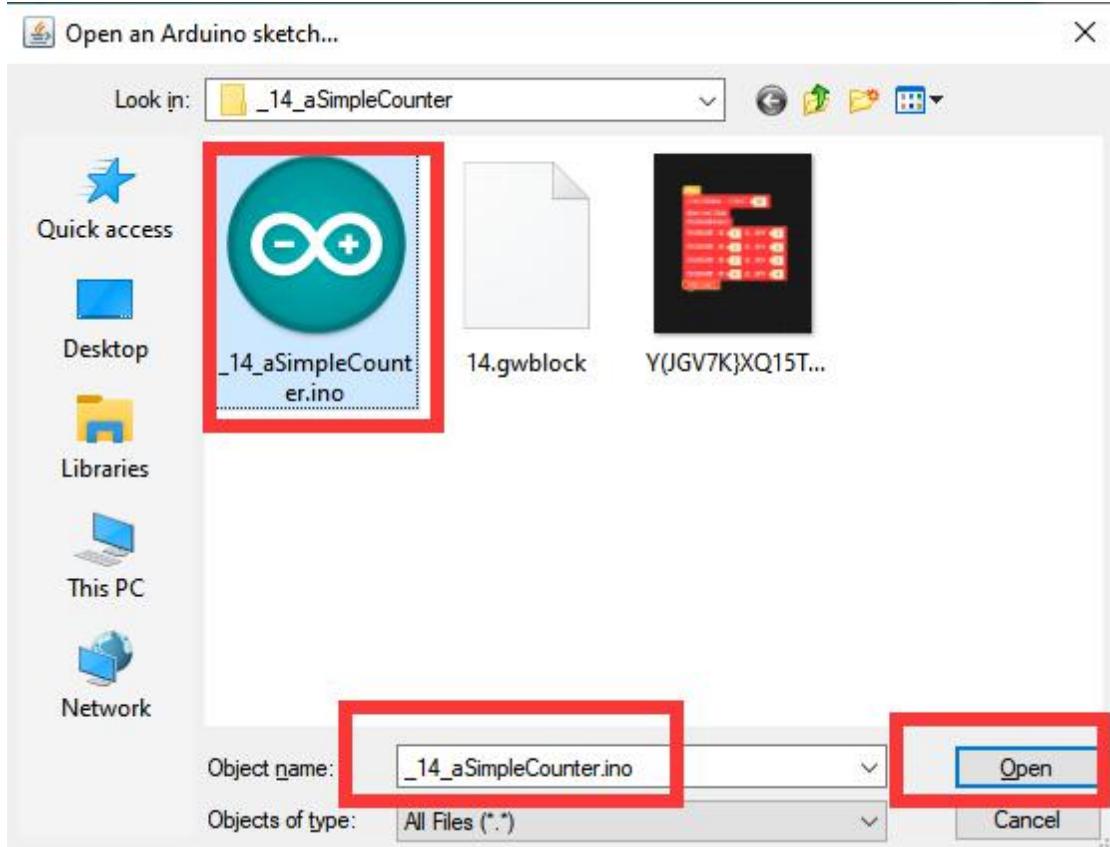
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\14_aSimpleCounter directory. Select _14_aSimpleCounter.ino. This file is the code program we need in this course. Then click Open.



4. After opening, click  to upload the code program to the Arduino UNO. If there is

no error warning in the console below, it means that the Upload is successful.

```

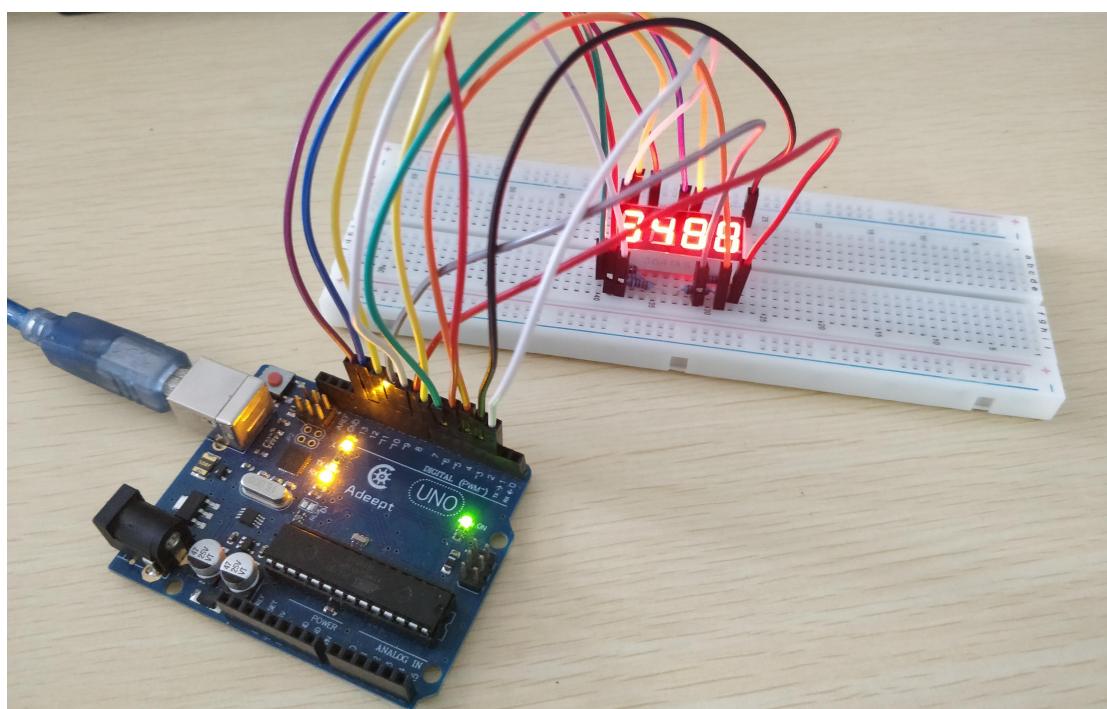
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1
Arduino Uno on COM4

```

5. After successfully running the program, we will find that the digital tube will display 4 digits, , and the digits will change and expand. This shows that our experimental test is successful. The physical connection diagram of the experiment is as bellow:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to make a simple counter on Arduino UNO. We will introduce how our core code can be achieved:

1.In the setup() function, set btn1pin and btn2pin to INPUT_PULLUP mode through pinMode() function, and set the pin connected to Arduino UNO to OUTPUT mode through pinMode() function.

```

void setup()
{
    pinMode(btn1pin, INPUT_PULLUP); //Set digital 2 port mode, the INPUT for the input
    pinMode(btn2pin, INPUT_PULLUP); //Set digital 2 port mode, the INPUT for the input
    pinMode(d1, OUTPUT);
    pinMode(d2, OUTPUT);
    pinMode(d3, OUTPUT);
    pinMode(d4, OUTPUT);
    pinMode(a, OUTPUT);
    pinMode(b, OUTPUT);
    pinMode(c, OUTPUT);
    pinMode(d, OUTPUT);
    pinMode(e, OUTPUT);
    pinMode(f, OUTPUT);
    pinMode(g, OUTPUT);
    pinMode(p, OUTPUT);

    pickDigit(1);           //Selection of a digital display
    pickNumber(1);          //Display digital d1

    pickDigit(2);           //Selection of a digital display
    pickNumber(2);          //Display digital d1
    pickDigit(3);           //Selection of a digital display
    pickNumber(3);          //Display digital d1
    pickDigit(4);           //Selection of a digital display
    pickNumber(4);          //Display digital d1
}

}

```

2.In the loop() function, set the counter to start counting through data5++. The pickDigit(1) function is used to light up the nixie tube on the 4-digit 7-segment display. The pickNumber(data5/1000%10) function is used to set the data displayed on this bit.

```

void loop()
{
    data5++;
    pickDigit(1);           //Selection of a digital display
    pickNumber(data5/1000%10);      //Display digital d1
    delay(5);
    pickDigit(2);           //Selection of a digital display
    pickNumber(data5/100%10);      //Display digital d1
    delay(5);
    pickDigit(3);           //Selection of a digital display
    pickNumber(data5/10%10);      //Display digital d1
    delay(5);
    pickDigit(4);           //Selection of a digital display
    pickNumber(data5%10);       //Display digital d1
    delay(5);

}

```

2. Programming to make a simple counter on Arduino UNO with graphical code blocks

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1.Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

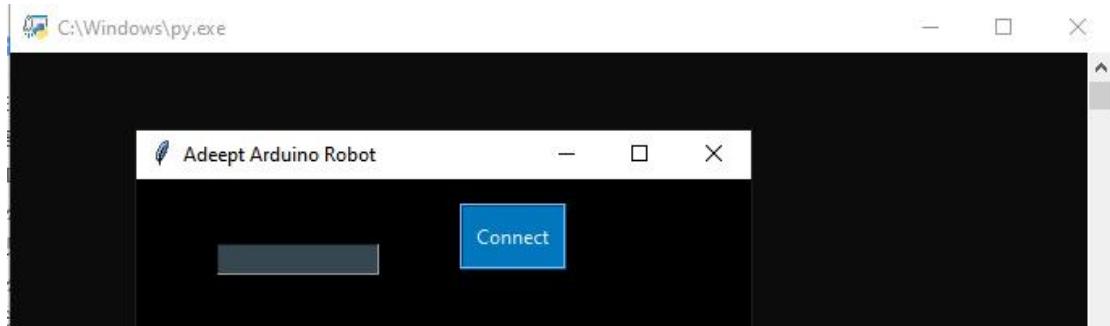
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

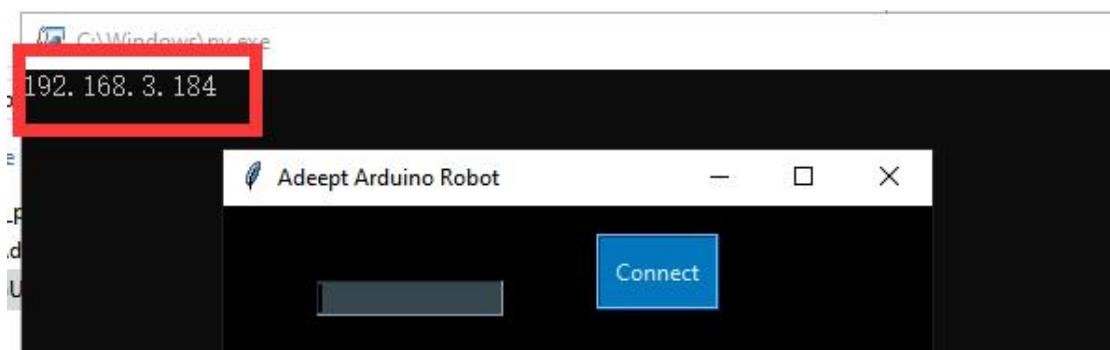
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

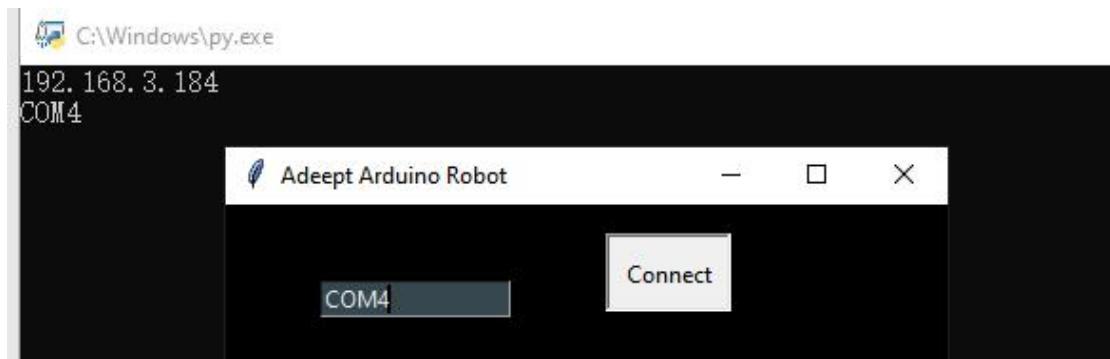
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



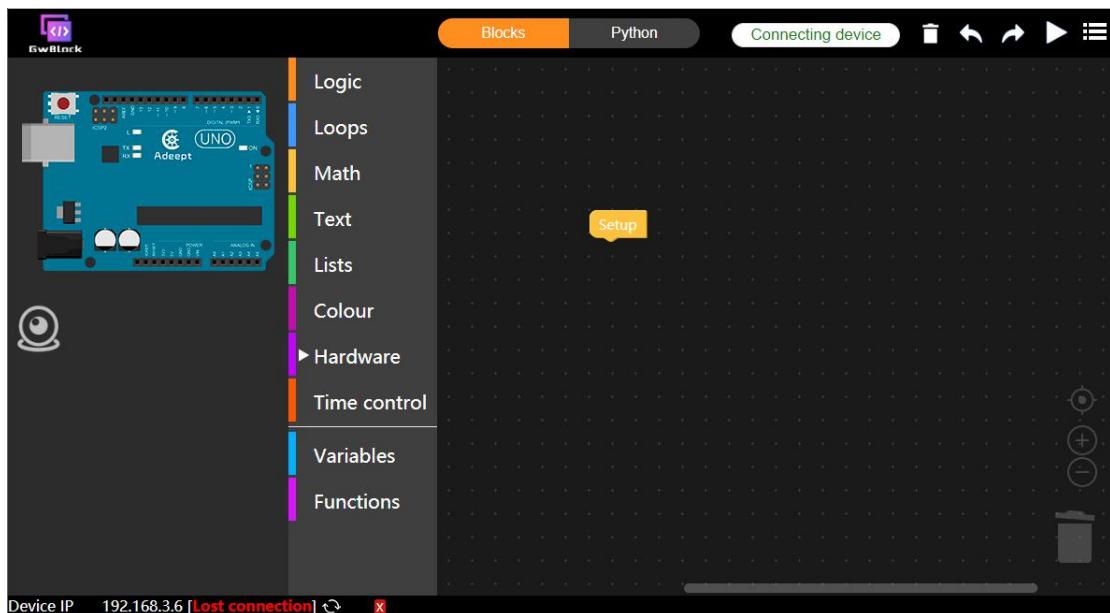
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



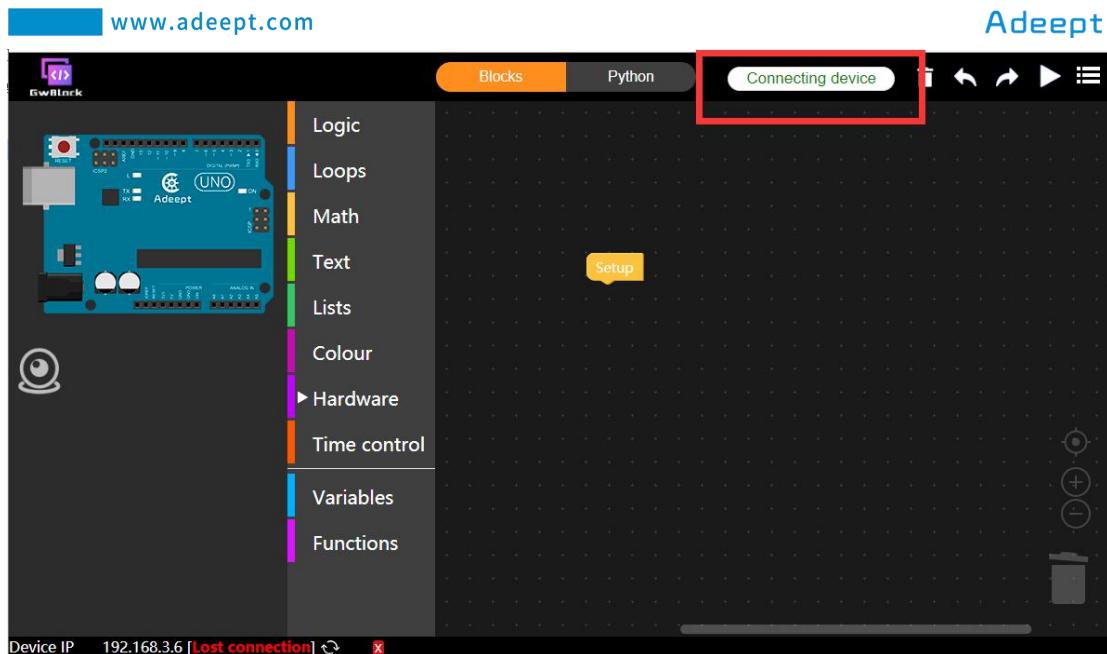
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

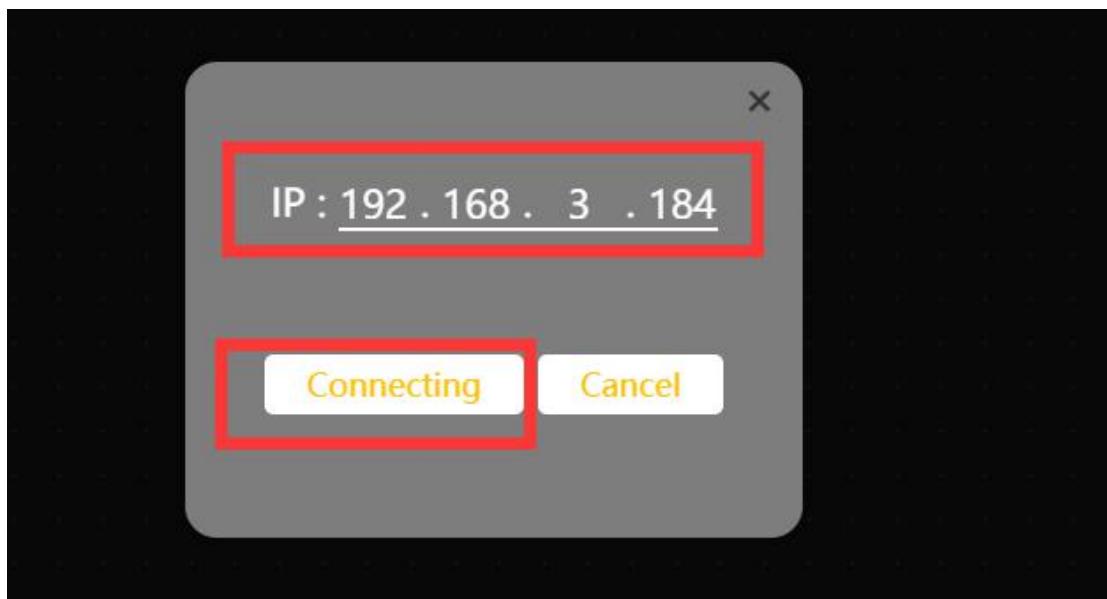
After successfully entering the website, the interface is as below:



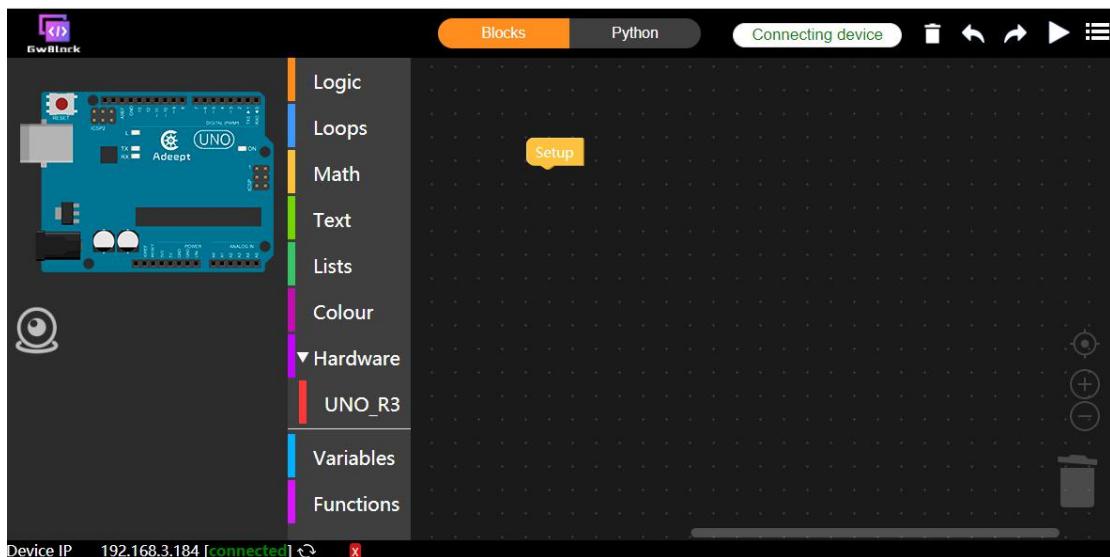
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:



(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3.The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

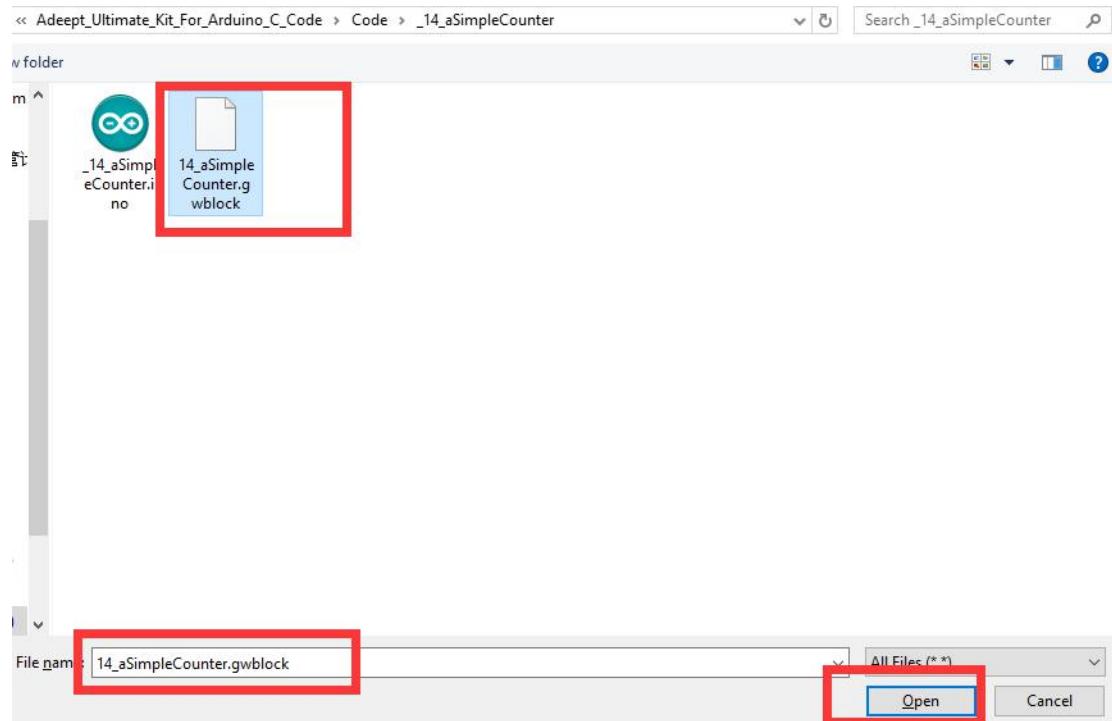
4.Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_14_aSimpleCounter

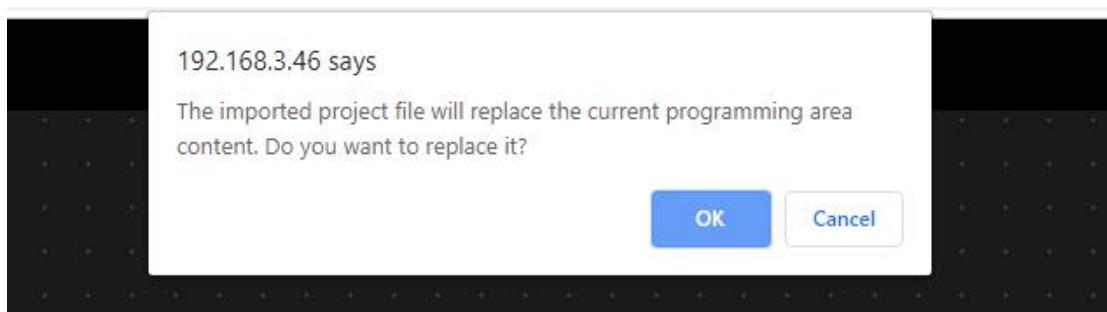
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

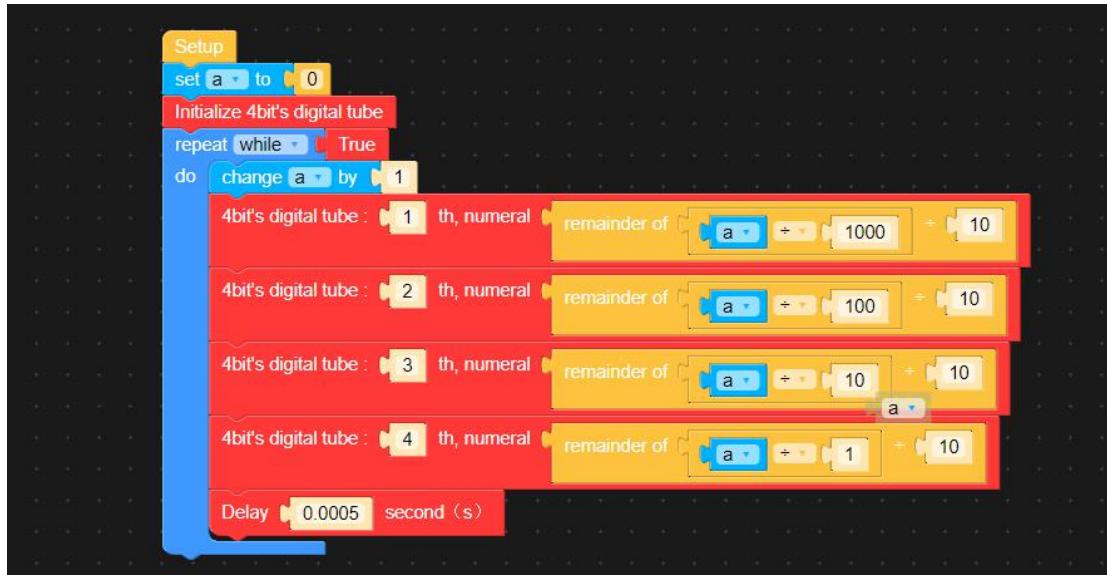
5. Select the "14_aSimpleCounter.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



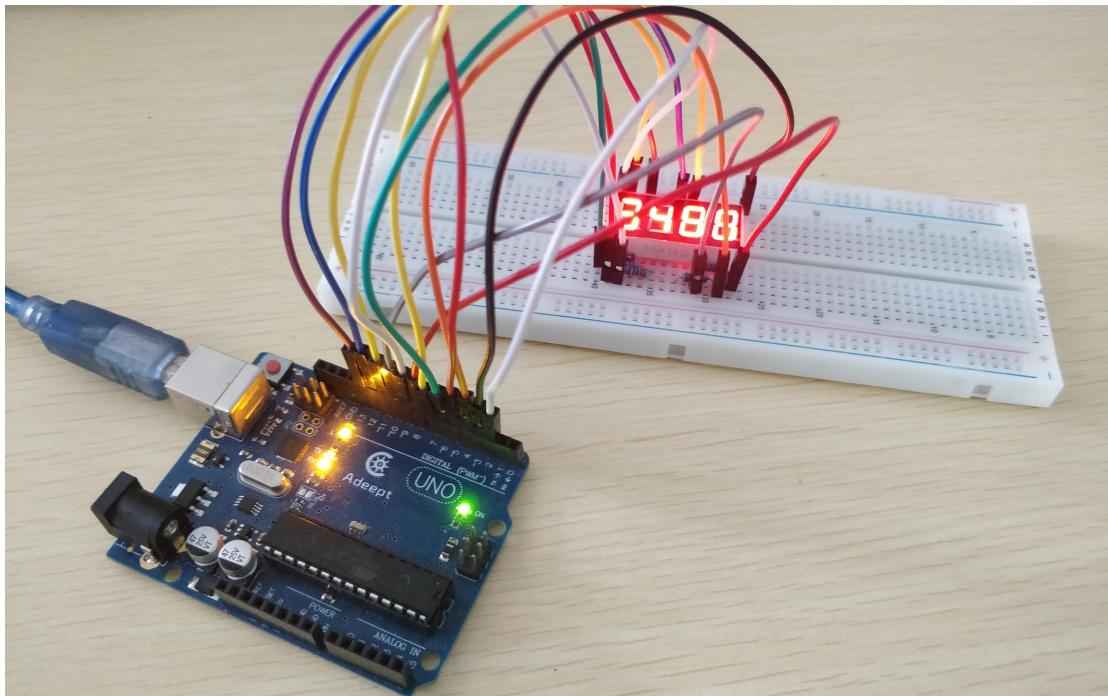
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



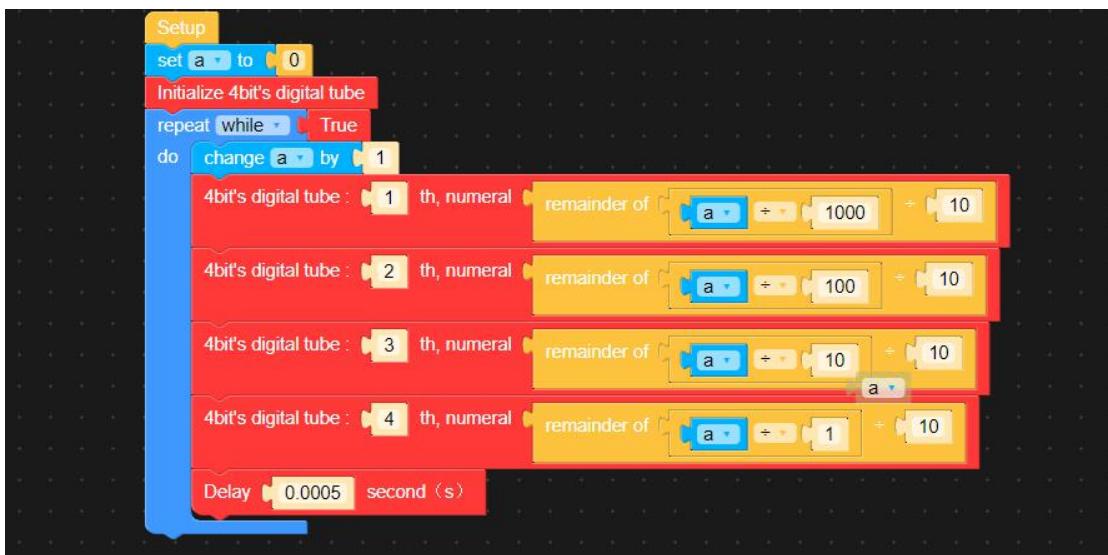
8.Click the button  on the upper right corner.After successfully running the program, we will find that the digital tube will display 4 digits, , and the digits keep changing . This shows that our experimental test is successful. The physical connection diagram of the experiment is as bellow:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to make a counter on Arduino UNO with graphical code blocks.We will introduce how our core code can be achieved:

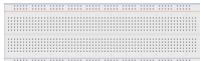
In the GwBlock graphical editor, all code programs are executed from **Setup**.Initialize the 4-digit 7-segment display through instructions **Initialize 4bit's digital tube**.Set the counter to start counting through the instruction block **change a by 1**. Sets the thousand-digit display data on the 4-digit 7-segment display through the instruction block .



Lesson 15 Controlling the Servo Motor

In this lesson, we will learn how to control the Servo Motor.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
AD002 Servo	1	

2. The introduction of the Servo Motor

(1) Servo Motor

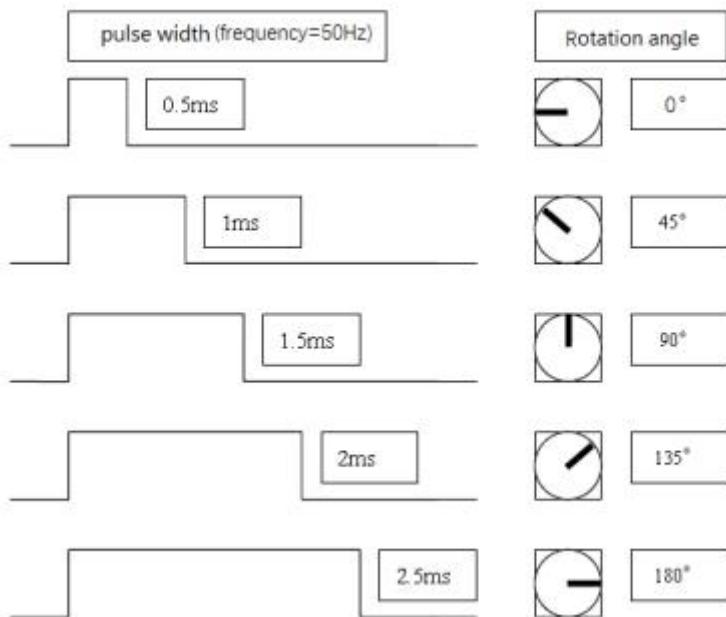
Servo motor refers to the engine that controls mechanical component operation in the servo system. It is a kind of auxiliary motor indirect transmission device. The servo motor is a gear motor that can rotate only 180 degrees. It is controlled by sending pulses from the microcontroller. These pulses tell the server where to move. The servo motor system includes housing, circuit board, non-core motor, gearing and position detection. Servo motor is shown in the figure:



(2)The working principle of the Servo Motor

The servo mechanism is an automatic control system that enables the object's position, orientation, state and other output controlled quantities to follow arbitrary changes in the input target (or given value). The servo mainly depends on Pulsefor location. Basically, it can be understood that the servo motor receives an impulse and rotates the angle corresponding to the impulse to realize displacement. Because the servo motor itself has the function of sending out pulses, the servo motor rotates every time at an angle, and a corresponding number of pulses will be sent out. In this way, the pulses received by the servo motor form a response, or a closed loop. In this way, the system will know how many pulses are sent to the servo motor and how many pulses are received. In this way, it is possible to precisely control the rotation of the motor, thereby achieving precise positioning

Arduino sends a PWM signal to a servomotor, which is then processed by an IC on the circuit board to calculate the rotation direction of the drive motor, which is then transmitted through a reduction gear to the swing arm. At the same time, the position detector returns a position signal to determine whether the set position has been reached or not.

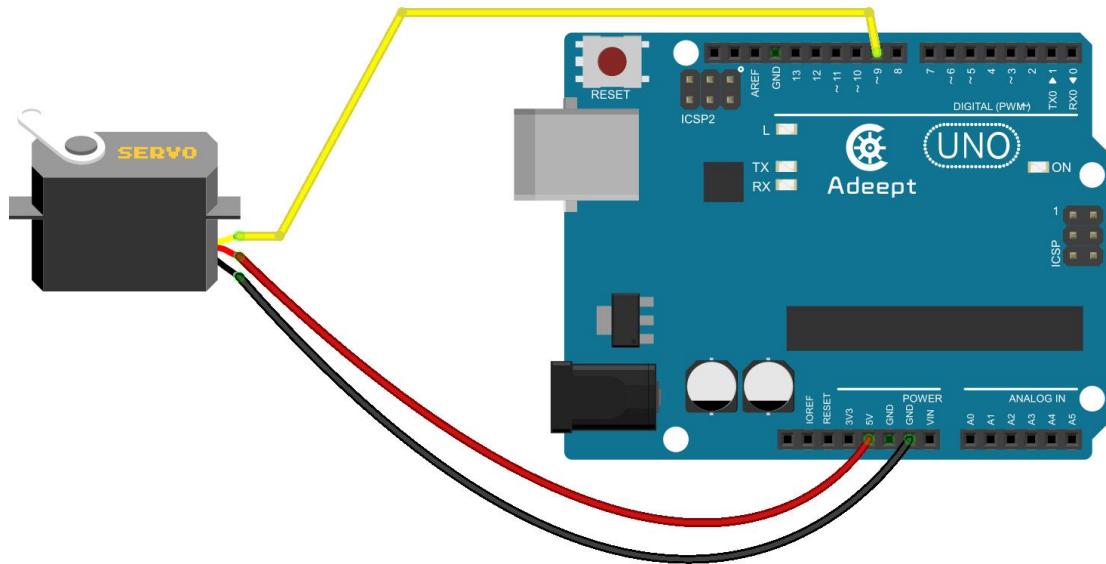


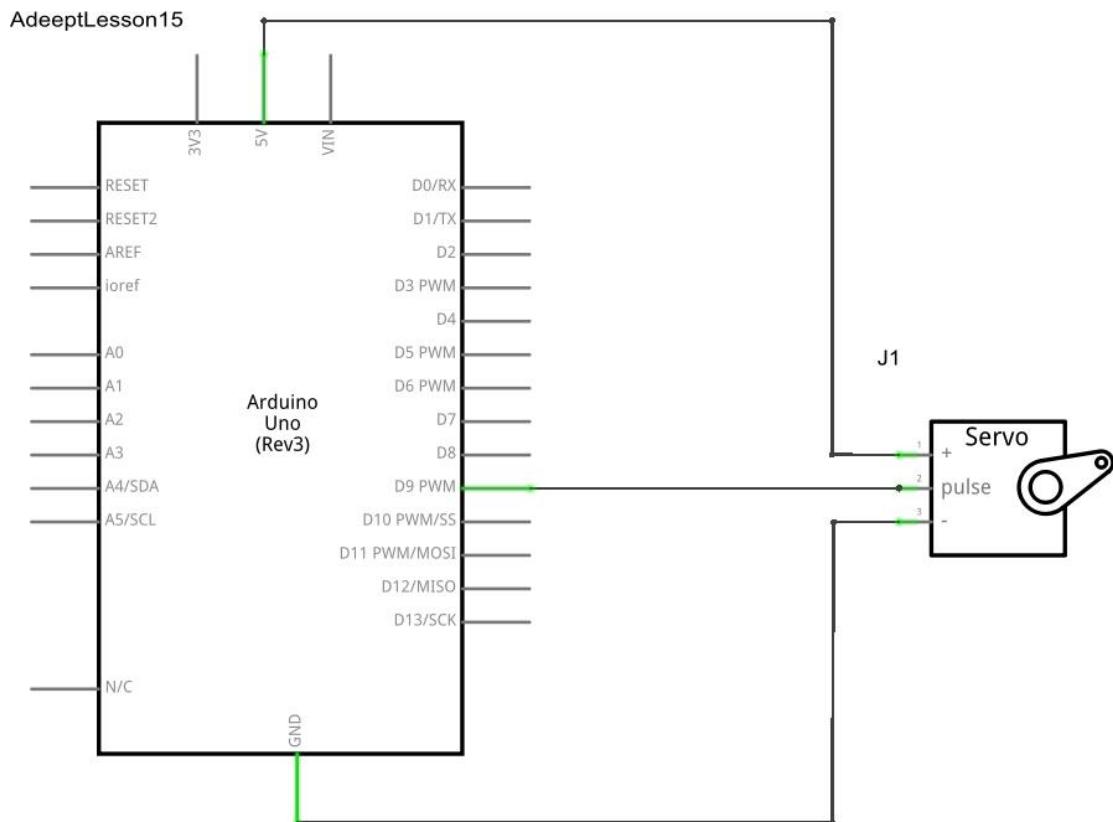
The servo motor has three wires: power, ground, and signal. The power wire is typically red, and should be connected to the 5V pin on the Arduino board. The ground wire is typically black or brown and should be connected to a ground pin on the Arduino board. Usually the signal pin is yellow, orange or white, and should be connected to a digital pin on the Arduino board. Note that the servo motor draws a considerable amount of power, if you need to drive more than one or two servos, you'll probably need to power them with an extra supply (i.e. not the +5V pin on your Arduino). Be sure to connect the grounds of the Arduino and external power supply together.



3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:





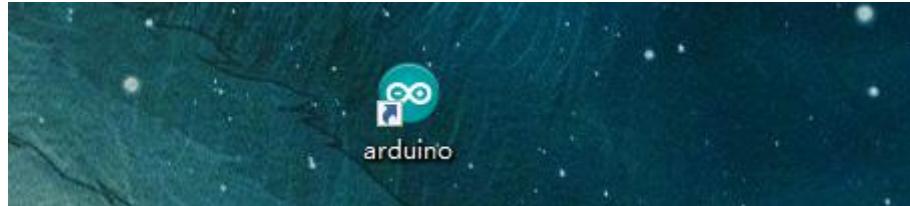
4. Controlling the Servo Motor

We provide two different methods to control the servo motor. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

1. Using C language to program to control the servo motor on Arduino UNO

(1) Compile and run the code program of this course

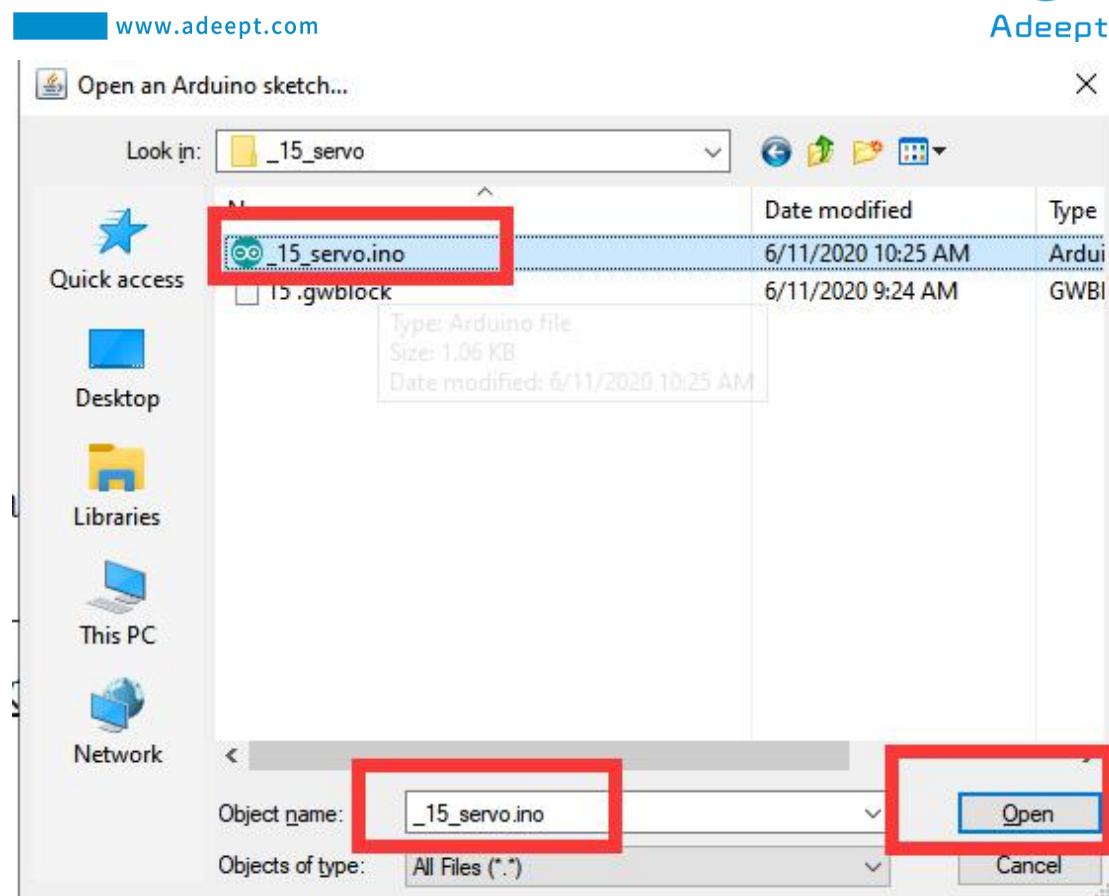
1. Open the Arduino IDE software, as shown below:



2.Click Open in the File drop-down menu:



3.Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\15_servo directory. Select _15_servo.ino. This file is the code program we need in this course. Then click Open.



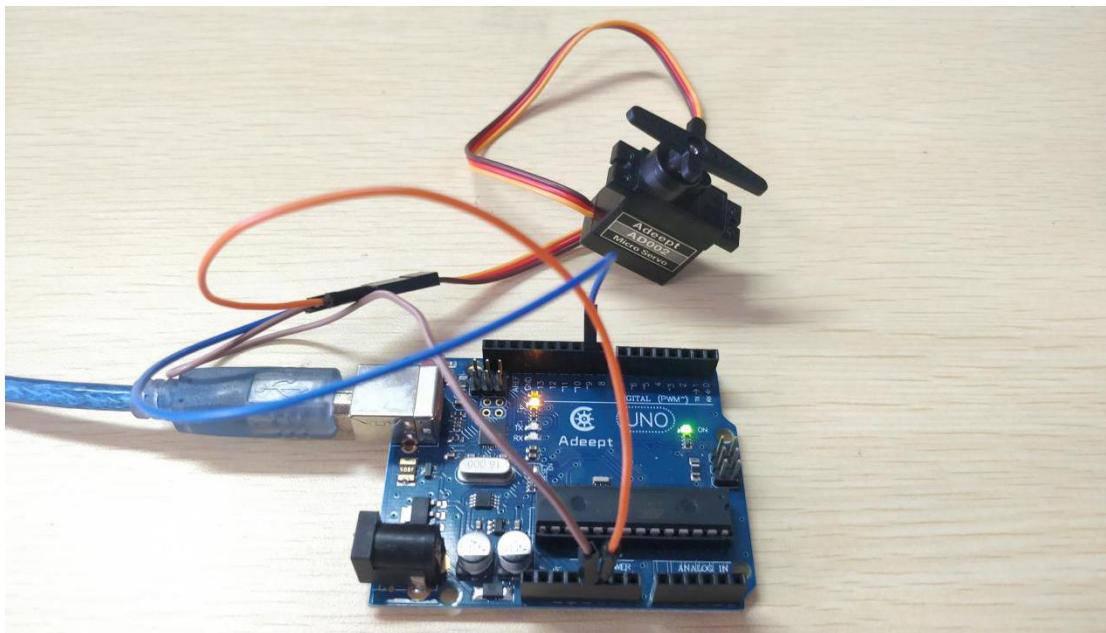
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                                         Arduino Uno on COM4
```

5. After successfully running the program, we observe the experiment and find that the servo motor will rotate clockwise and counterclockwise, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



(2)The core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to control the servo motor on Arduino UNO. We will introduce how our core code can be achieved:

1. In the setup() function, the myservo.write(0) function initializes the servo motor to 0 degrees.

```
void setup()
{
    myservo.attach(9); //attaches the servo on pin 9 to servo object
    myservo.write(0); //back to 0 degrees
    delay(1000); //wait for a second
}
```

2. In the loop() function, the myservo.write(180) function controls the servo motor to rotate 180 degrees. If you want to control how much the servo motor rotates, you only need to enter the corresponding angle value in the myservo.write() function. Then the myservo.write() function will continue to control the servo motor to rotate 90 degrees, and then turn back to 0 degrees.

```
void loop()
{
    myservo.write(180); //goes to 180 degrees
    delay(2000); //wait for a second

    myservo.write(90); //goes to 90 degrees
    delay(2000); //wait for a second.33

    myservo.write(0); //goes to 0 degrees
    delay(2000); //wait for a second.33

}
```

2. Programming to control the servo motor on Arduino UNO with graphical code blocks

(1) Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

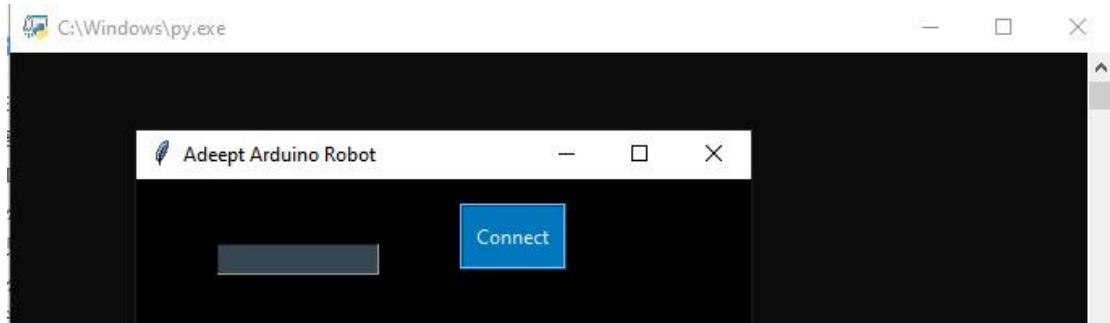
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

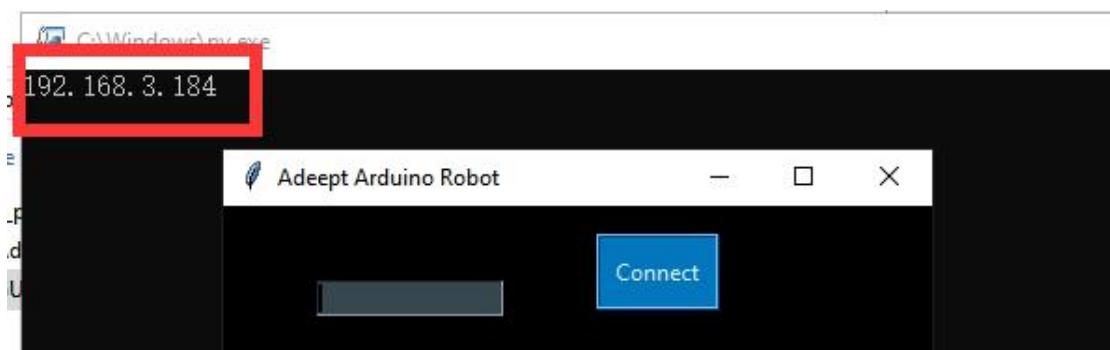
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



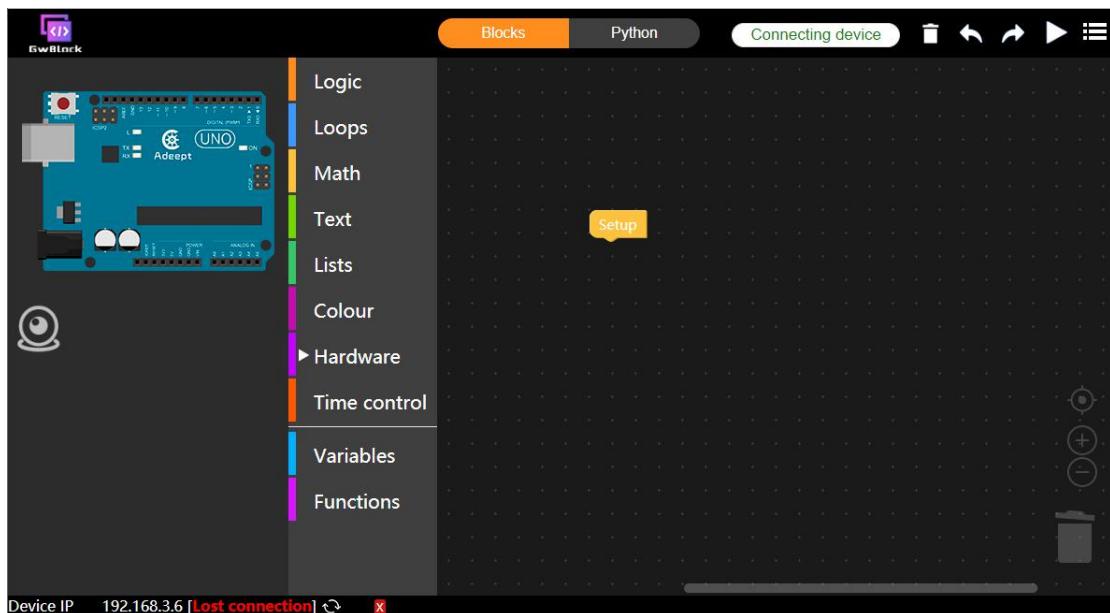
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



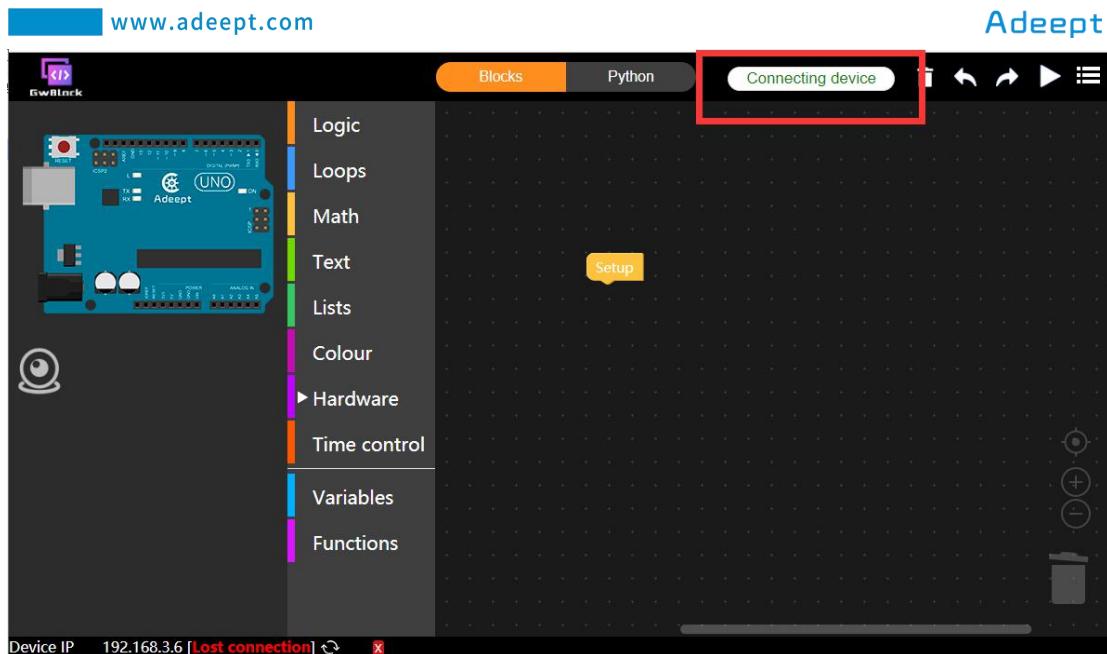
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

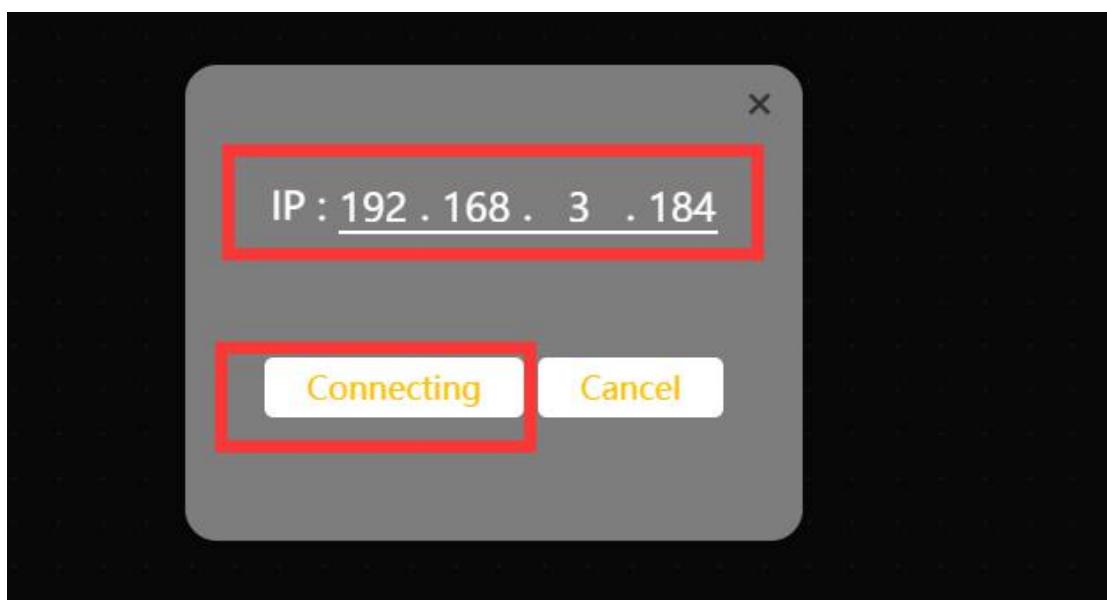
After successfully entering the website, the interface is as below:



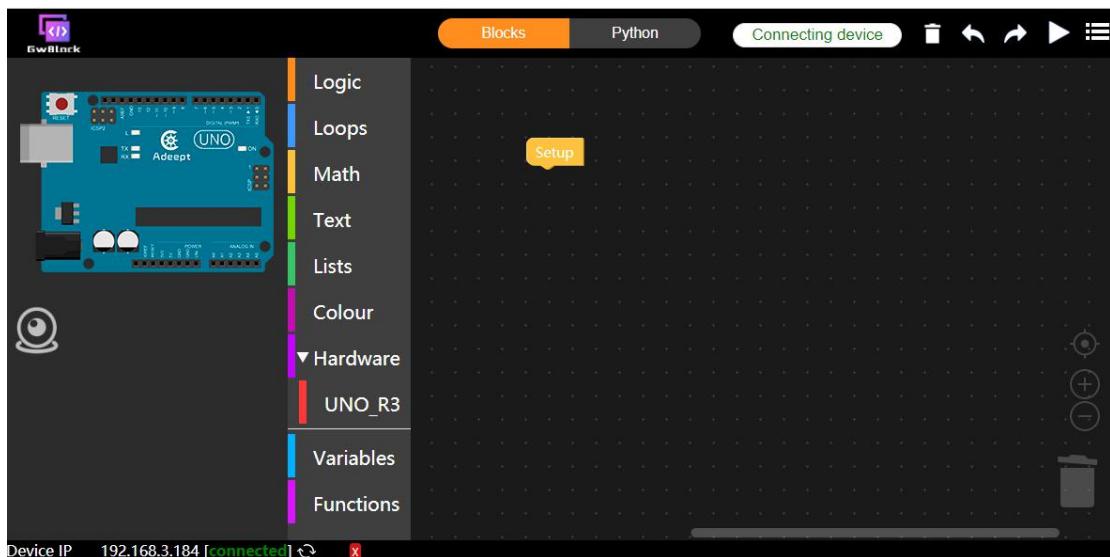
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

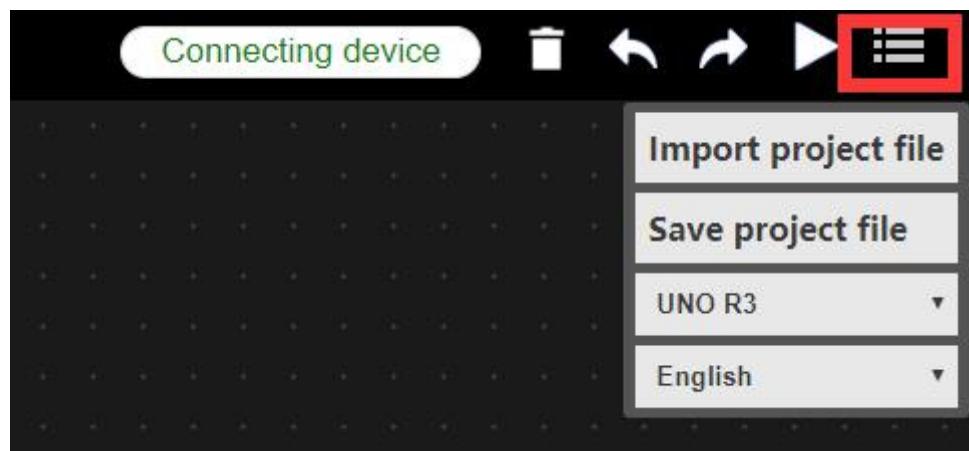


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3.The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

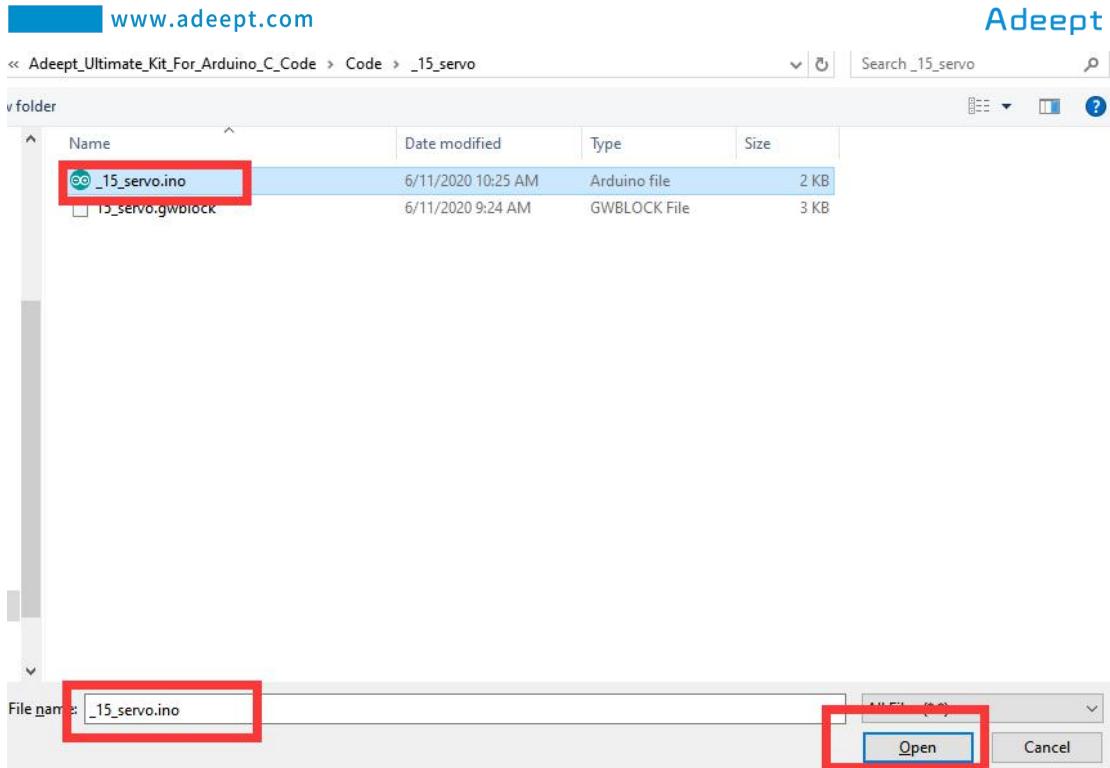
4.Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_15_servo

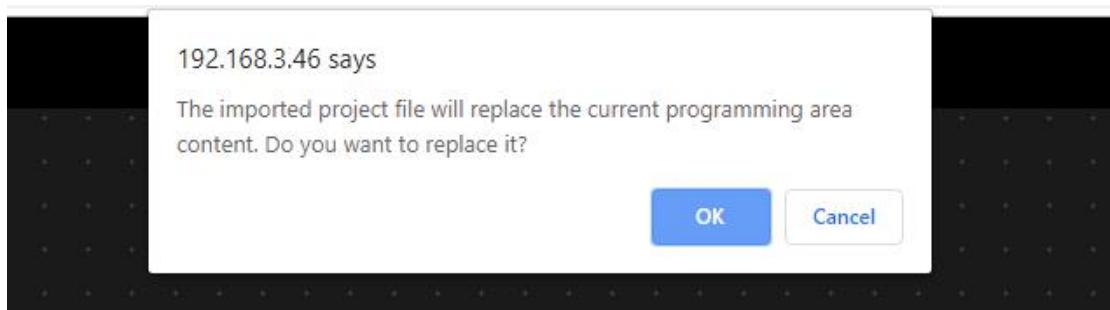
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

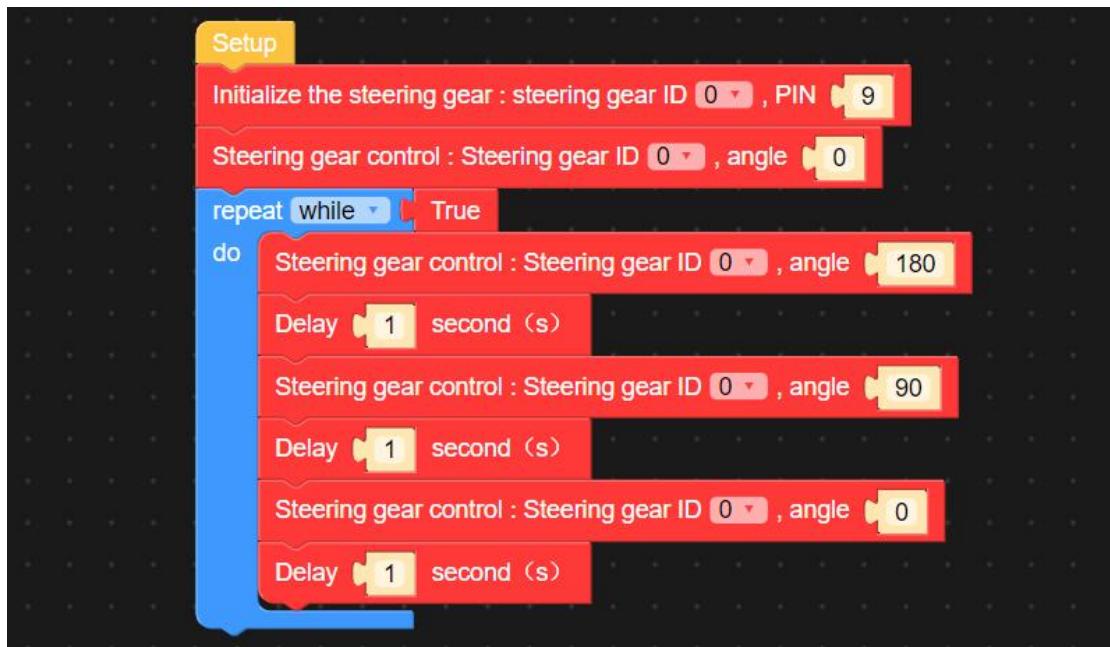
5. Select the "15_servo.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



6.Click OK.It will show as below:

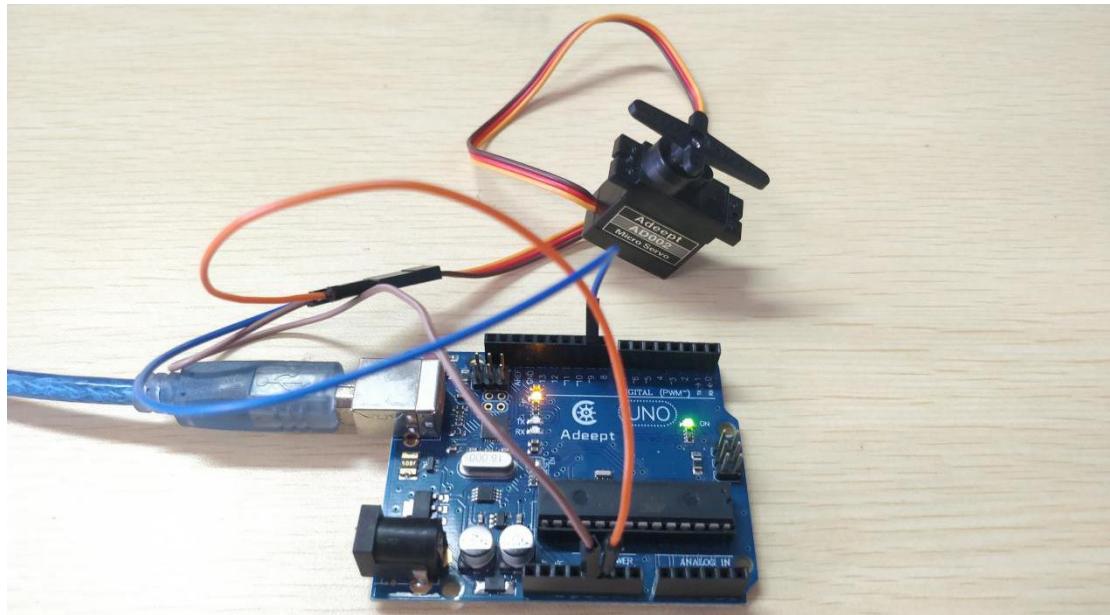


7.It will show as below after successfully opening:



8. Click the button  on the upper right corner. After successfully running the program, we observe the experiment and find that the servo motor will rotate clockwise and counterclockwise by a certain angle, indicating that our experimental test is successful.

The physical connection diagram of the experiment is as below:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to control the servo motor on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

In the GwBlock graphical editor, all code programs are executed from **Setup**.

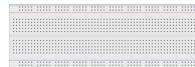
Initialize the servo motor through the instruction block **Initialize the steering gear : steering gear ID 0 , PIN 9**. The command block **Steering gear control : Steering gear ID 0 , angle 0** controls the servo motor to turn to 0 degrees, which is also the initial position. In the while loop, the instruction block **Steering gear control : Steering gear ID 0 , angle 180** controls the servo motor to rotate 180 degrees. If you need to control the servo motor to rotate other degrees, you can directly modify **180** to other data.



Lesson 16 Making a Digital Thermometer

In this lesson, we will learn how to use thermistor and LCD1602 module to make a digital thermometer.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
LCD1602	1	
Adeept IIC Module	1	
Male to Female Jumper Wires	Several	
Resistor(10KΩ)	1	
Analog Temperature Sensor(Thermistor)	1	

2. The introduction of thermistor

(1) Thermistor

Thermistors are a type of sensitive components, which are divided into positive temperature coefficient thermistors (PTC) and negative temperature coefficient thermistors (NTC) according to different temperature coefficients. The typical characteristic of the thermistor is that it is sensitive to temperature, showing different

resistance values at different temperatures. The positive temperature coefficient thermistor (PTC) has a larger resistance value at a higher temperature, and the negative temperature coefficient thermistor (NTC) has a lower resistance value at a higher temperature. They both belong to semiconductor devices.

The main characteristics of the thermistor:

- ①High sensitivity:its temperature coefficient of resistance is 10 to 100 times greater than that of metal, and it can detect the temperature change of $10\text{--}6^\circ\text{C}$;
- ②Wide operating temperature range:normal temperature device is suitable for $-55^\circ\text{C} \sim 315^\circ\text{C}$.High temperature device is suitable for temperature higher than 315°C (currently up to 2000°C). Low temperature device is suitable for $-273^\circ\text{C} \sim -55^\circ\text{C}$;
- ③Small size:able to measure the temperature of voids, cavities and blood vessels in the body that cannot be measured by other thermometers;
- ④ Easy to use:the resistance value can be arbitrarily selected from 0.1 to $100\text{k}\Omega$;
- ⑤Easy to process into complex shapes:can be mass produced;
- ⑥Good stability and strong overload capacity.



(2) Working principle

The thermistor will be in an inactive state for a long time; when the ambient temperature and current are in the c zone, the heat dissipation power of the thermistor is close to the heating power, so it may or may not operate. When the thermistor is at the same ambient temperature, the operating time decreases sharply as the current increases; the thermistor has a shorter operating time and a smaller maintenance current and operating current when the ambient temperature is relatively high.

The use of the thermistor is like a normal fuse, which is used in series in the circuit. When the circuit is working normally, the temperature of the thermistor is

close to room temperature, and the resistance is very small. The series connection in the circuit will not hinder the passage of current; and when the circuit has an overcurrent due to a fault, the temperature rises due to the increase in heating power. When the temperature exceeds the switching temperature, the resistance will increase sharply in an instant, and the current in the loop will quickly decrease to a safe value. After the thermistor operates, the current in the circuit has been greatly reduced. Due to the design of good performance of the polymer ptc thermistor, it can adjust its sensitivity to temperature by changing its own switching temperature (t_s), so it can play two roles of over-temperature protection and over-current protection at the same time.

A thermistor is a type of resistor whose resistance varies significantly with temperature, more so than in standard resistors. In this experiment we use an MF52 NTC-type thermistor, and it is usually used as a temperature sensor.

The key parameters of an MF52 thermistor:

B-parameter: 3470.

25°C resistance: 10kΩ.

The relationship between the resistance of thermistor and temperature is as follows:

$$R_{thermistor} = R * e^{(B * (\frac{1}{T_1} - \frac{1}{T_2}))}$$

R_{thermistor}: the resistance of the thermistor at temperature T1

R: the nominal resistance of the thermistor at room temperature T2

e: 2.718281828459

B: one of the important parameters of thermistor

T₁: the Kelvin temperature that you want to measure

T₂: Under the condition of a 25 °C (298.15K) room temperature, the standard resistance of MF52 thermistor is 10K;

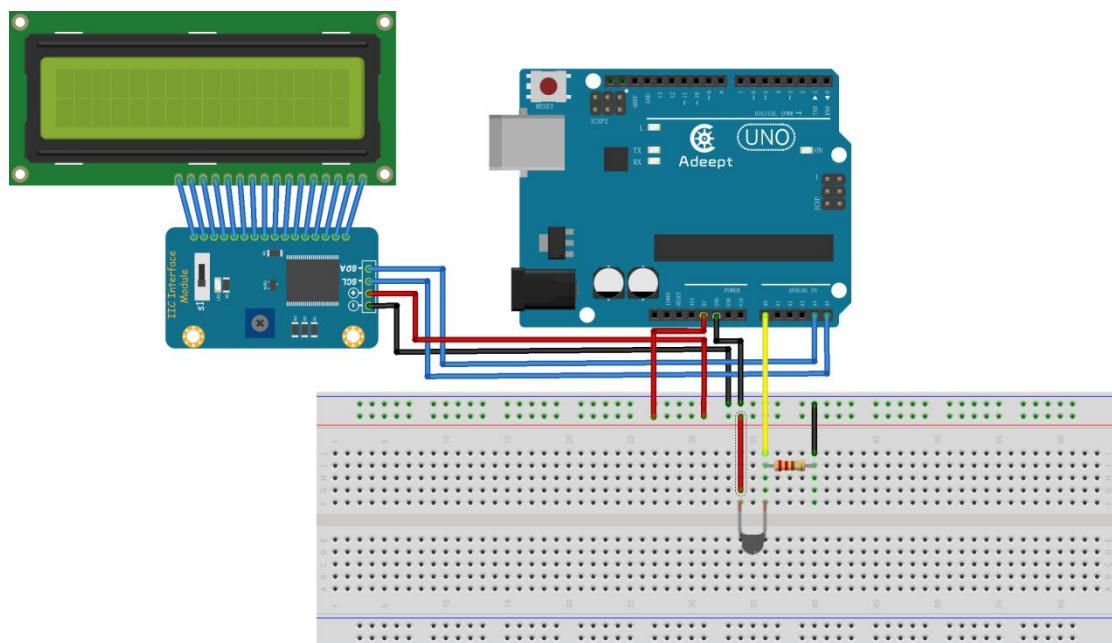
Kelvin temperature = 273.15 (absolute temperature) + degrees Celsius;

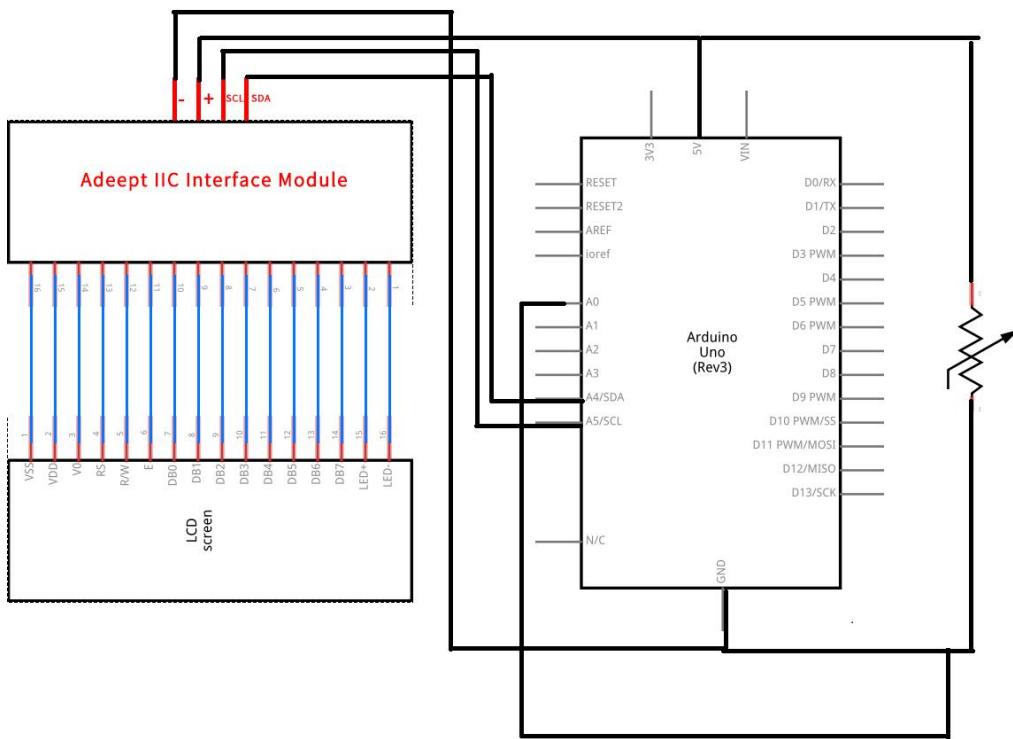
After transforming the above equation, we can get the following formula:

$$T_1 = \frac{B}{\left(\ln\left(\frac{R_{thermistor}}{R}\right) + \frac{B}{T_2} \right)}$$

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. Use $10K \Omega$ resistor. As shown in the following figure:





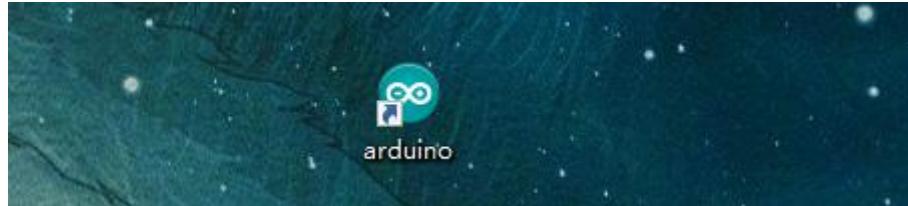
4.How to make digital thermometer

We provide two different methods to read the temperature of the thermistor. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

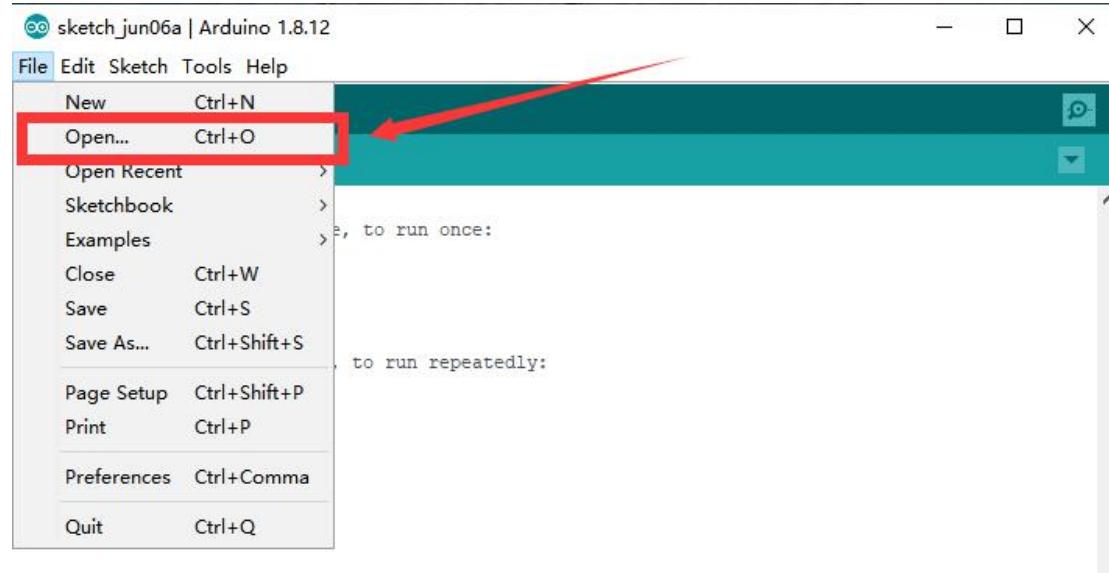
1.Using C language to program to read the temperature of the thermistor on LCD1602 on Arduino UNO

(1)Compile and run the code program of this course

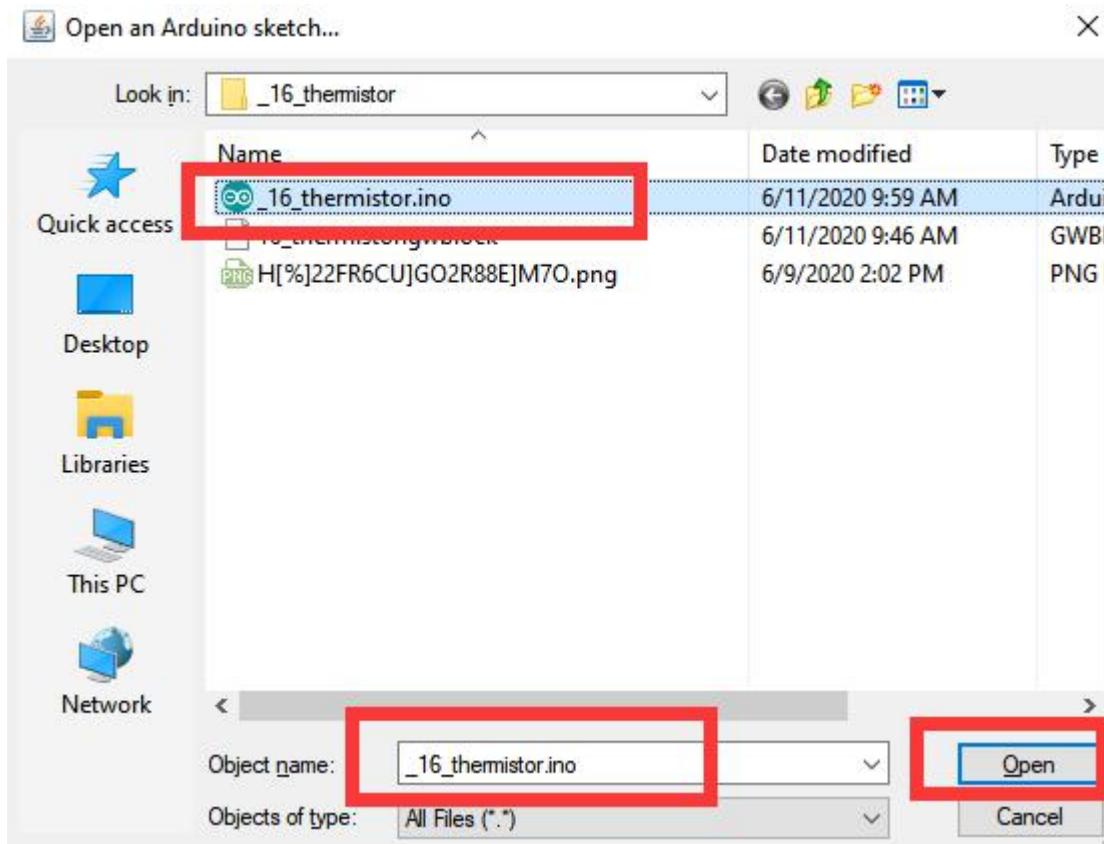
1. Open the Arduino IDE software, as shown below:



2.Click Open in the File drop-down menu:



3.Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\16_thermistor directory. Select 16_thermistor.ino. This file is the code program we need in this course. Then click Open.



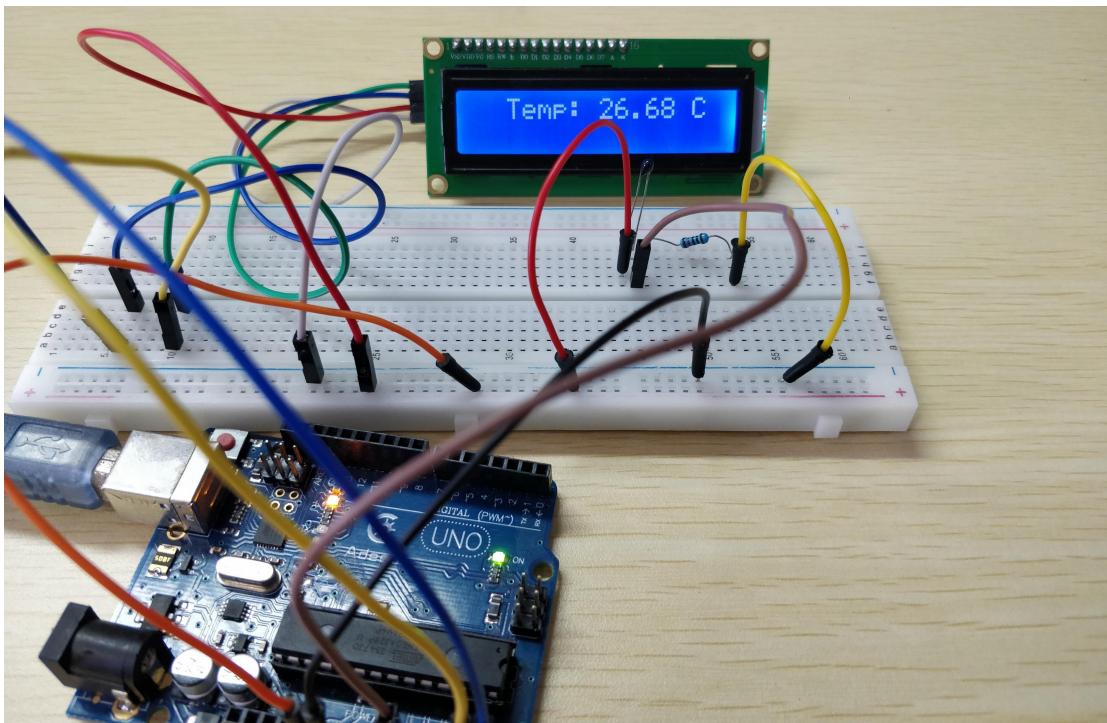
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                                         Arduino Uno on COM4
```

5. After successfully running the program, it will be observed that the LCD1602 screen displays the currently detected temperature. This shows that our experimental test is successful. The physical connection diagram of the experiment is as below:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to measure temperature on Arduino UNO. We will introduce how our core code can be achieved:

1. First, initialize the LCD through lcd.init() in the setup() function.

```
void setup()
{
    lcd.init(); //initialize the lcd
    lcd.backlight(); //turn on the backlight
}
```

2. In the loop() method, read the value of the thermistor through analogRead(thermistorPin). After the arithmetic processing, display the temperature measured by the thermistor on the LCD with the lcd.print(tempC) function.

```

void loop()
{
    float a = analogRead(thermistorPin);
    //the calculating formula of temperature
    float resistor = (1023.0*10000)/a-10000;
    float tempC = (3435.0/(log(resistor/10000)+(3435.0/(273.15+25)))) - 273.15;

    lcd.setCursor(0, 0); // set the cursor to column 0, line 0
    lcd.print("      adeept      ");// Print a message of "Temp: "to the LCD.

    lcd.setCursor(0, 1); // set the cursor to column 0, line 0
    lcd.print("  Temp: ");// Print a message of "Temp: "to the LCD.
    lcd.print(tempC);// Print a centigrade temperature to the LCD.
    lcd.print(" C ");// Print the unit of the centigrade temperature to the LCD.
    delay(500);
}

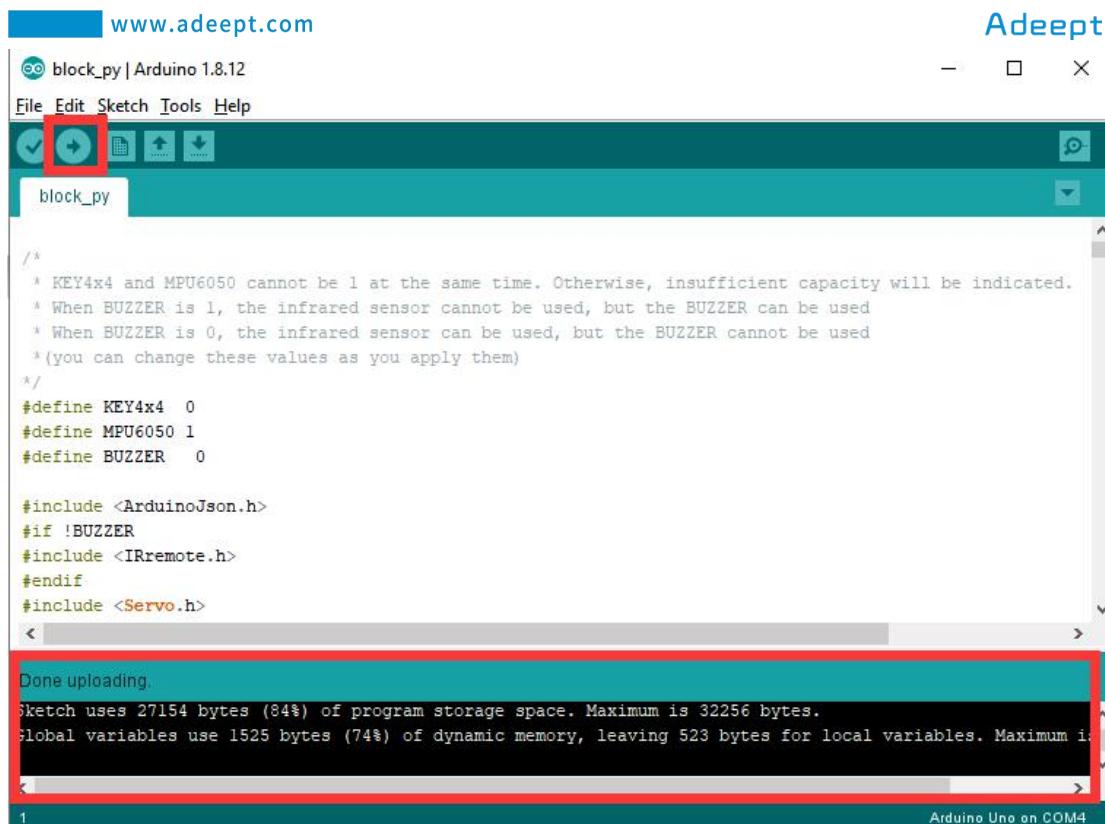
```

2. Using graphical code blocks to program and measure temperature on Arduino UNO

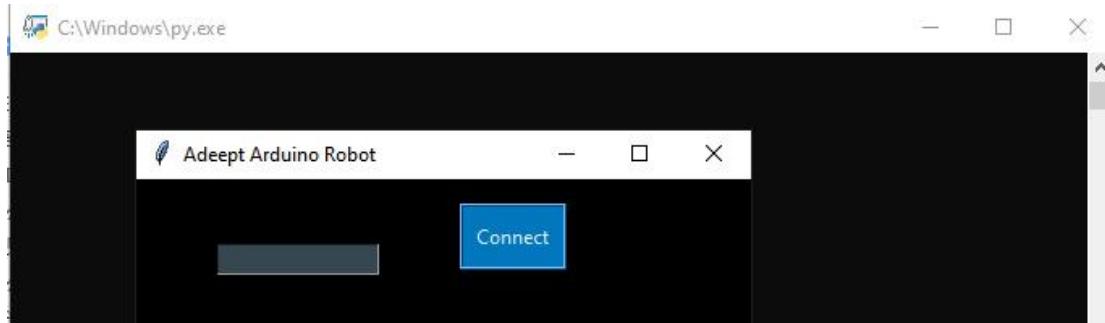
(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

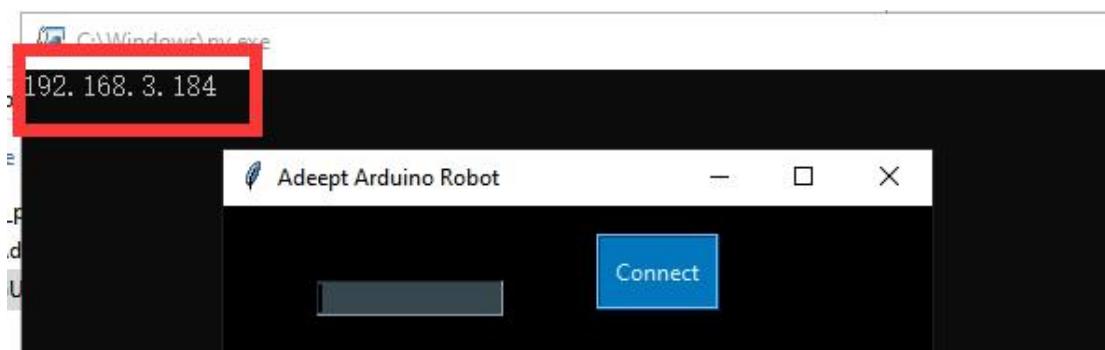
1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:



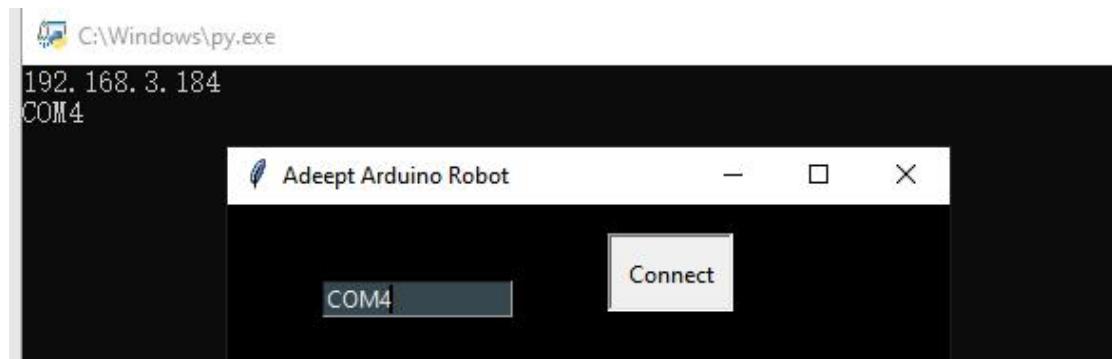
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3.Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



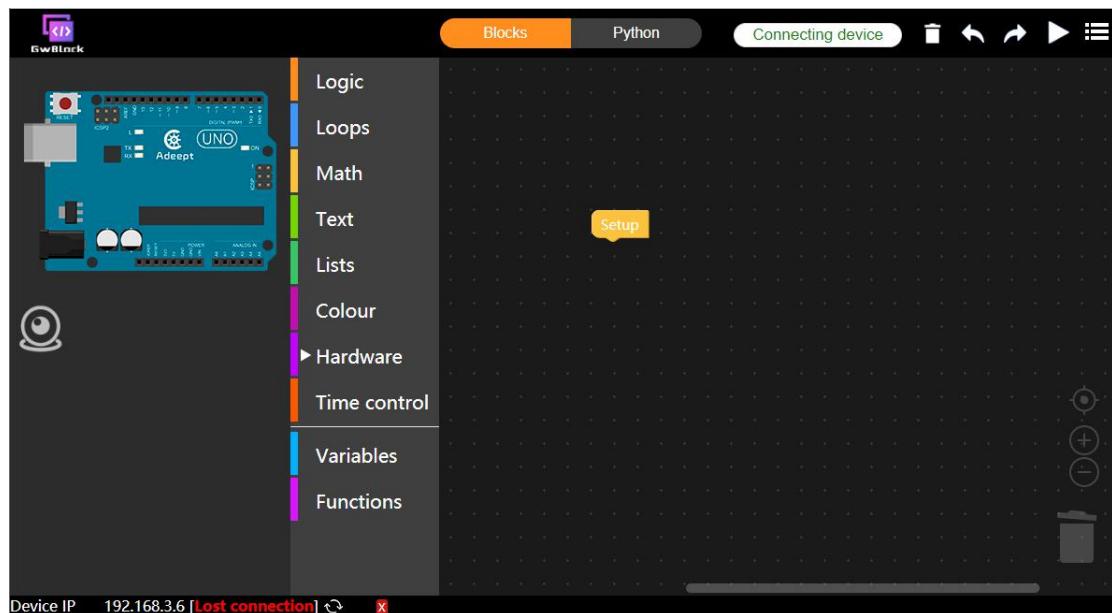
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



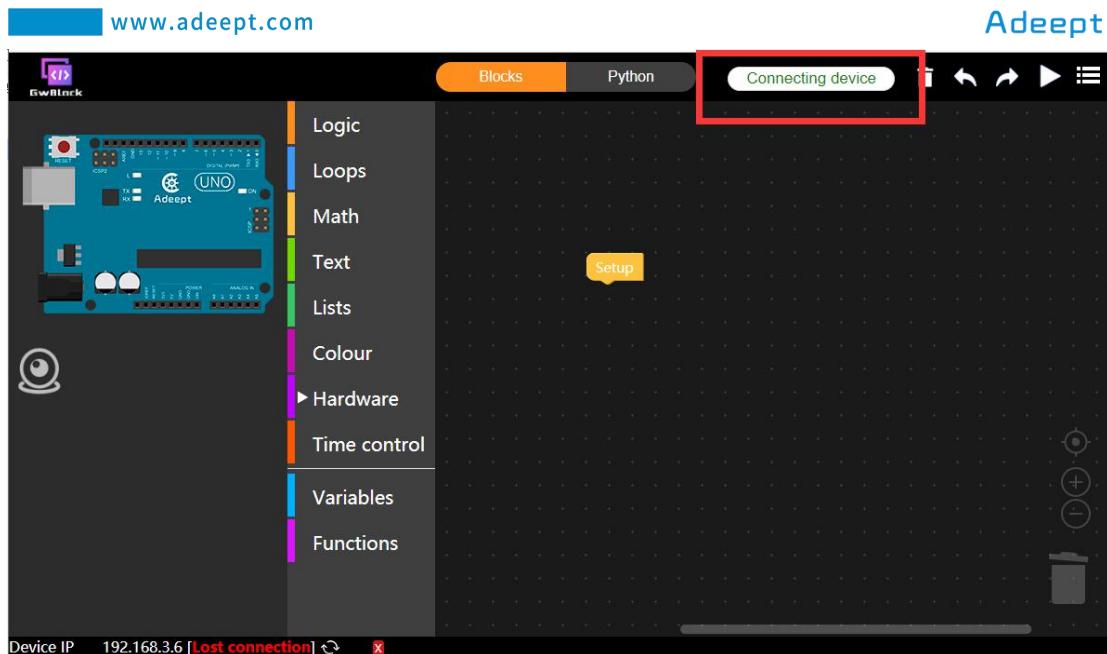
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

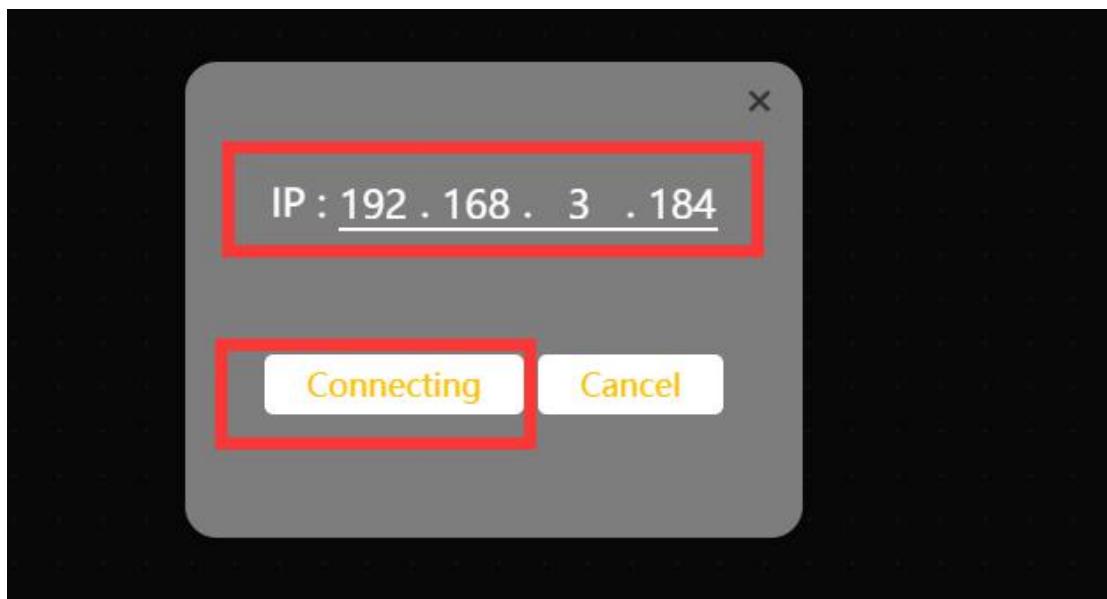
After successfully entering the website, the interface is as follows:



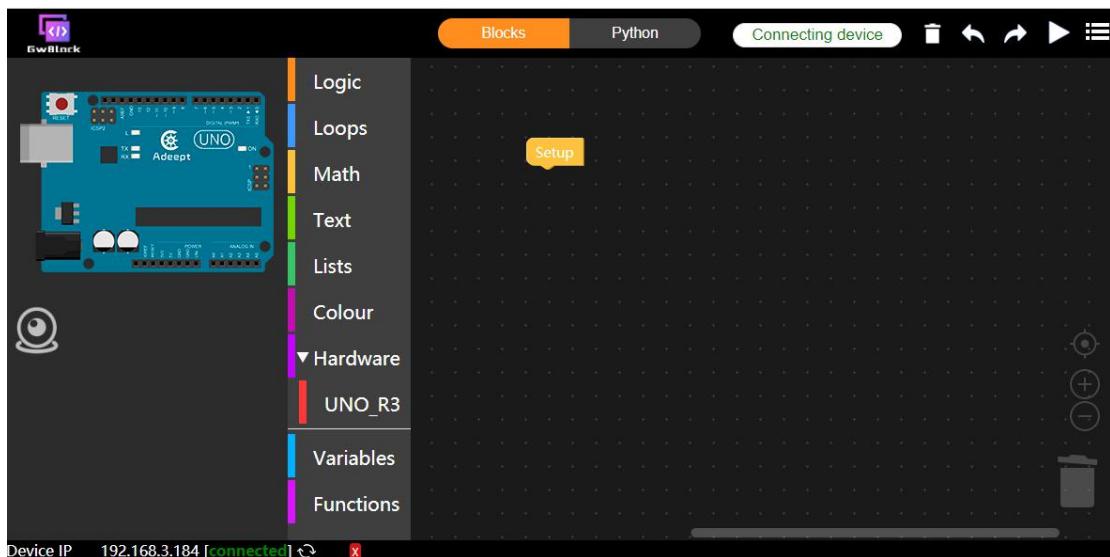
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

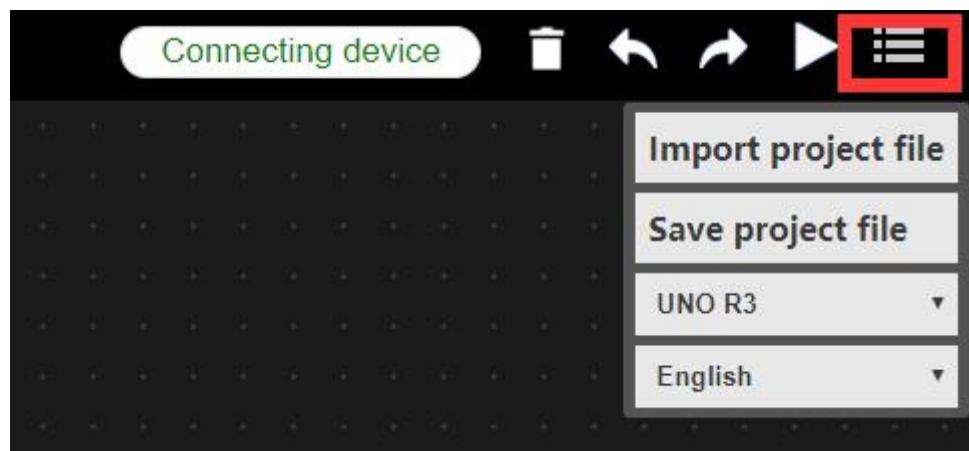


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

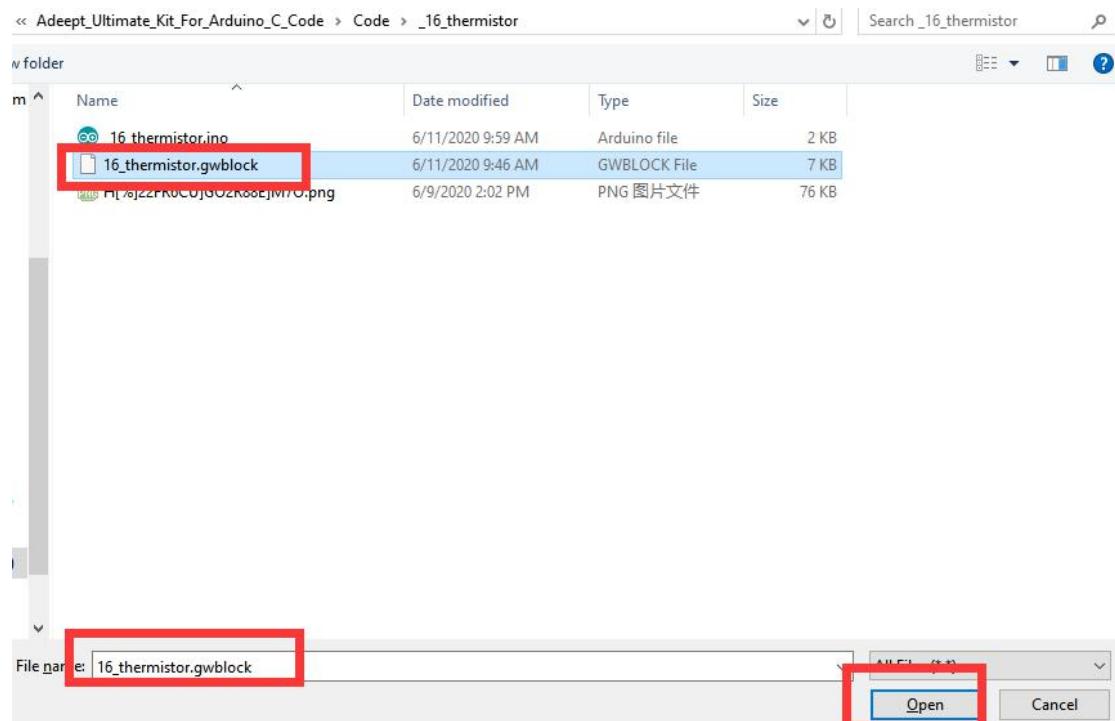
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_16_thermistor

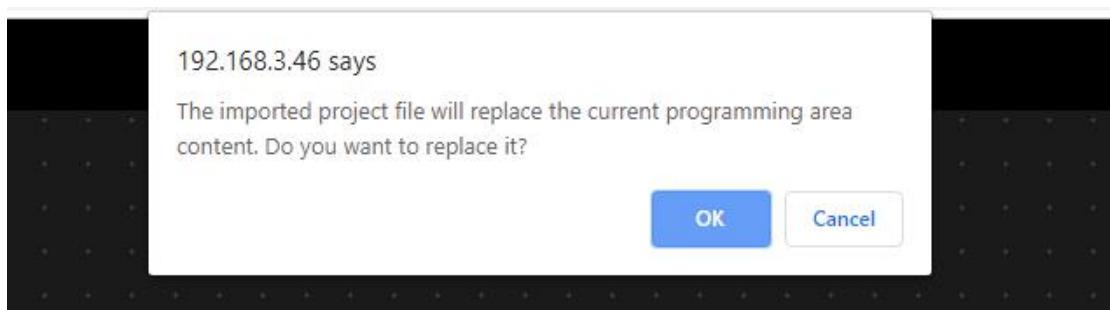
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

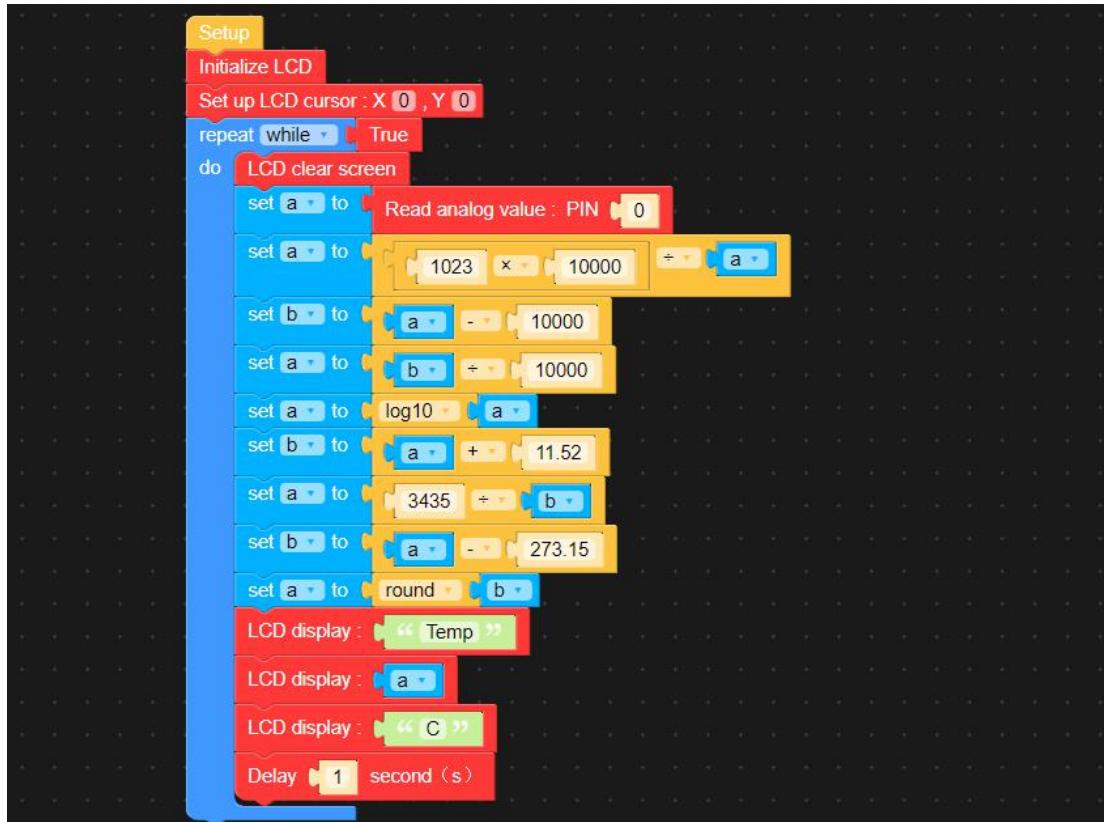
5. Select the "16_thermistor.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



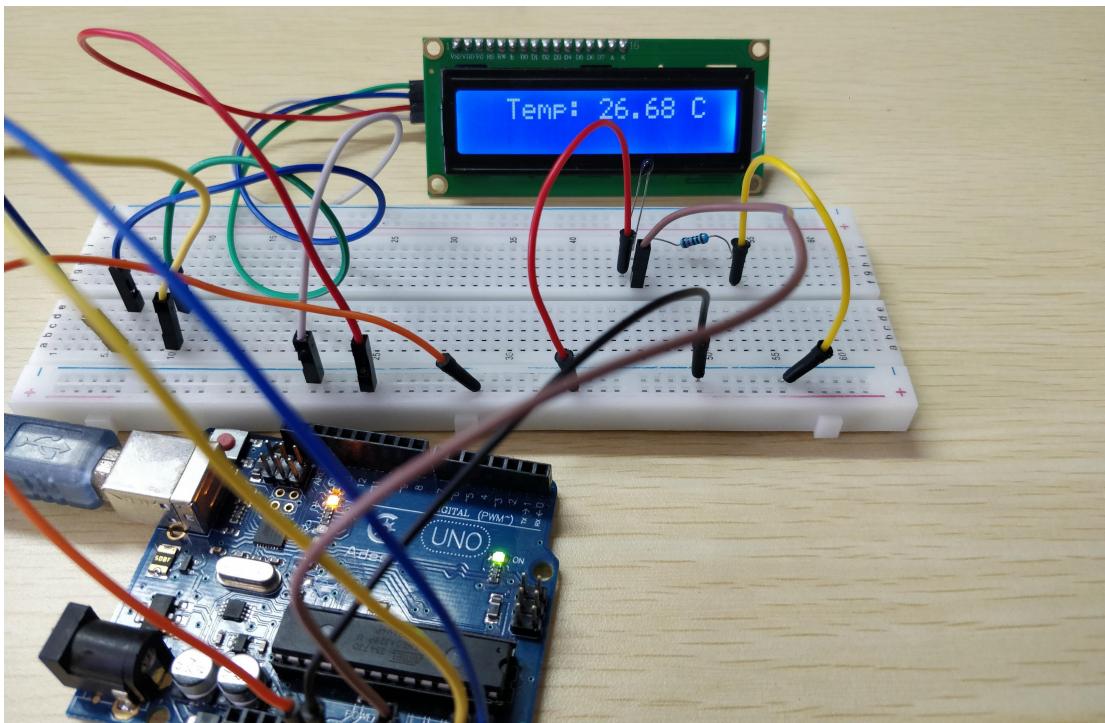
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



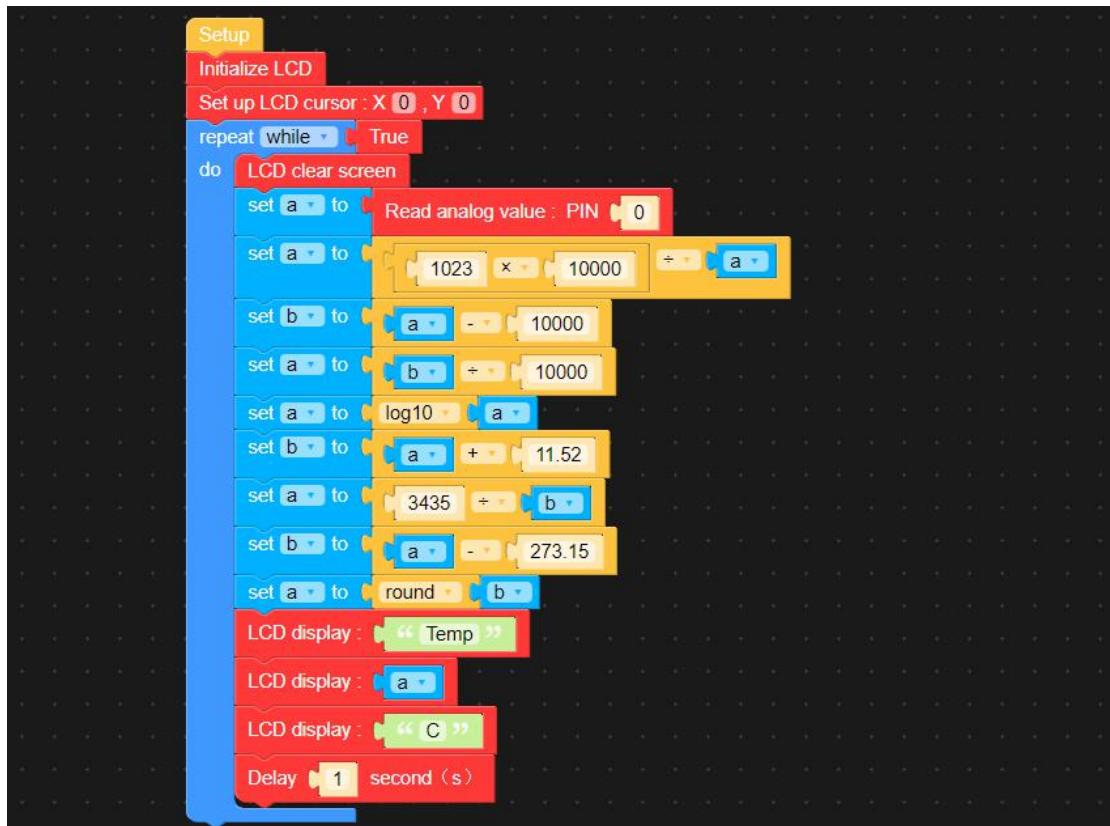
8. Click the button  on the upper right corner. After successfully running the program, we will observe the current temperature detected on the LCD1602 screen. This shows that our experimental test is successful. The physical connection of the experiment is as below:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to read the temperature value of the thermistor on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

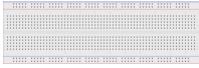
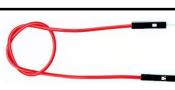
In the GwBlock graphical editor, all code programs are executed from **Setup**. Initialize the LCD through the instruction **Initialize LCD**. In the while loop, the instruction block **set a to Read analog value : PIN 0** will store the read thermistor data to variable a. After calculation, the thermistor data will be output to the LCD1602 screen through the instruction block **LCD display : a**.



Lesson 17 The Application of IR Remoter Controller

In this lesson, we will learn how to use the IR Remoter Controller.

1. Components used in this course

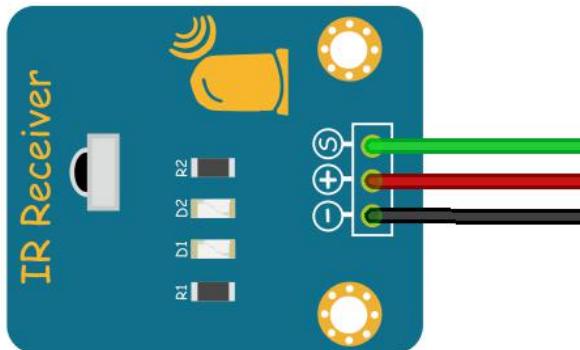
Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
IR Receiver HX1838	1	
Remote Controller	1	
Male to Female Jumper Wires	3	

2. The introduction of receiver and remote controller

(1)IR receiver and IR Remoter Controller

IR Remoter Controller is mainly composed of infrared transmitter (infrared remote controller) and IR receiver.

The IR receiver HX1838 can receive signals from an infrared (IR) remote controller. It has only three pins: signal, VCC and GND. So it is simple to connect with an Arduino board.



The following figure shows an IR remote controller:



(2) Working principle

The signal from the IR Remoter Controller is a series of binary pulse codes. In order to protect it from the interference of other infrared signals during wireless transmission, it is usually modulated on a specific carrier frequency before being emitted through the infrared emitting diode, and the infrared receiving device must filter out other clutter. In addition, the signal of this specific frequency is received and restored to a binary pulse code. This process is demodulation.

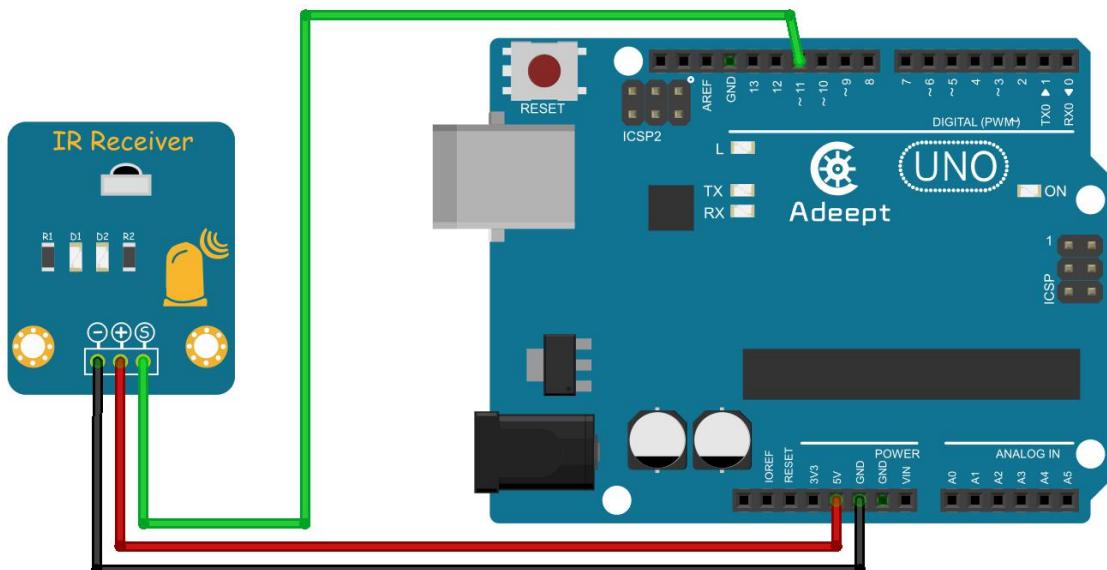
The built-in receiver tube converts the light signal emitted by the infrared emission tube into a weak electrical signal. This signal is amplified by the IC's internal amplifier, and then restored to the remote control transmitter after automatic gain control, band pass filtering, demodulation, and waveform shaping. The original code is input to the code recognition circuit on the appliance via the signal output pin of the receiving head. When the remote control button is pressed, the remote control sends out an infrared carrier signal, and the infrared receiver receives the signal, and the program decodes the carrier signal. We will judge which key is pressed by the difference of the data code.

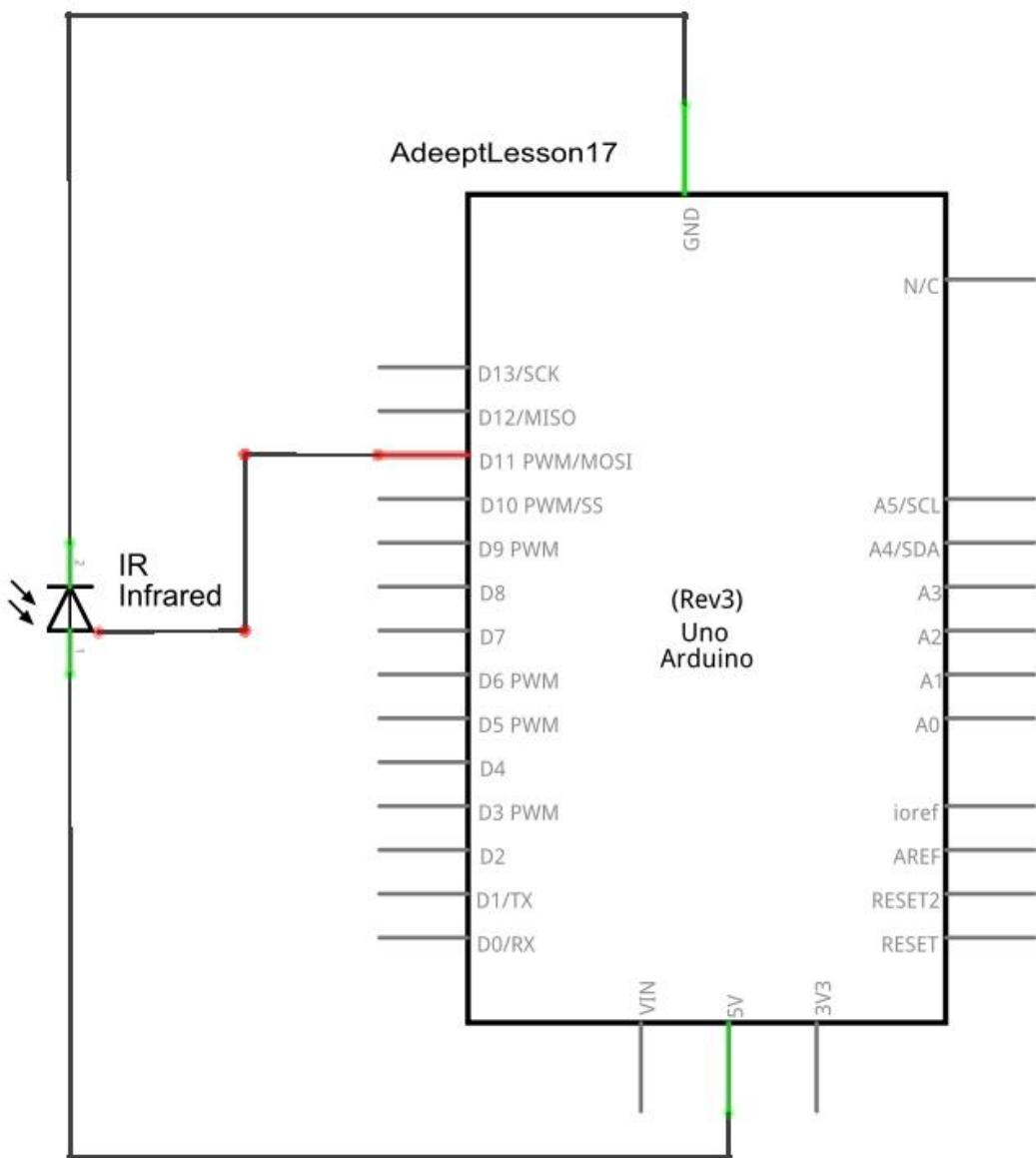
In this experiment, we program the Arduino board to receive the infrared signals,

and then send the received data to Serial Monitor. In the program, we use the Arduino-IRremote-master library (provided).

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:





4.How to use IR Remoter Controller

We provide two different methods to control the IR Remoter Controller. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

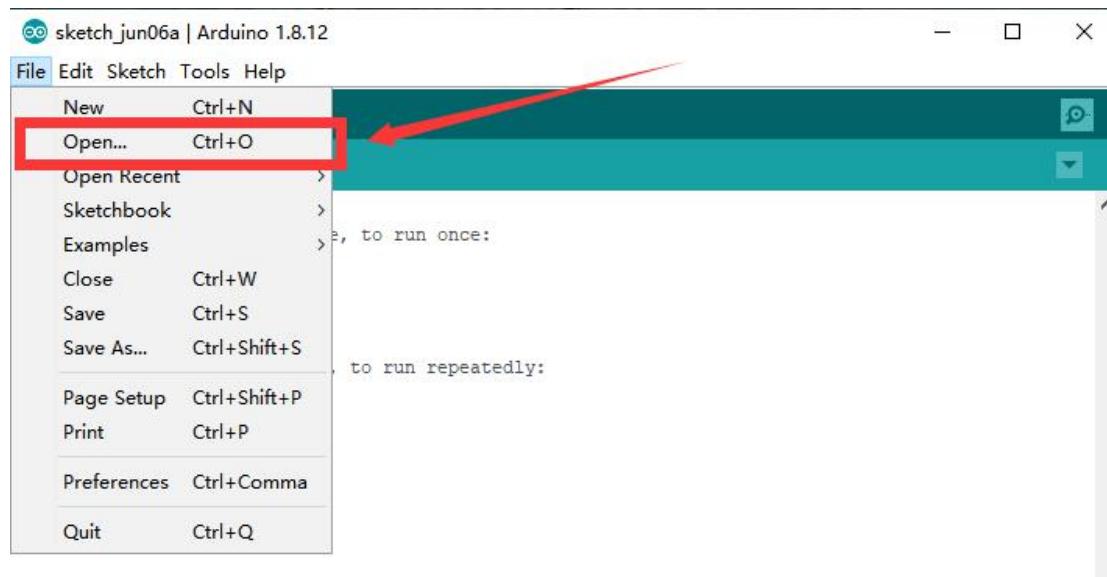
1.Using C language to program to control the IR Remoter Controller on Arduino UNO

(1)Compile and run the code program of this course

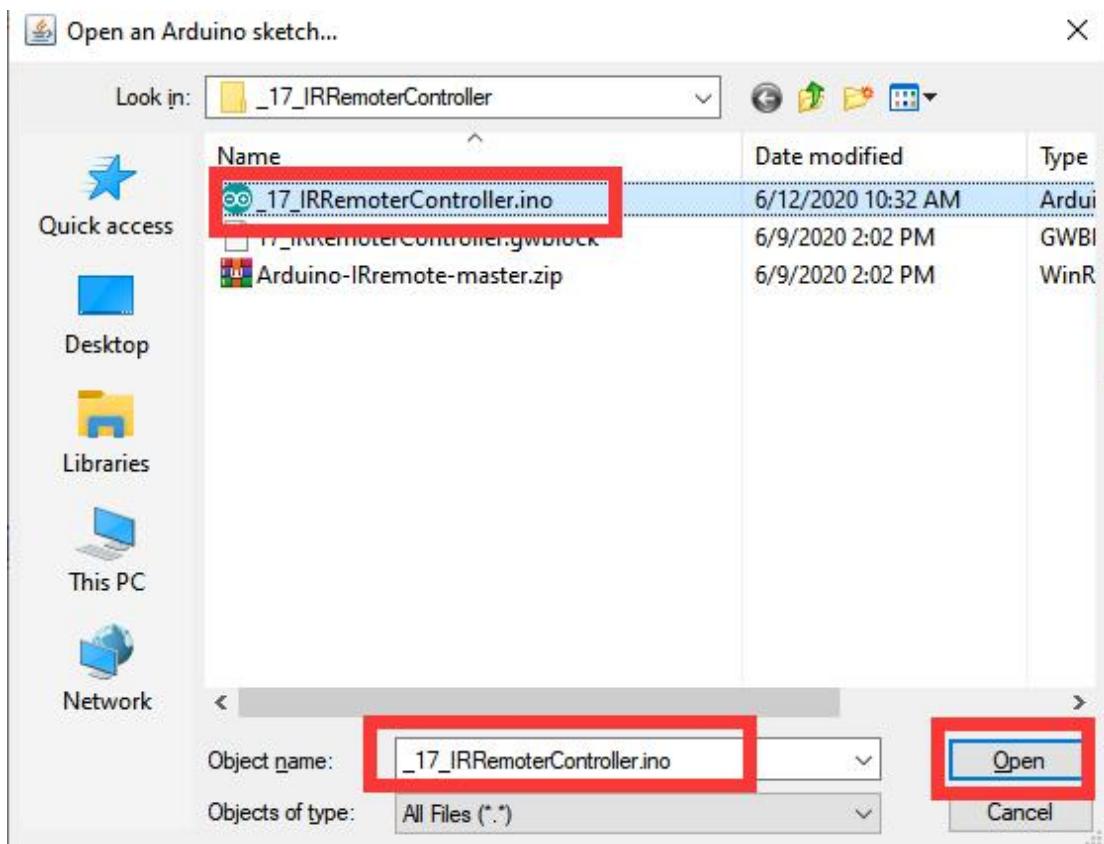
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\17_IRRemoterController directory. Select _17_IRRemoterController.ino. This file is the code program we need in this course. Then click Open.



4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

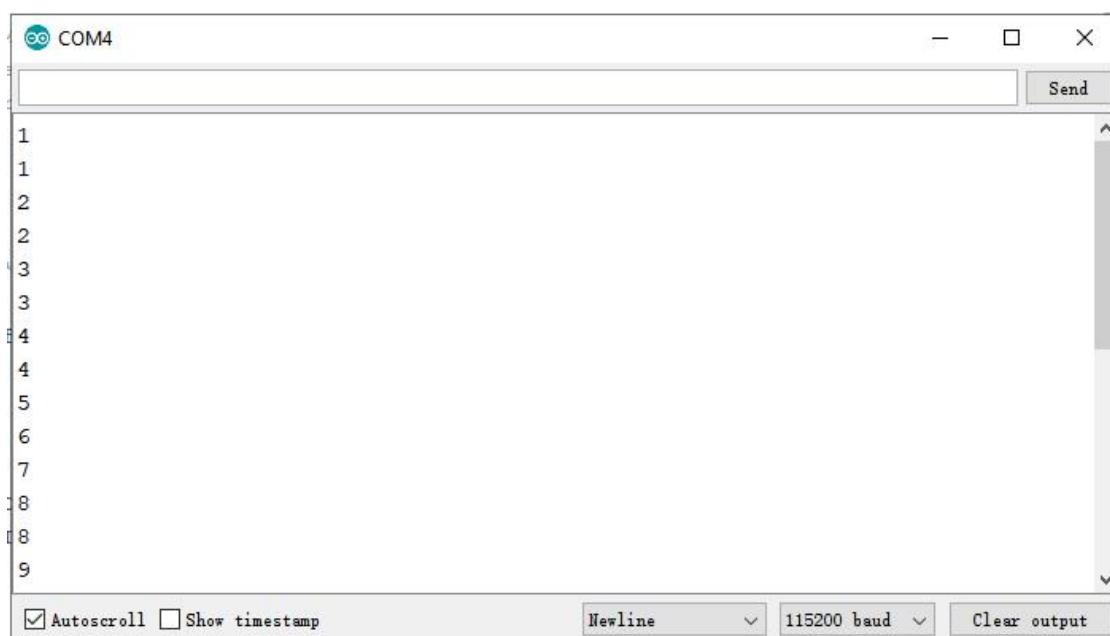
Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                                         Arduino Uno on COM4
```

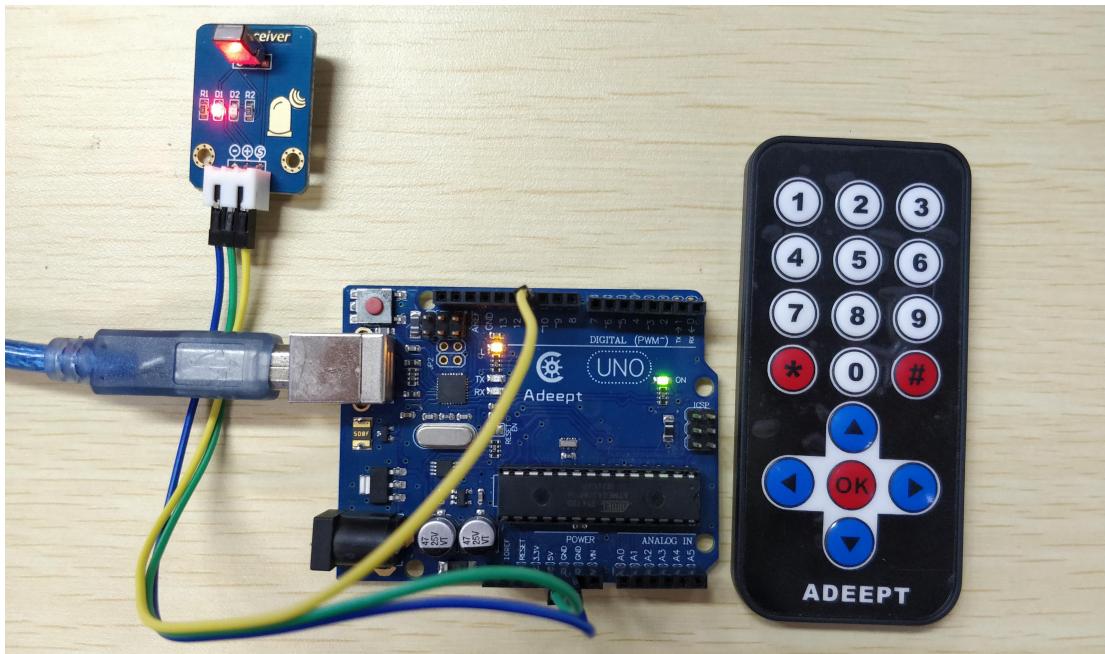
5. After successfully running the program, we need to open the serial monitor on the Arduino software. When we press the button on the infrared remote control, the serial monitor will print out the corresponding number. How to open the serial monitor? You need to click the "Serial Monitor" button  in the upper right corner, as shown below:



6. After clicking , the serial monitor window will pop up. At this time, you need to press the button on the IR Remoter Controller, and the corresponding button number will be printed out in the serial monitor window. As shown below:



7. The physical connection diagram of the experiment is as below:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to control the IR Remoter Controller on Arduino UNO. We will introduce how our core code can be achieved:

1. Open the serial port through `Serial.begin(115200)` in the `setup()` method, and initialize the IR receiver with the `irrecv.enableIRIn()` function.
2. In the `loop()` function, use the `switch_irr(results.value)` function to decode the signal value of the IR receiver, and the `Serial.println(irrecv_data)` function will print the decoded value on the serial port. .

```
void setup()
{
    Serial.begin(115200); //Open serial
    irrecv.enableIRIn(); // Initialization infrared receiver
}
void loop()
{
    int irrecv_data = 0;
    if (irrecv.decode(&results)) { // Decoding successful, received a set of infrared signal
        irrecv_data = switch_irr(results.value);
        Serial.println(irrecv_data); //Wrap output in hex receive code
        //Serial.println(); //For ease of viewing the output to add a blank line
        irrecv.resume(); //Receiving the next value
    }
}
```

2. Using graphical code blocks to program and control the IR

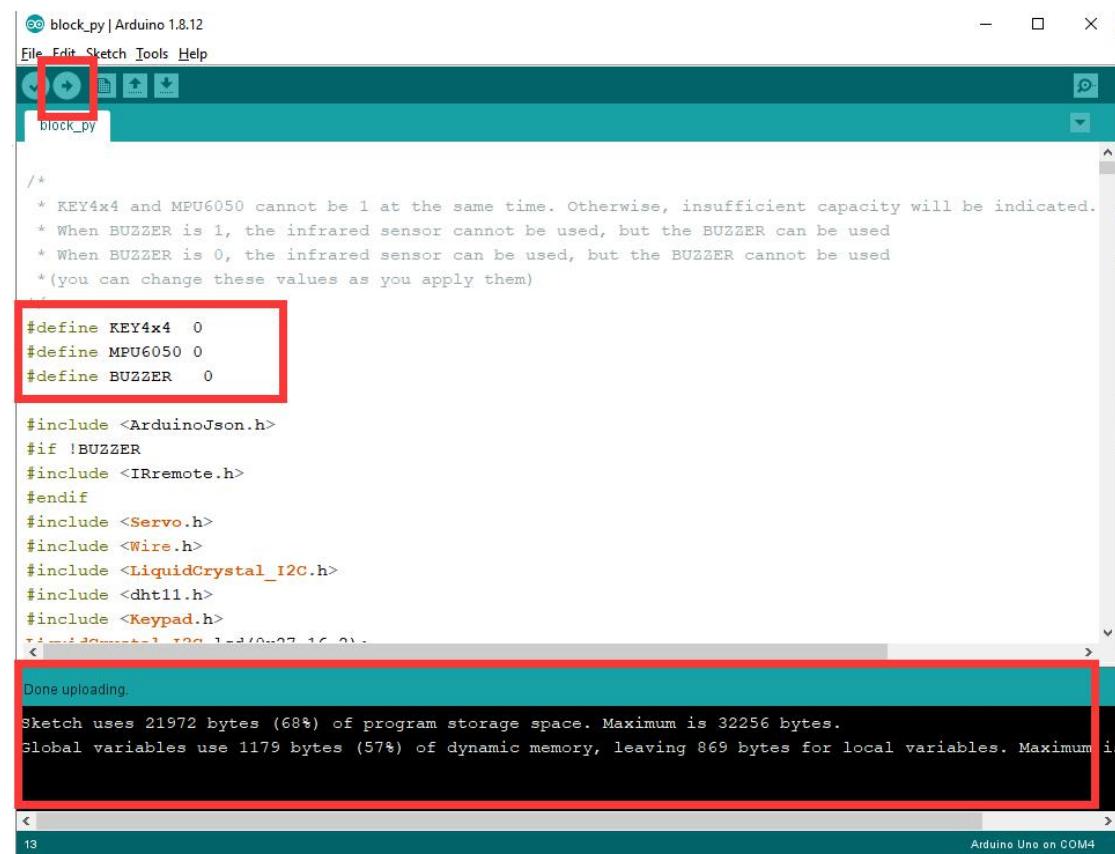
Remoter Controller on Arduino UNO

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1.Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). [Note]: We need to modify #define MPU6050 1 in the opened block_py.ino file to #define MPU6050 0, and #define BUZZER 1 to #define BUZZER 0.Then click  and upload the program to the Arduino UNO.

After successful Upload, it will show as below:



```

block_py | Arduino 1.8.12
File Edit Sketch Tools Help
block_py

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */

#define KEY4x4 0
#define MPU6050 0
#define BUZZER 0

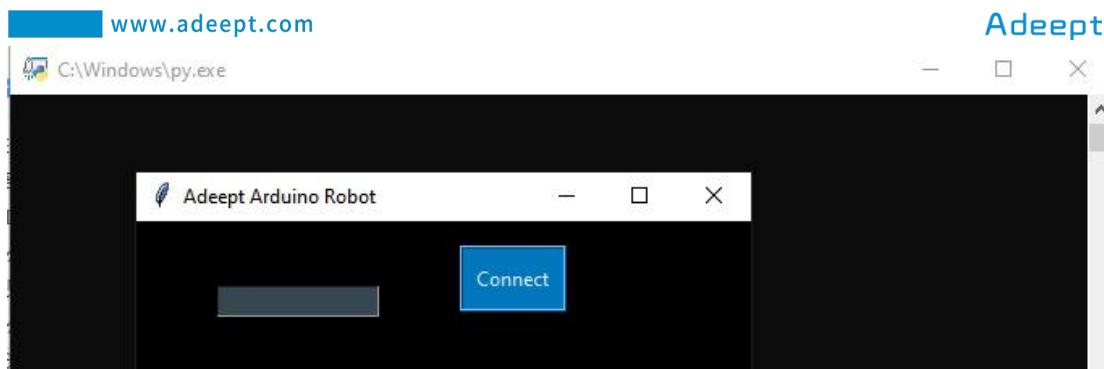
#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <dht11.h>
#include <Keypad.h>
#include <OneWire.h>
#include <DallasTemperature.h>

Done uploading.

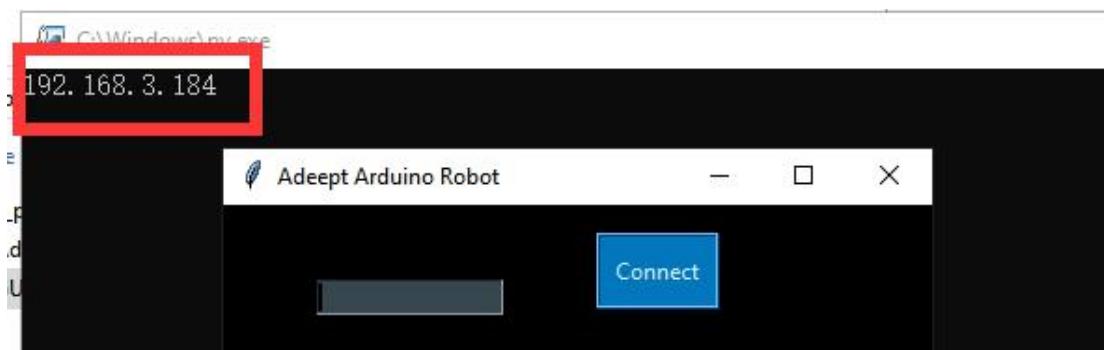
Sketch uses 21972 bytes (68%) of program storage space. Maximum is 32256 bytes.
Global variables use 1179 bytes (57%) of dynamic memory, leaving 869 bytes for local variables. Maximum is

```

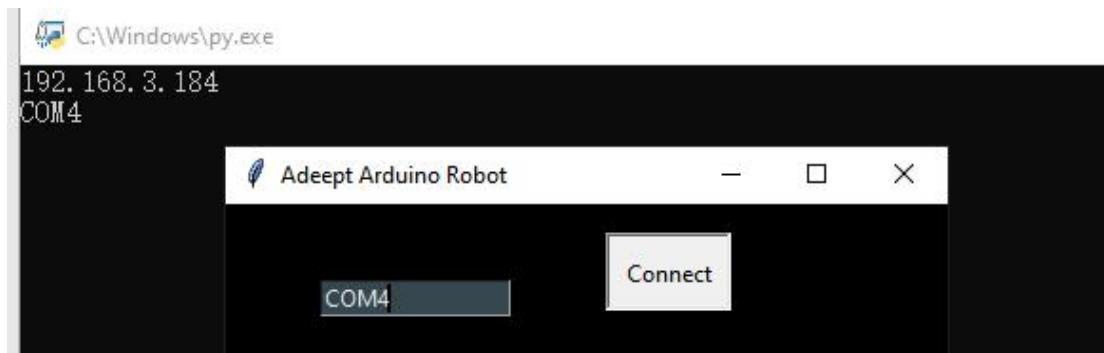
2.Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again.Then open the websocket folder. Double-click to open the GUI info v1.0.py file.It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



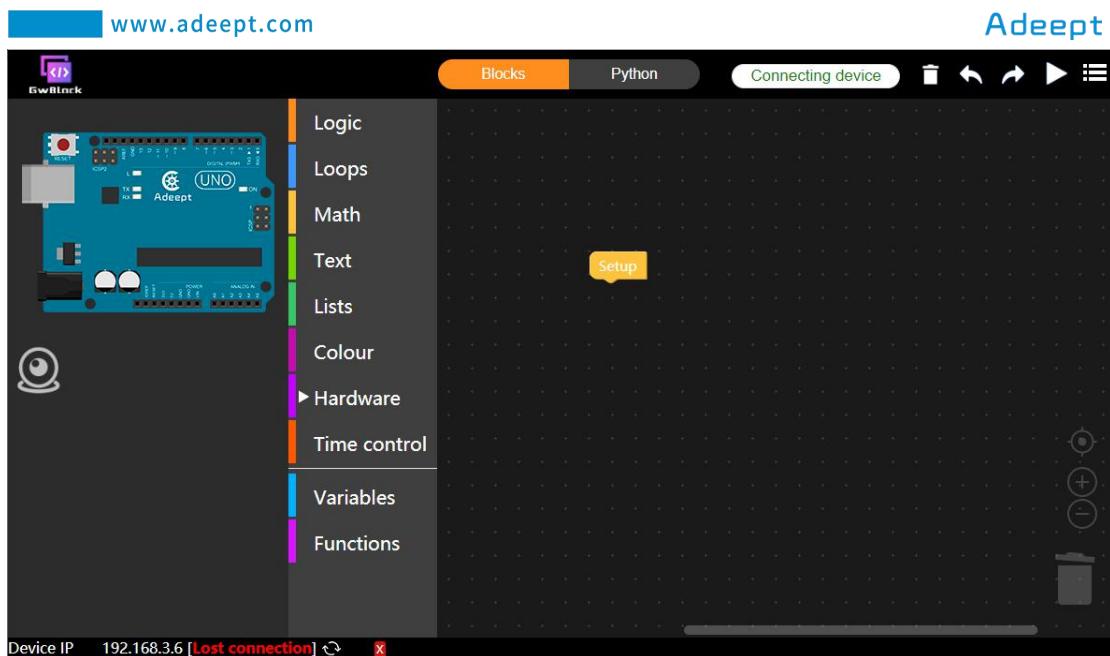
4. Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



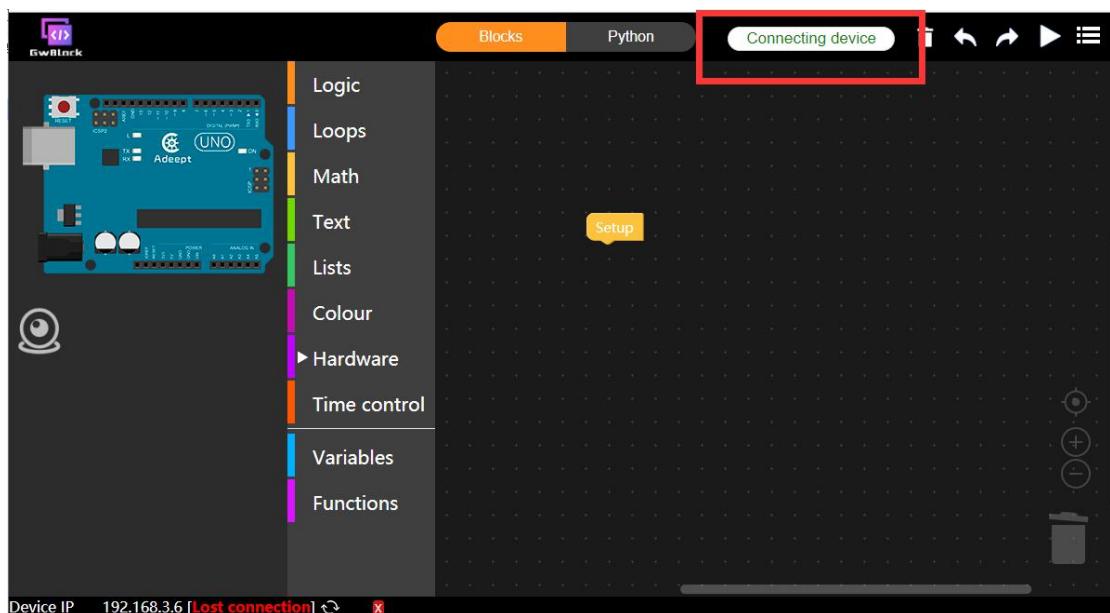
5. Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

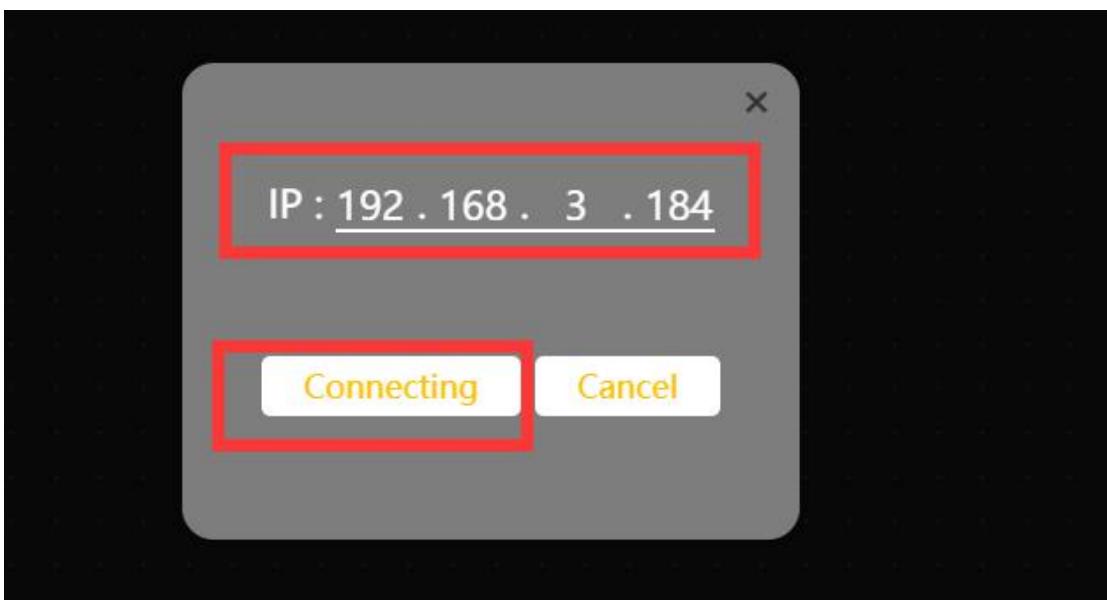
After successfully entering the website, the interface is as follows:



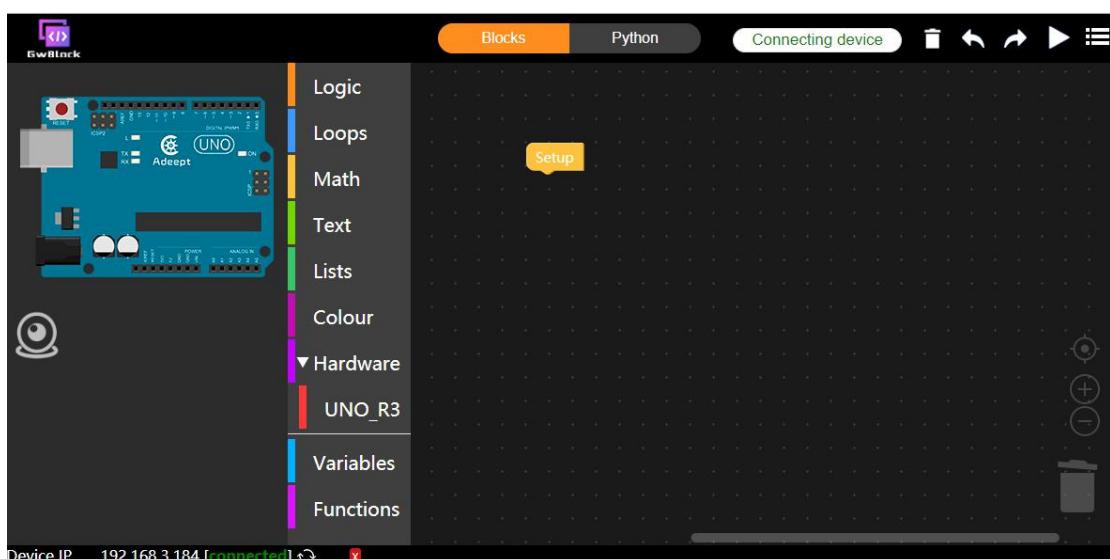
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8. After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:



(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

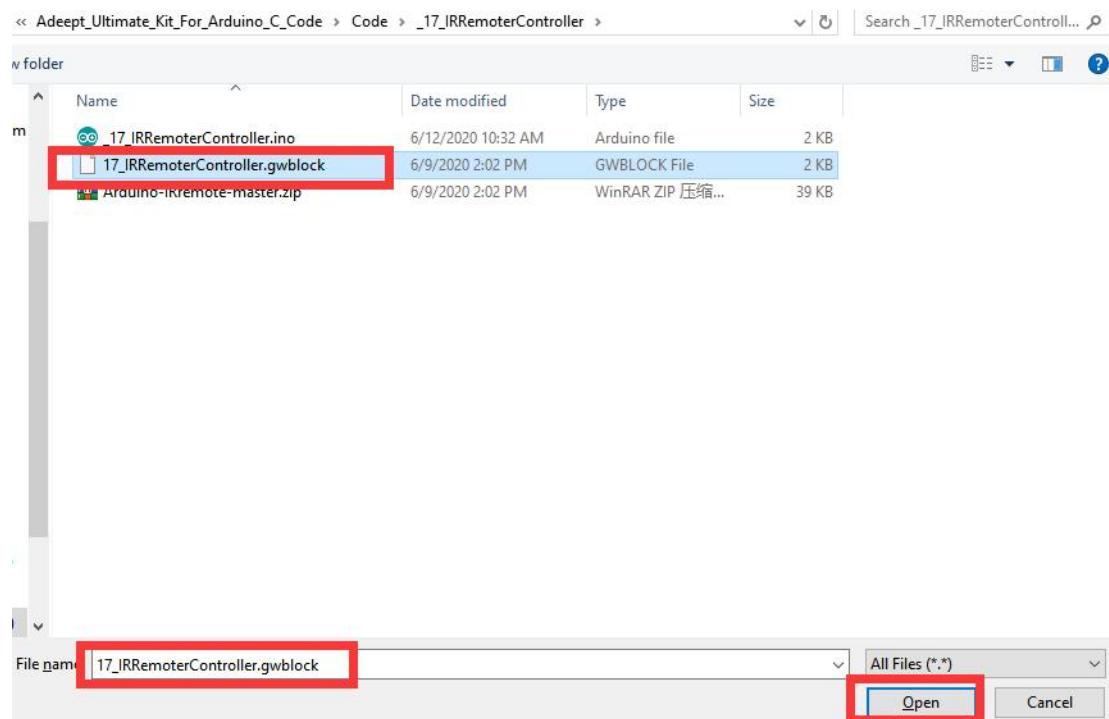
Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_17_IRRemoterController

It will show as below:

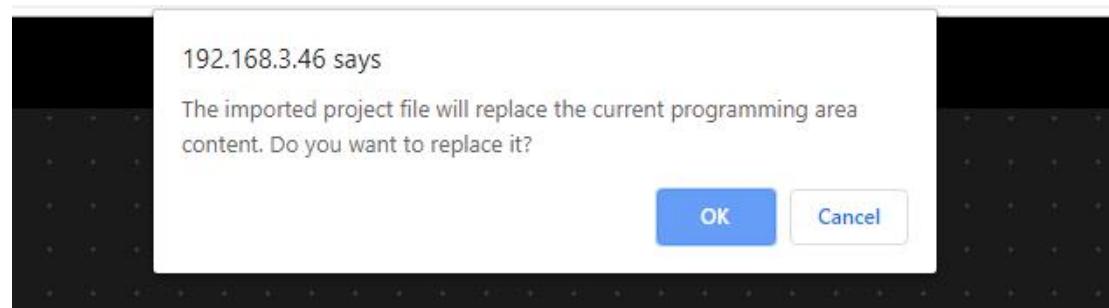
Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

5. Select the "17_IRRemoterController.gwblock" file. This file is the graphical code

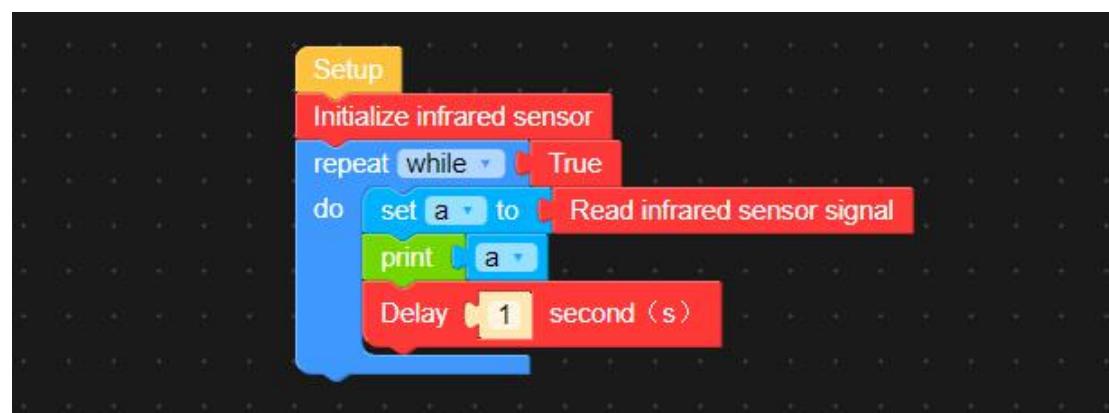
program for our lesson. Click "Open" in the lower right corner. It will show as below:



6. Click OK. It will show as below:



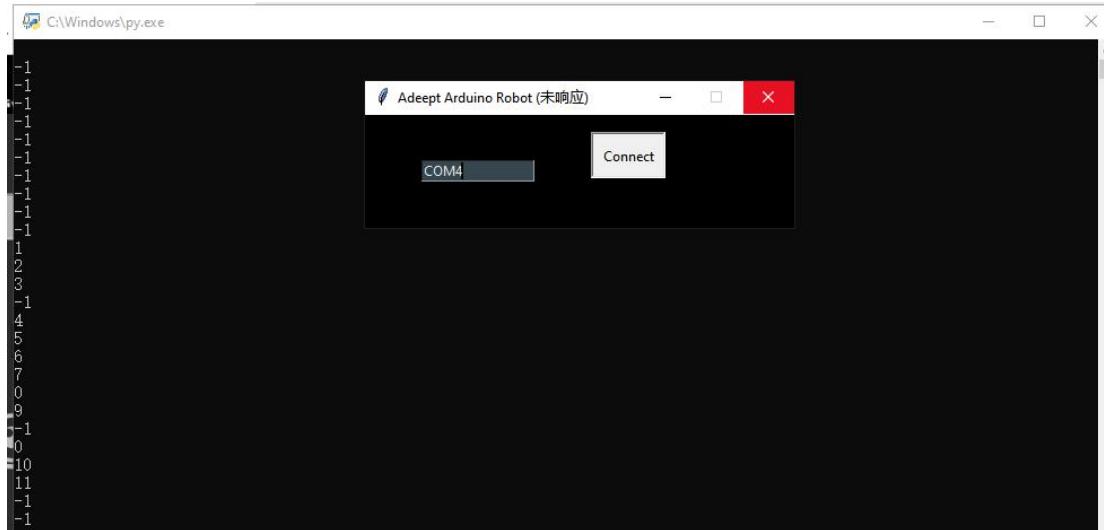
7. It will show as below after successfully opening:



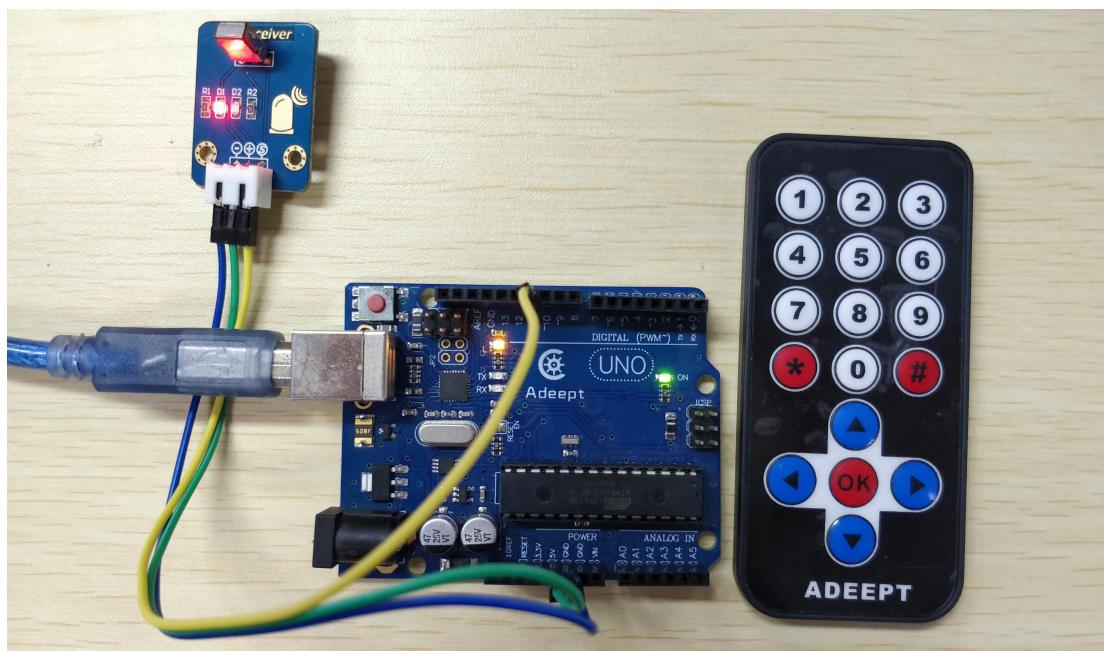
8. Click the button  on the upper right corner. Open the window of GUI info v1.0.py.

If you don't press the button of the IR Remoter Controller, it will print out -1. If you

press the button on the IR Remoter Controller, it will print out the number of the corresponding button. As shown below:



9. The physical connection of the experiment is as below:

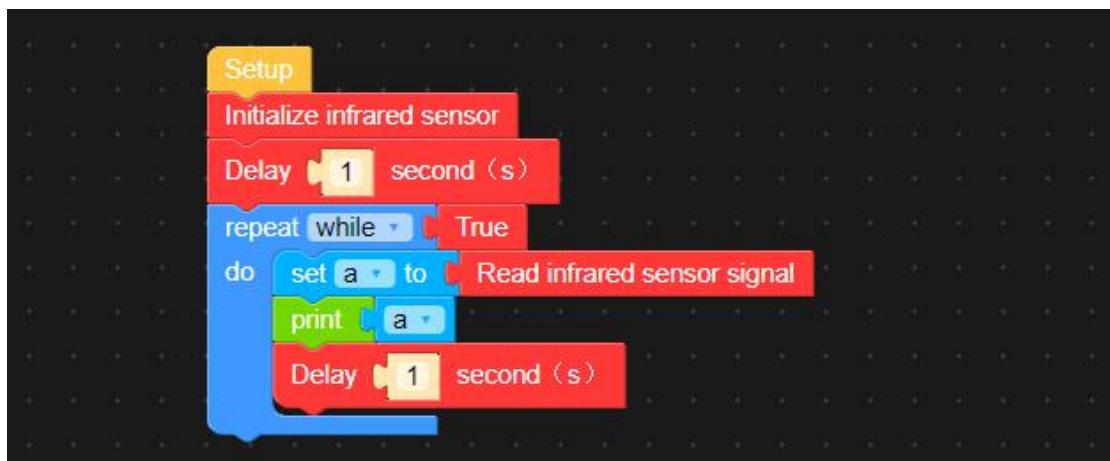


(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to control the IR Remoter Controller on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

In the GwBlock graphical editor, all code programs are executed from **Setup**. Initialize the IR receiver through the instruction **Initialize infrared sensor**. The

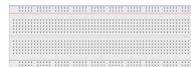
command block `set [a] to [Read infrared sensor signal]` can read the signal value of the IR receiver. Print the corresponding remote controller key value on the command window through the instruction block `print [a]`.



Lesson 18 The Application of DHT-11(Digital Temperature & Humidity Sensor)

In this lesson, we will learn the application of the DHT-11(Digital Temperature & Humidity Sensor).

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
LCD1602	1	
Adeept IIC Module	1	
Male to Female Jumper Wires	Several	
Adeept DHT-11 Temperature & Humidity Sensor	1	

2. The introduction of the DHT-11

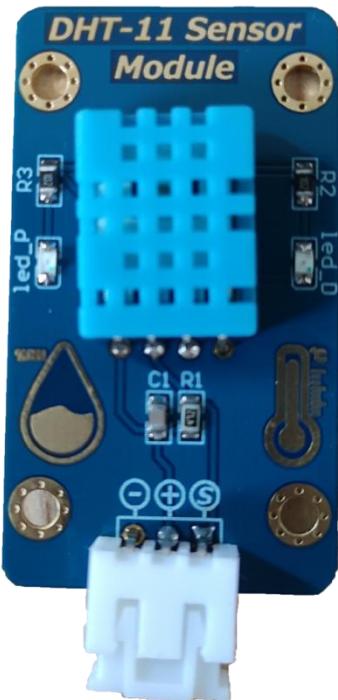
(1) DHT-11(Digital Temperature & Humidity Sensor)

DHT11 is a temperature and humidity sensor with calibrated digital signal output. It can be used to measure humidity and temperature, its precision humidity +5%RH, temperature +2°C, range humidity 20-90%RH, temperature 0~50°C. It uses special

digital module acquisition technology and temperature and humidity sensing technology to ensure the product has high reliability and excellent long-term stability. The sensor consists of a resistive moisture sensing element and an NTC temperature measuring element, and is connected to a high-performance 8-bit MCU. Therefore, this product has the advantages of excellent quality, meteoric response, strong anti-interference ability and high cost performance. Each DHT11 sensor is calibrated in a highly accurate humidity calibration chamber. Calibration coefficients are stored in the OTP memory in the form of a program, and these calibration coefficients are called by the sensor during the processing of the detection signal. Single-wire serial interface makes system integration easy and fast. Its ultra-small size and very low power consumption make it the best choice for this kind of applications in demanding applications.

Notes:

- (1) Avoid the use of condensation.
- (2) Long-term storage conditions: temperature 10-40°C, humidity below 60%
- (3) Measurement Range: 20-95%RH, 0-50°C
- (4) Humidity Accuracy: ±5%RH
- (5) Temperature Accuracy: ±2°C
- (6) '+': VCC (3.3~5.5V)
- (7) '-': GND (0V)
- (8) 'S': data pin



(2) The features of DHT-11(Digital Temperature & Humidity Sensor)

DHT11 uses the single bus protocol, which transmits 40 bits of data at a time, i.e., 40 bits of data. Every time the data of DHT11 is read, it should be read 40 times at a time, i.e., read 40 bits. Data format : 40bit data = 8-bit humidity integer + 8-bit humidity decimal + 8-bit temperature integer + 8-bit temperature decimal + 8-bit calibration. The first 16 bits of data are related to humidity, the middle 16 bits are related to temperature, and the last eight bits are used for calibration.

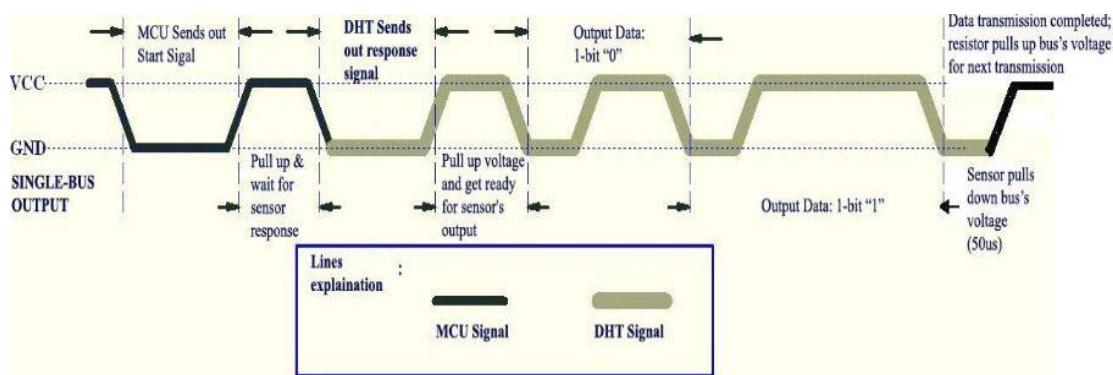
(3) Principle of reading data of DHT-11(Digital Temperature & Humidity Sensor)

The DHT-11(Digital Temperature & Humidity Sensor) we use in this lesson only has 3 pins, and the 3 pins are used as VCC (+), GND (-), and DATA (out), respectively. Since only high and low levels are transmitted to Raspberry Pi GPIO, how can we

read the temperature and humidity Numbers? With fewer pins, it needs a sequence signal of high and low variation to express the value, as well as other signals such as the start signal and so on. Let's first understand the timing signal of DHT11:

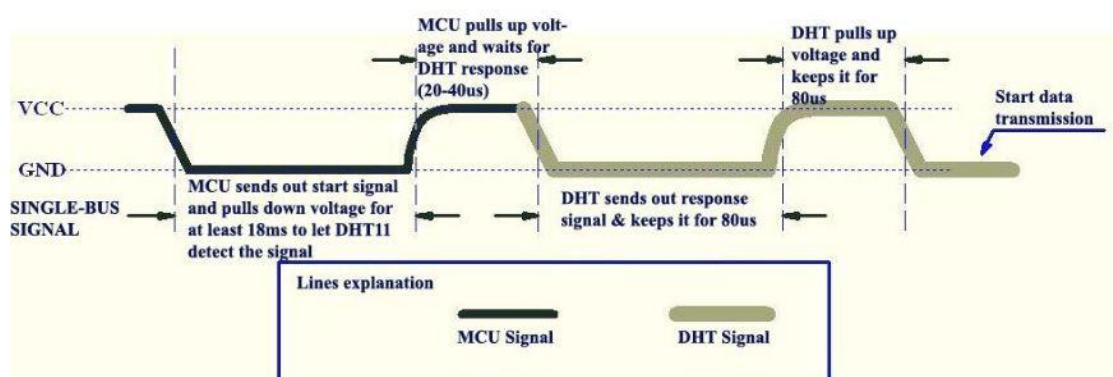
(1) Data frame format

DHT11 will send 40 bits (5 bytes) of data to the host. The first and second bytes of data represent the temperature value. 3,4 bytes of data represent the humidity value; The fifth byte data is the check code: data format :40bit data =8 bit humidity integer +8 bit humidity decimal +8 bit temperature integer +8 bit temperature decimal +8 bit check. If the data is correct, the sum of the first four bytes equals the last checksum.



(2) Handshake stage

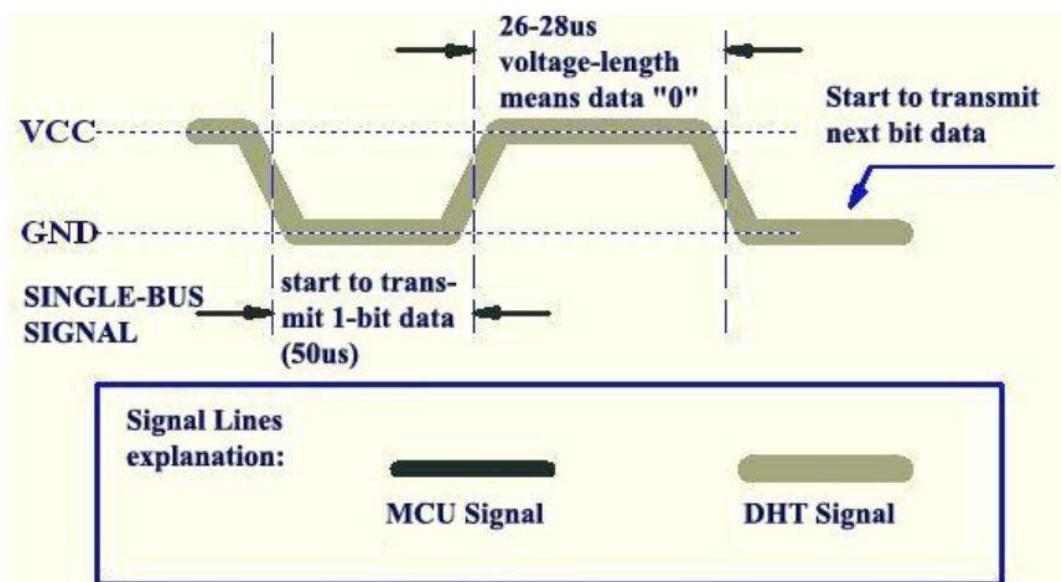
By default, the DATA (out) foot is at a high level, and the host GPIO sends the start signal. First, pull down the DATA foot at least 18ms, and then pull up the DATA foot 20-40us to wait for the DHT11 response signal. Once DHT11 receives the start signal, DHT11 will send a response signal to the host, at the same time, pull the DATA foot down 80us as the response, and then DHT11 will pull up the DATA foot 80us. Then it's done.



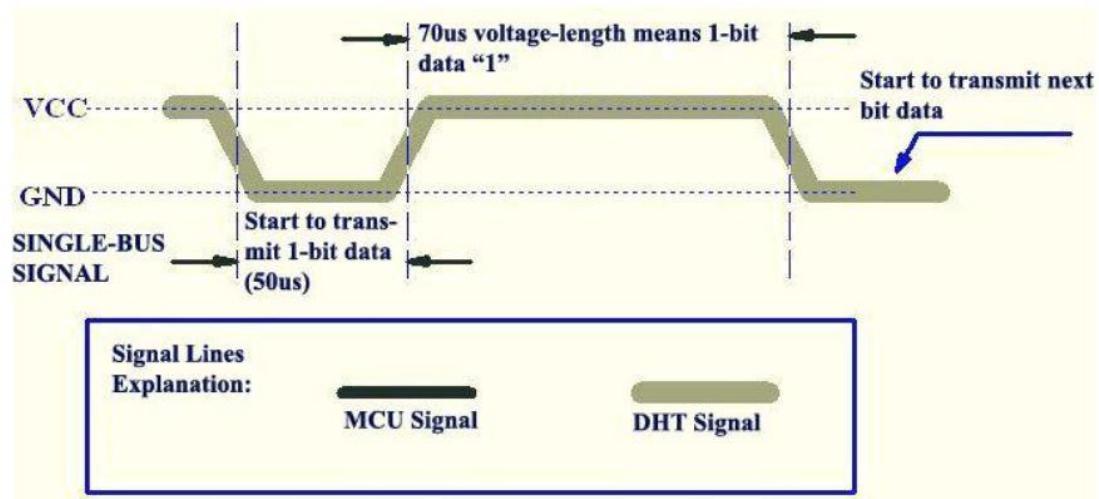
(3) Data transmission stage

For sending humidity and temperature data once, DHT11 needs to send 40bits of data. Each bit of data starts with a 50us low level, followed by a high-level timing signal. A continuous 26us-28us means that the bit is 0. A continuous 70us means that the bit is 1, and then continues with a 50us low level, followed by a high level of the next bit.

Data "0" :



Data "1" :



(4) Data reception and verification

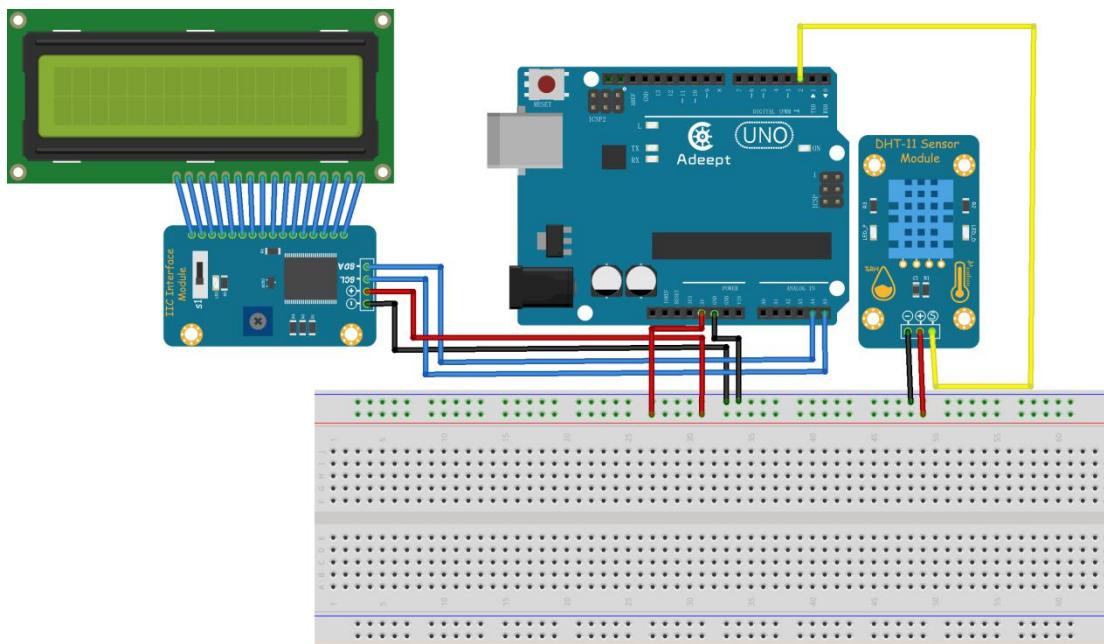
The 40bit data is encoded as below:

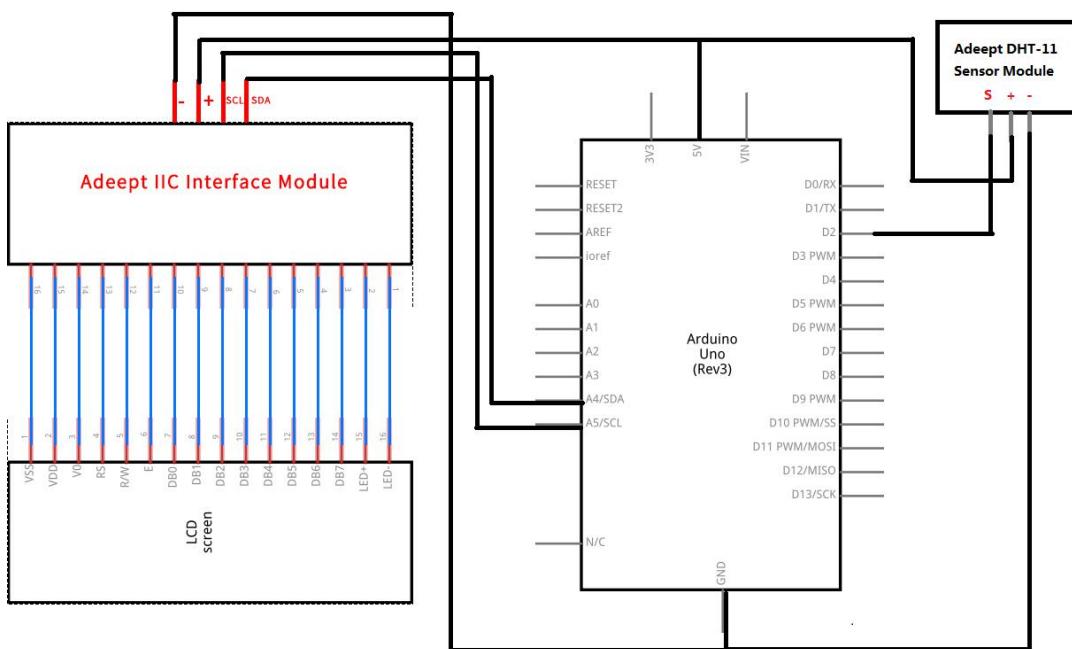
1	byte4	byte3	byte2	byte1	byte0
2	01010101	00000000	10101010	00000000	01010101
3	-----	-----	-----	-----	-----
4	Integer	Decimal	Integer	Decimal	Check Digit
5	-----	-----	-----	-----	-----
6	Humidity		Temperature		

In order to ensure the accuracy of the received data, it is necessary to verify the data. If $\text{byte1} + \text{byte2} + \text{byte3} + \text{byte4} == \text{byte0}$, the received data will be correct. But the DHT11 decimal doesn't work, so you have to worry about byte2 plus byte4. After successful verification, we will verify the correctness of data reading results of the temperature and humidity.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:





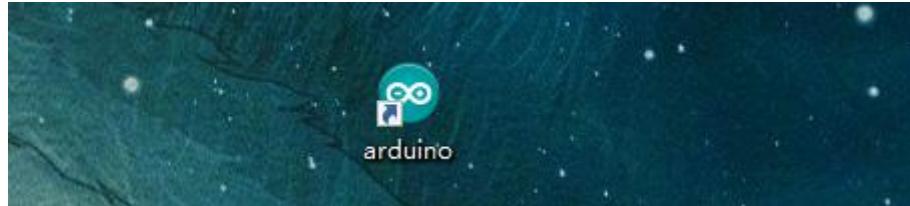
4.Read the temperature and humidity value of DHT11 sensor

We provide two different methods to read the temperature and humidity value of DHT11 sensor. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

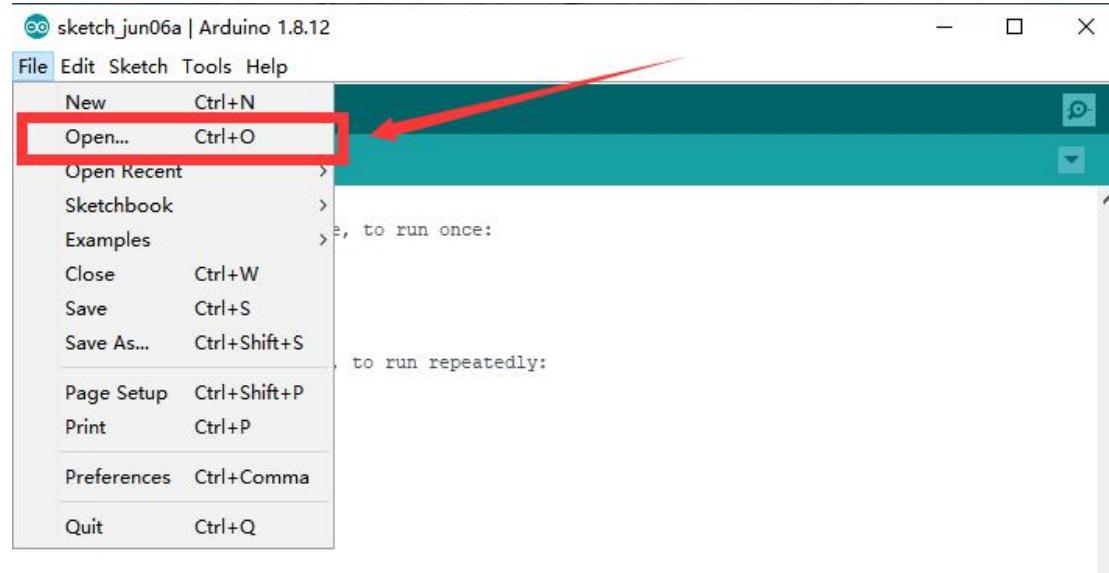
1.Using C language to program to read the temperature and humidity value of DHT11 sensor on Arduino UNO

(1)Compile and run the code program of this course

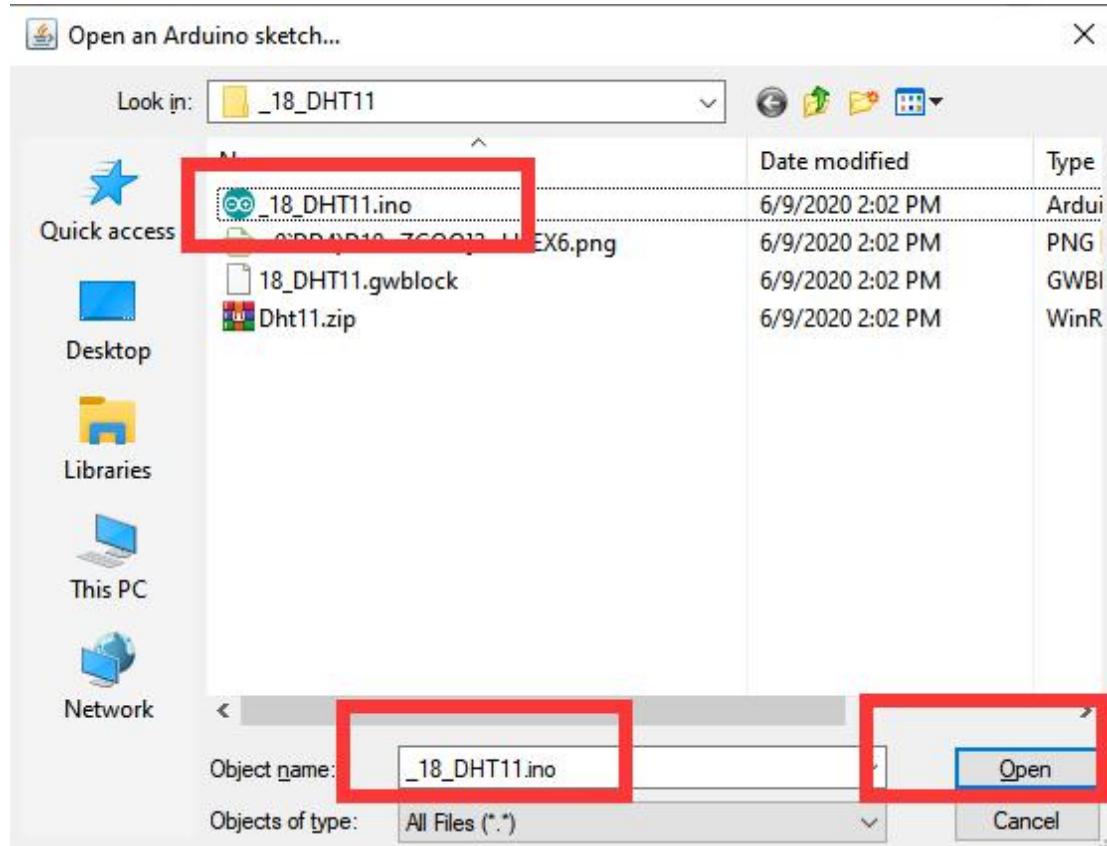
1.Open the Arduino IDE software, as shown below:



2.Click Open in the File drop-down menu:



3.Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\18_DHT11 directory. Select_18_DHT11.ino. This file is the code program we need in this course. Then click Open.



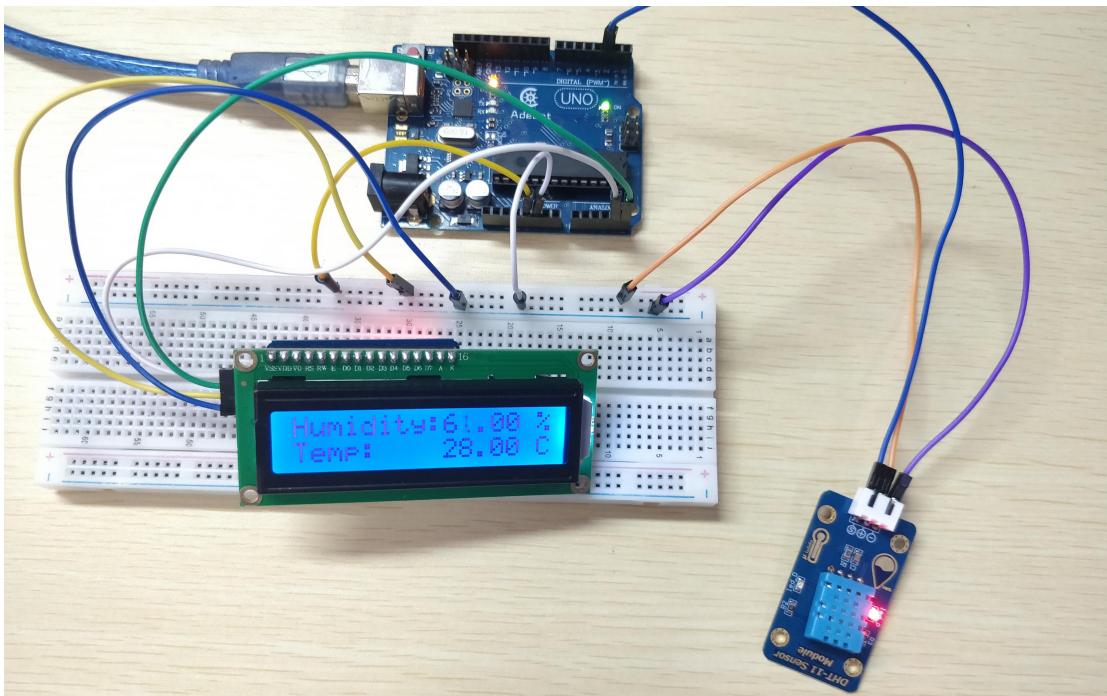
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After successfully running the program, we will observe that the Humidity and Temp values are detected by the DHT11 sensor on the LCD1602 screen, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to read the temperature and humidity value of DHT11 sensor on Arduino UNO. We will introduce how our core code can be achieved:

1.In the setup() function, the LCD1602 is initialized by lcd.init(), and the LCD1602 backlight can be set by the lcd.backlight() function.

```
void setup()
{
    lcd.init(); //initialize the lcd
    lcd.backlight(); //turn on the backlight
    delay(1000); //延时1000ms
}
```

2.In the loop() function, read the DHT11 data through DHT11.read(DHT11PIN); output the humidity (Humidity) and temperature (Temp) data on the LCD1602 screen through the lcd.print() function.

```

void loop()
{
    int chk = DHT11.read(DHT11PIN);
    lcd.setCursor(0, 0); // set the cursor to column 0, line 0
    lcd.print("Humidity:"); // Print a message of "Humidity: " to the LCD.
    lcd.print((float)DHT11.humidity, 2); // Print a message of "Humidity: " to the LCD.
    lcd.print(" %"); // Print the unit of the centigrade temperature to the LCD.

    lcd.setCursor(0, 1); // set the cursor to column 0, line 0
    lcd.print("Temp: "); // Print a message of "Temp: " to the LCD.
    lcd.print((float)DHT11.temperature, 2); // Print a centigrade temperature to the LCD.
    lcd.print(" C "); // Print the unit of the centigrade temperature to the LCD.
    delay(1000);
}

```

2. Using graphical code blocks to program and read the temperature and humidity value of DHT11 sensor on Arduino UNO

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

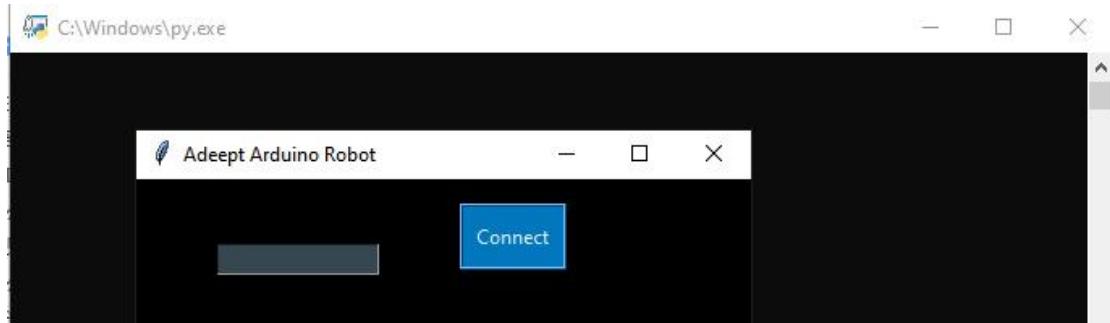
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

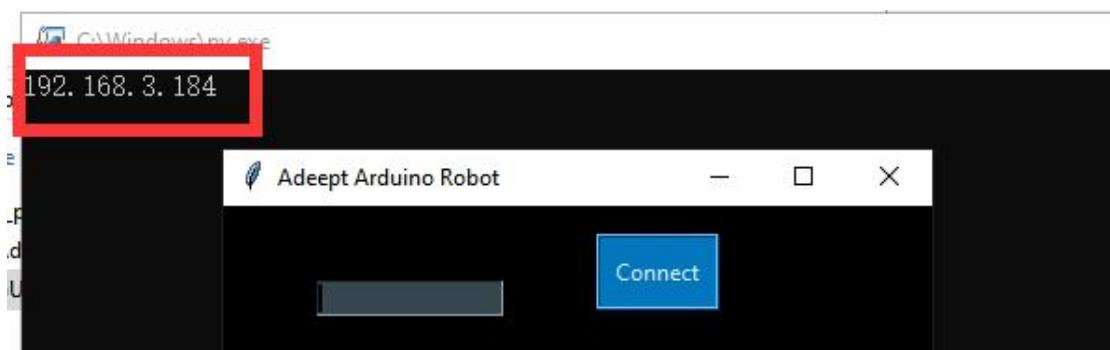
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

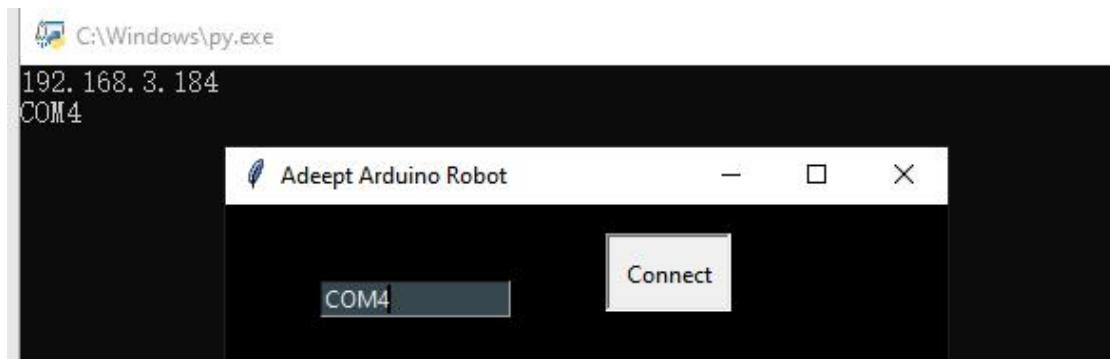
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



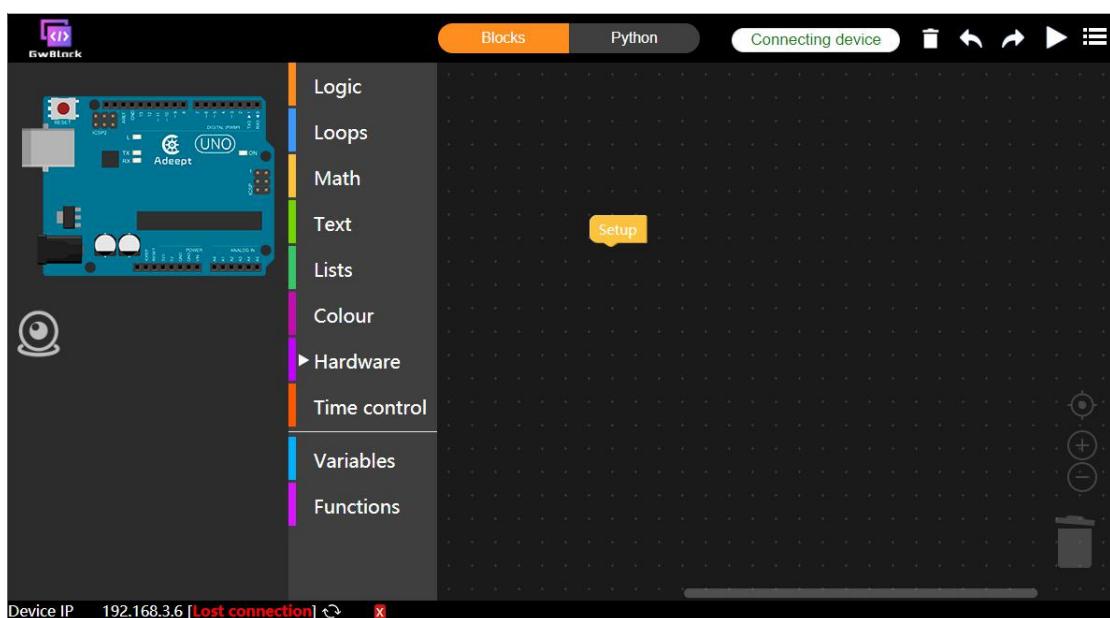
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



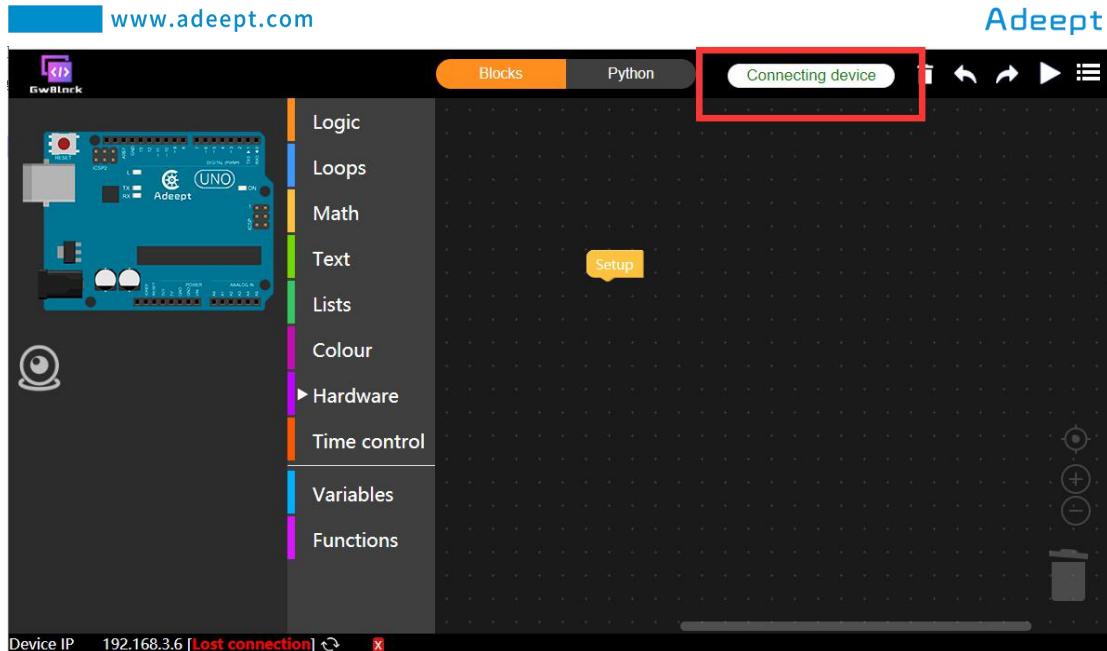
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

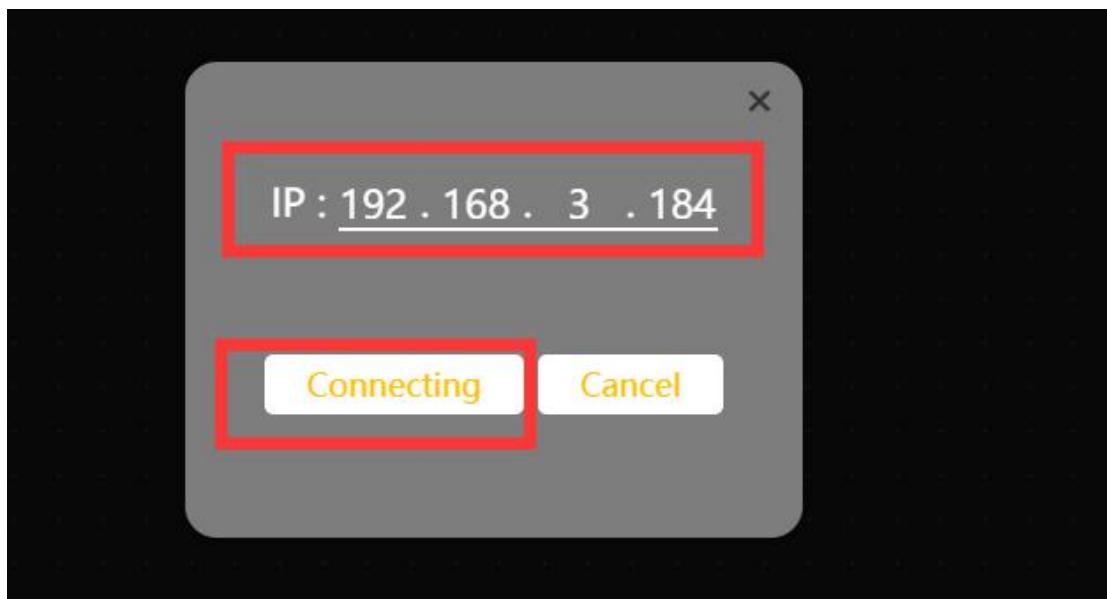
After successfully entering the website, the interface is as follows:



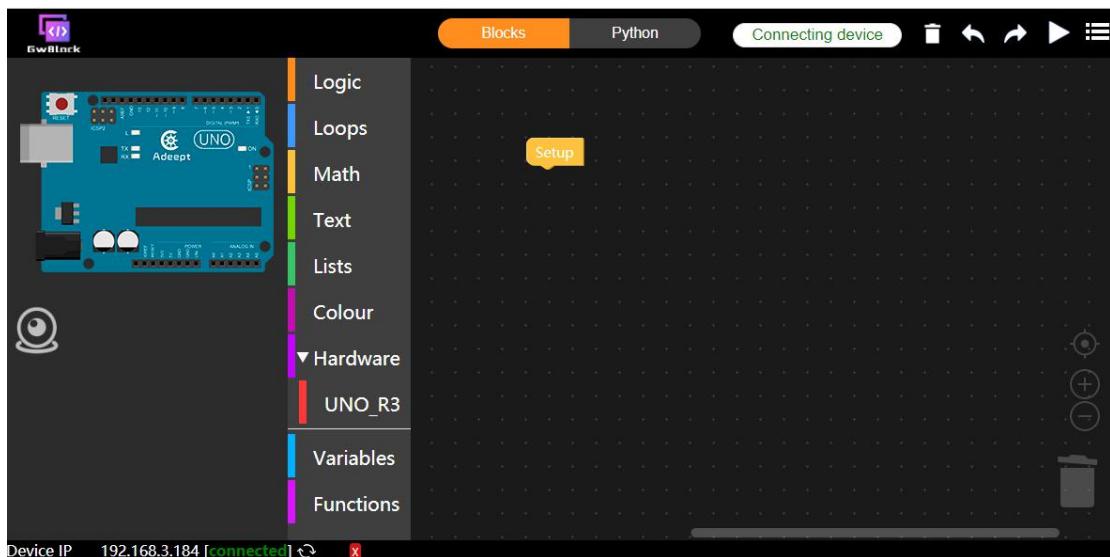
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

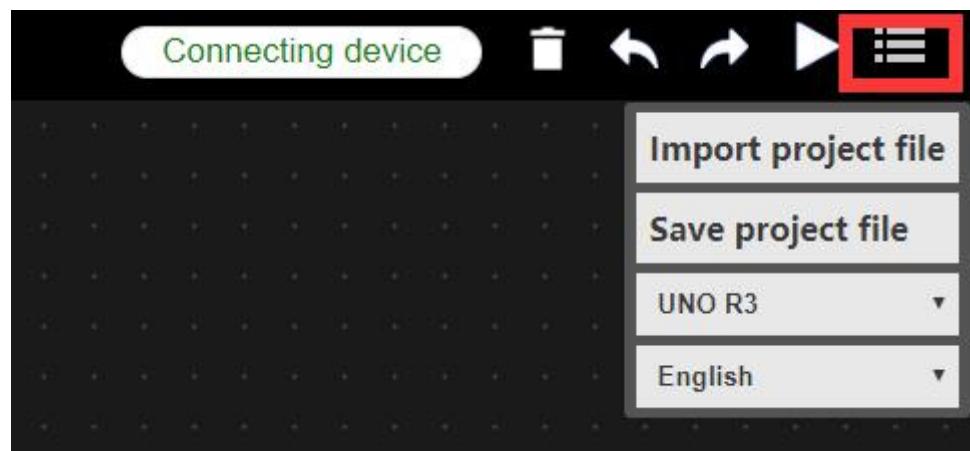


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

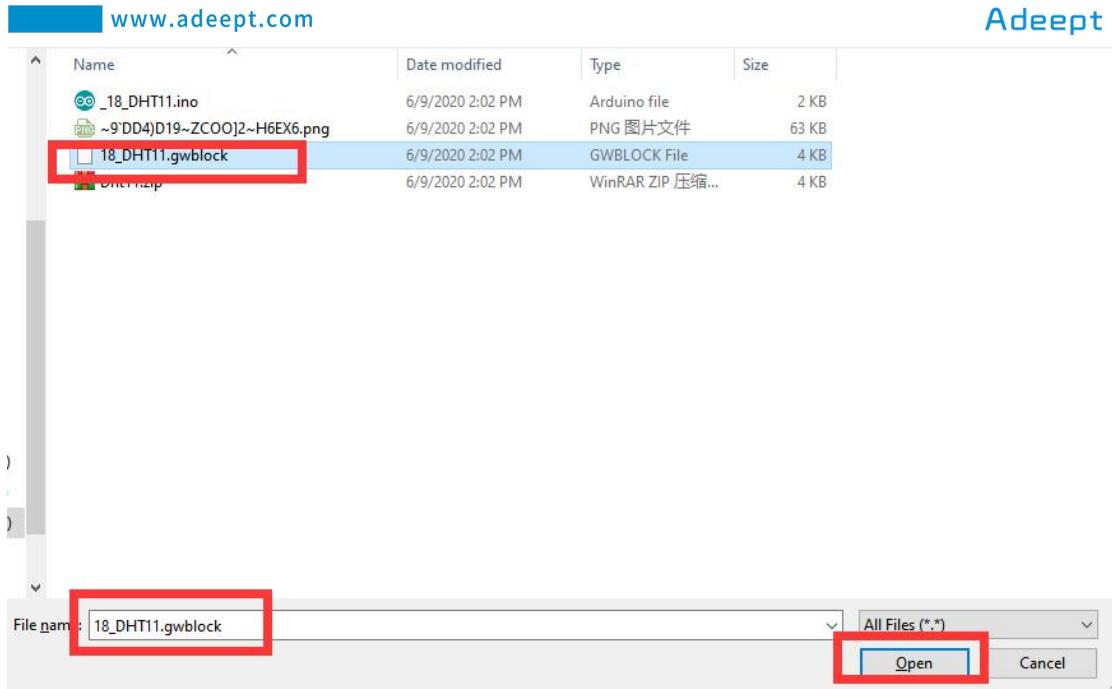
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_18_DHT11

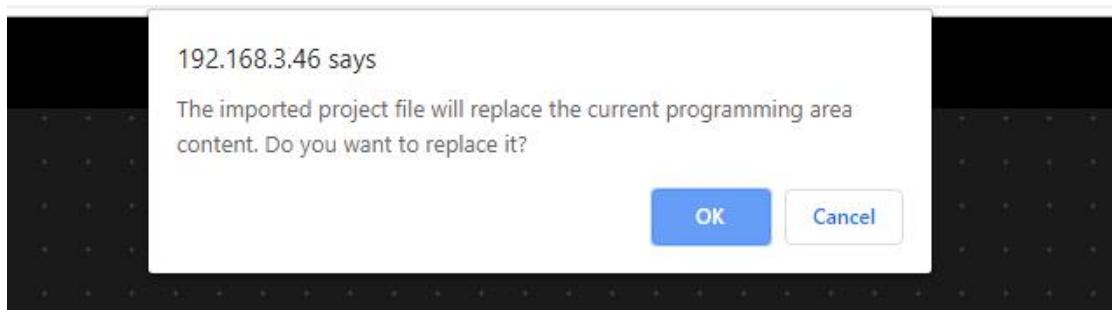
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

5. Select the "18_DHT11.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



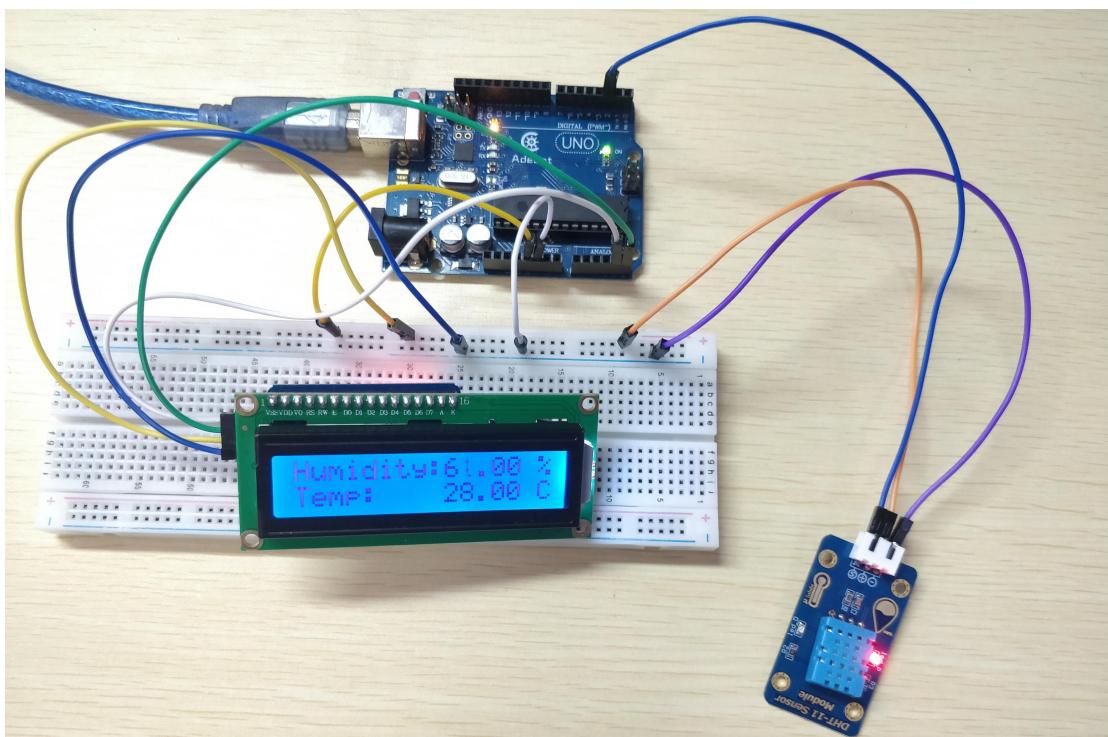
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



8. After successfully running the program, we will observe that the Humidity and Temp values are detected by the DHT11 sensor on the LCD1602 screen, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to read the temperature and humidity value of DHT11 sensor on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

In the GwBlock graphical editor, all code programs are executed from **Setup**. The temperature data of DHT11 is read by the instruction block **set a to Obtain temperature and humidity : PIN 2 Temperature**, and the humidity data is read by the instruction block **set b to Obtain temperature and humidity : PIN 2 Humidity**. Finally, the data of temperature and humidity is displayed on the LCD1602 screen through **LCD display : b** and **LCD display : a** instruction block.



Lesson 19 The Application of the Ultrasonic Distance Sensor

In this lesson, we will learn the application of ultrasonic distance sensor.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
LCD1602	1	
Adeept IIC Module	1	
Male to Female Jumper Wires	Several	
Ultrasonic Distance Sensor	1	

2. The application of the Ultrasonic Distance Sensor

(1) Ultrasonic Distance Sensor

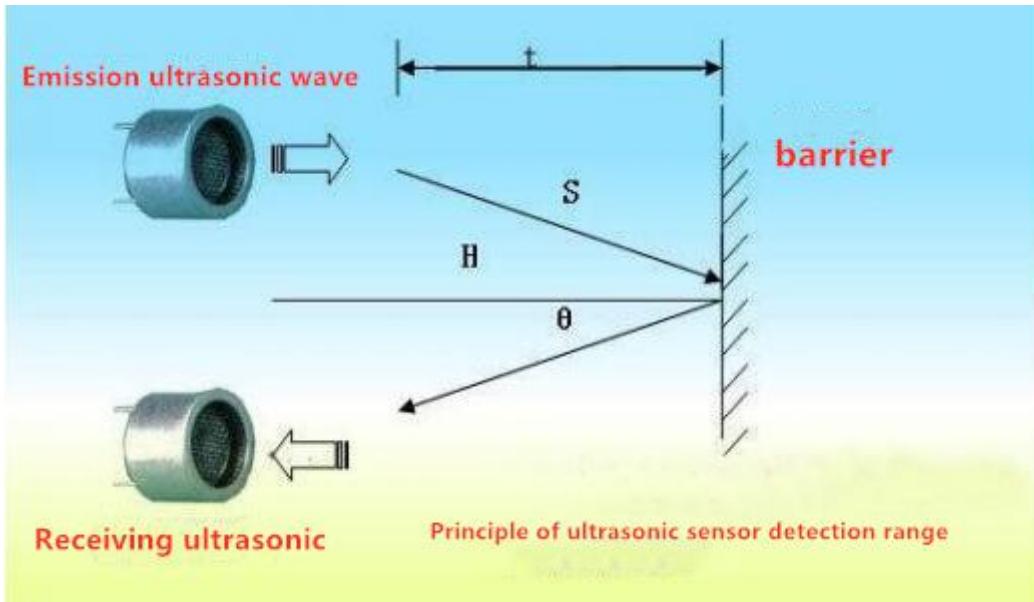
The model of the ultrasonic distance sensor we use is hc-sr04, it is mainly composed of two left and right probes, looking like our human eyes. One probe is

responsible for transmitting sound waves for detection, and the other probe is responsible for receiving sound waves for return. It has 4 pins, which are VCC; Trig (control end - trigger signal input); Echo (receiver - recovery signal output); Gnd(ground).



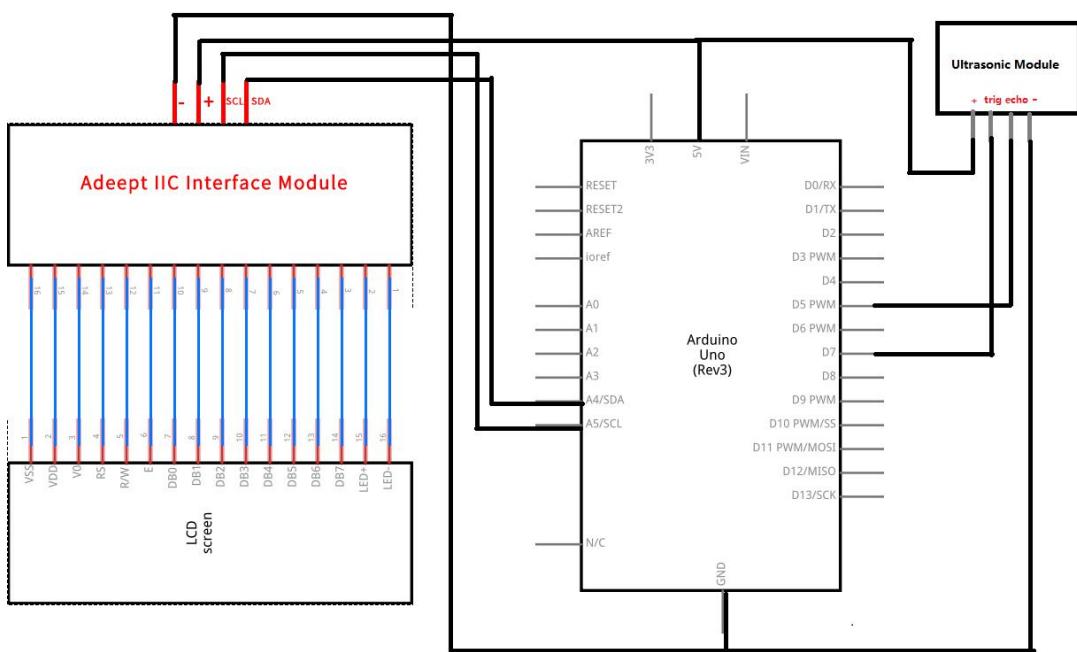
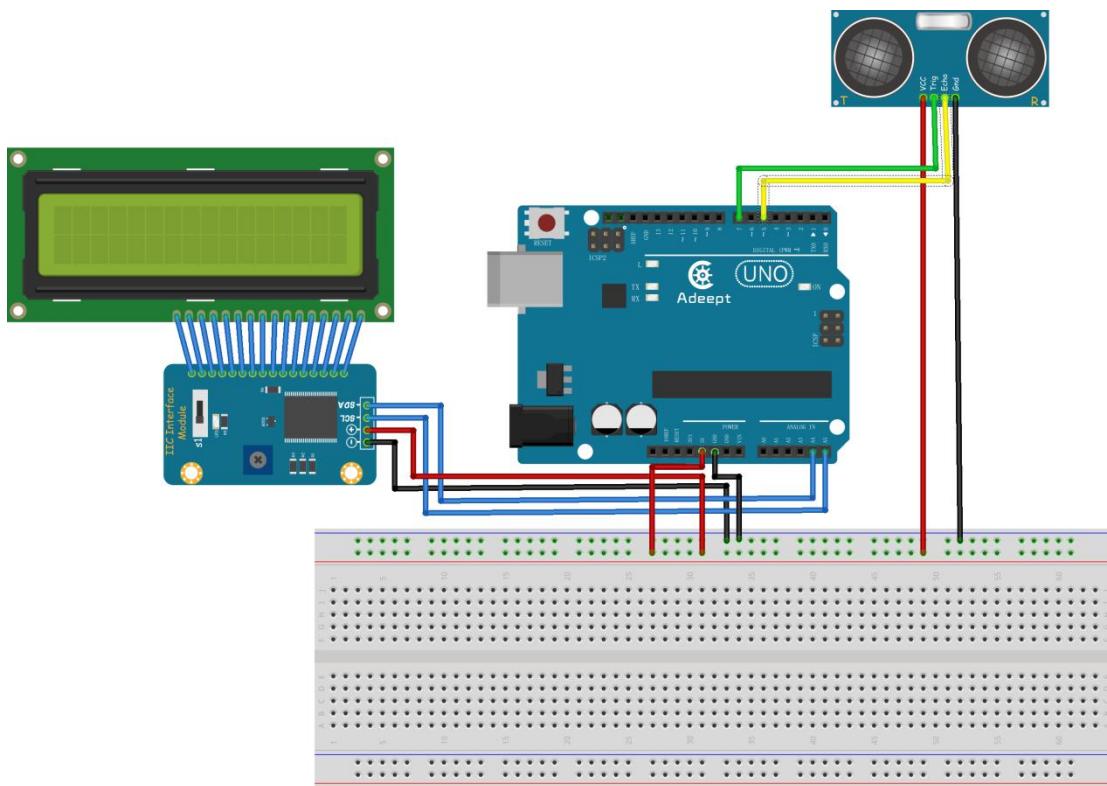
(2) The working principle of the Ultrasonic Distance Sensor

The method of detecting distance of ultrasonic wave is called echo detection method, which ultrasonic emitter emits to a certain direction, in the moment of timer timing starts at the same time, the ultrasonic wave in air, run into obstacles on the way your face (objects) block was reflected immediately, ultrasonic receiver received the ultrasonic reflected back to immediately stop timing. The propagation speed of ultrasonic wave in the air is 340m/s. According to the time t recorded by the timer, the distance s from the launch point to the obstacle surface can be calculated, that is, $s=340t/2$. Under this principle of ultrasound, ultrasonic ranging module is widely used in practical applications, such as car reversing radar, uav, and intelligent car.



3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:



4. The application of the Ultrasonic Distance Sensor

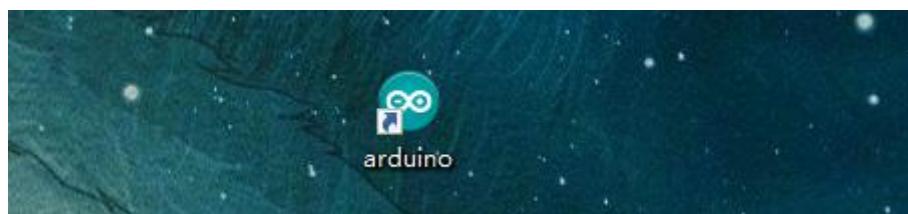
We provide two different methods to read the distance of the obstacle detected by

the ultrasonic distance sensor. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

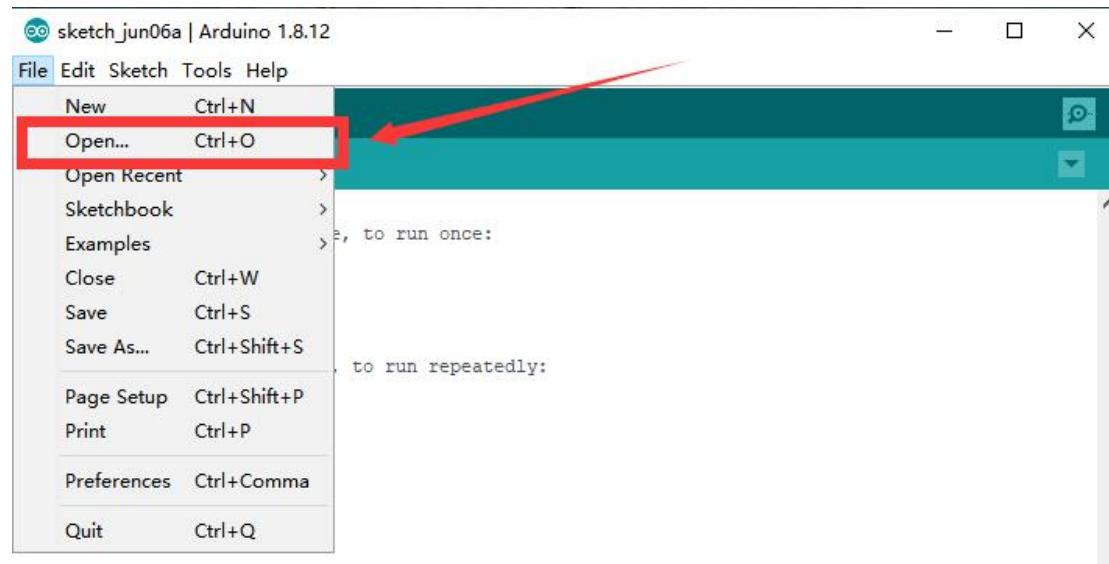
1. Using C language to program to read the value of ultrasonic distance sensor on Arduino UNO

(1) Compile and run the code program of this course

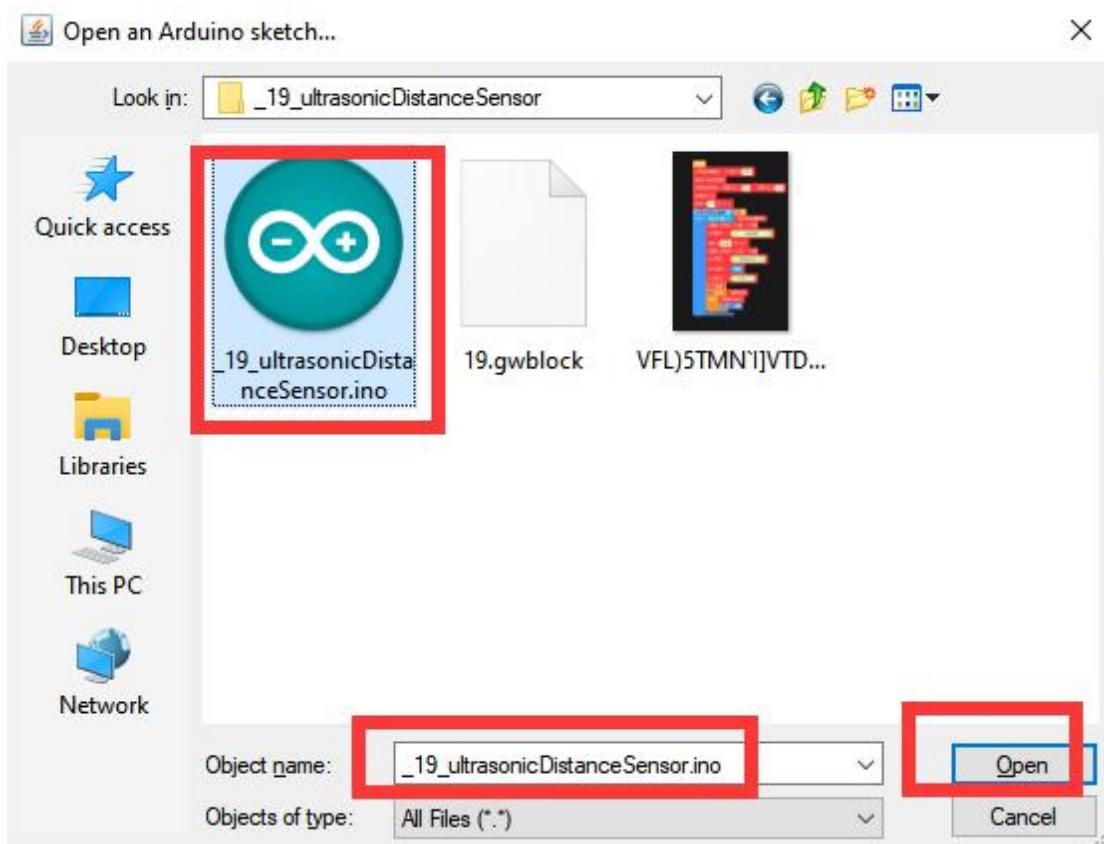
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the directory: Code\19_ultrasonicDistanceSensor .Select_19_ultrasonicDistanceSensor.ino. This file is the code program we need in this course. Then click Open.



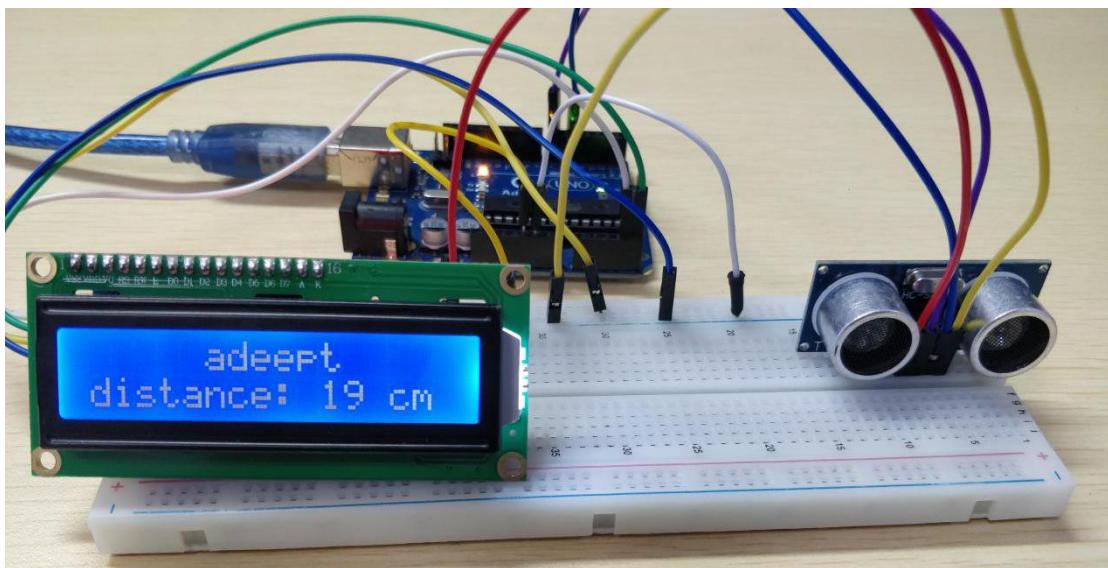
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After successfully running the program, we will observe the obstacle distance detected by the ultrasonic sensor displayed on the LCD1602 screen, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to read the distance of the obstacle detected by the ultrasonic distance sensor on Arduino UNO. We will introduce how our core code can be achieved:

- 1.In the setup() function, set pingPin to INPUT mode by pinMode(pingPin, INPUT), and set trigPin to OUTPUT mode by pinMode(trigPin, OUTPUT).

```
void setup()
{
    pinMode(pingPin, INPUT); //Set the connection pin output mode Echo pin
    pinMode(trigPin, OUTPUT); //Set the connection pin output mode trig pin
    lcd.init();
    lcd.backlight();
    delay(1000);           //delay 1000ms
}
```

- 2.In the loop() function, the text information is displayed on the LCD1602 screen through the lcd.print() function, and we set the display content to be adept distance.

```

void loop()
{
    int cm = ping(pingPin) ;
    lcd.setCursor(0, 0);           // set the cursor to column 0, line 0
    lcd.print("      adeept      "); // Print a message of "Temp: "to the LCD.

    lcd.setCursor(0, 1);           // set the cursor to column 0, line 0
    lcd.print("distance: ");       // Print a message of "Temp: "to the LCD.
    lcd.print(cm);                // Print a centigrade temperature to the LCD.
    lcd.print(" cm     ");         // Print the unit of the centigrade temperature to the LCD.
    delay(500);
}

```

3.The detection distance of the ultrasonic ranging sensor can be calculated through the ping() function.

```

int ping(int pingPin)
{
    // establish variables for duration of the ping,
    // and the distance result in inches and centimeters:
    long duration, cm;
    // The PING))) is triggered by a HIGH pulse of 2 or more microseconds.
    // Give a short LOW pulse beforehand to ensure a clean HIGH pulse:
    pinMode(trigPin, OUTPUT);
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(5);
    digitalWrite(trigPin, LOW);

    pinMode(pingPin, INPUT);
    duration = pulseIn(pingPin, HIGH);

    // convert the time into a distance
    cm = microsecondsToCentimeters(duration);
    return cm ;
}

long microsecondsToCentimeters(long microseconds)
{
    // The speed of sound is 340 m/s or 29 microseconds per centimeter.
    // The ping travels out and back, so to find the distance of the
    // object we take half of the distance travelled.
    return microseconds / 29 / 2;
}

```

2.Using graphical code blocks to program to read the value of ultrasonic distance sensor on Arduino UNO

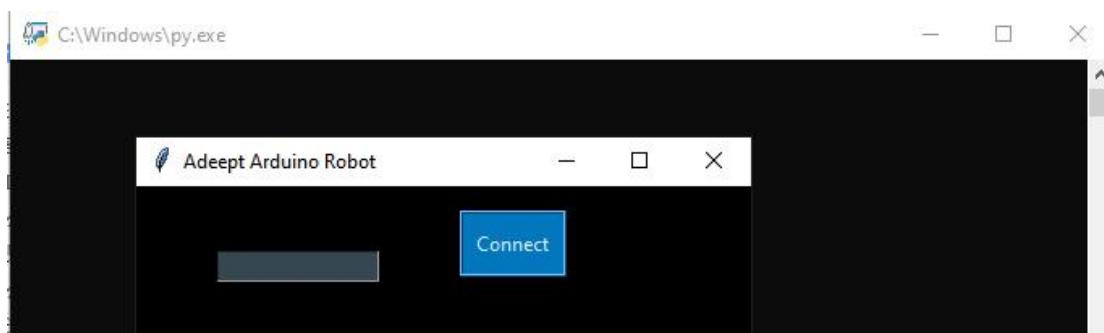
(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

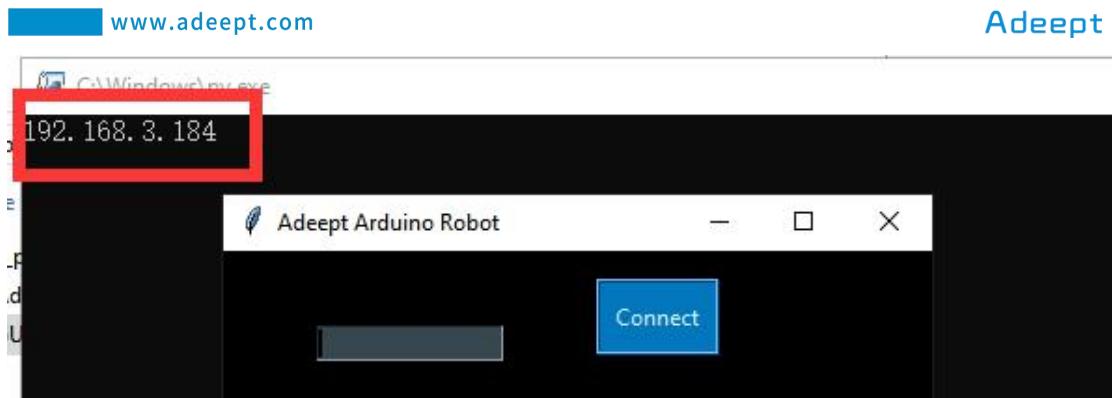
1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:



2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



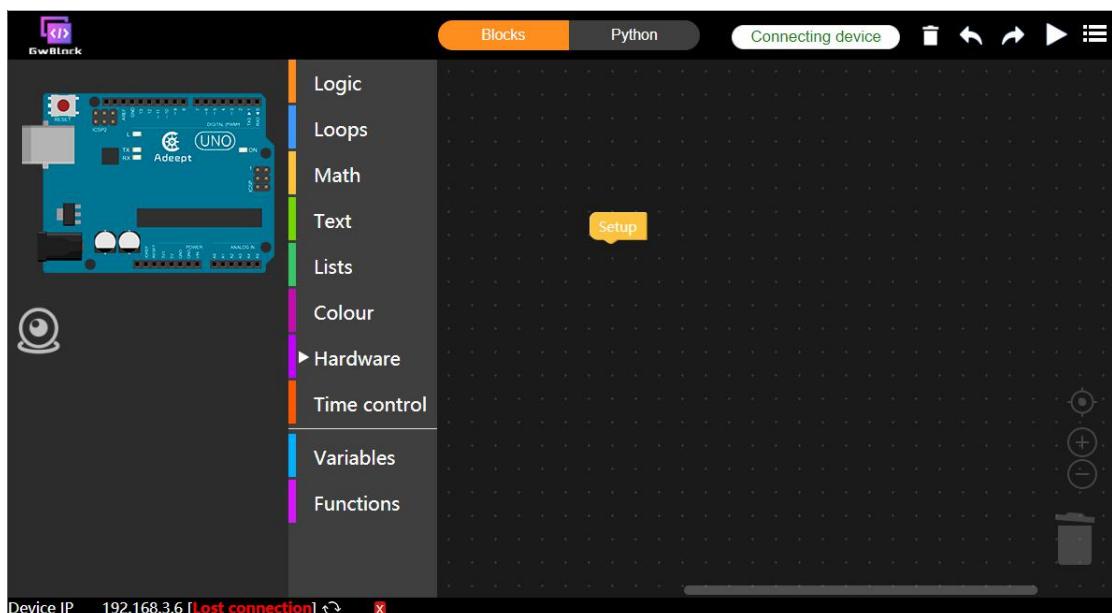
4. Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



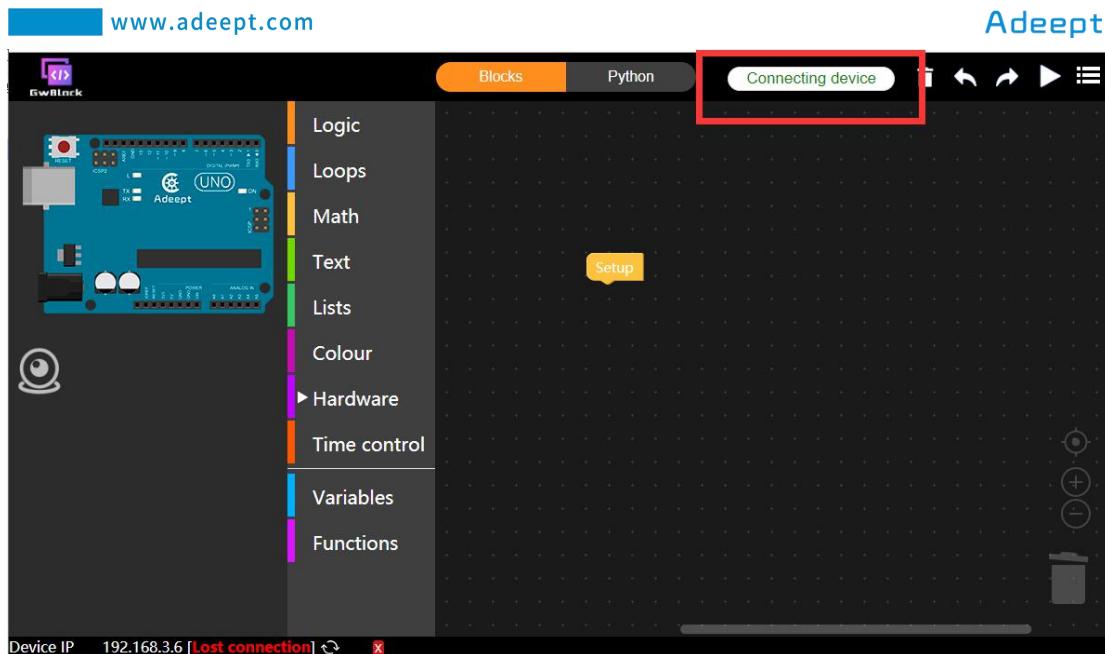
5. Enter the URL of the GwBlock graphical editor in the browser:

[http://www.adeept.com/gwblock/?hd_mo=uno_r3.](http://www.adeept.com/gwblock/?hd_mo=uno_r3)

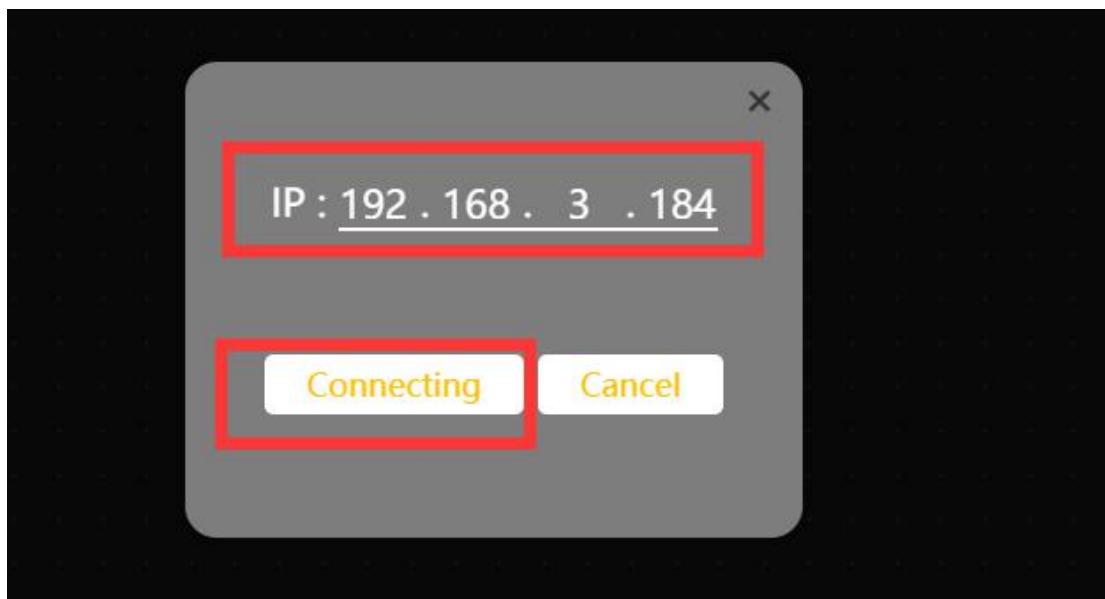
After successfully entering the website, the interface is as follows:



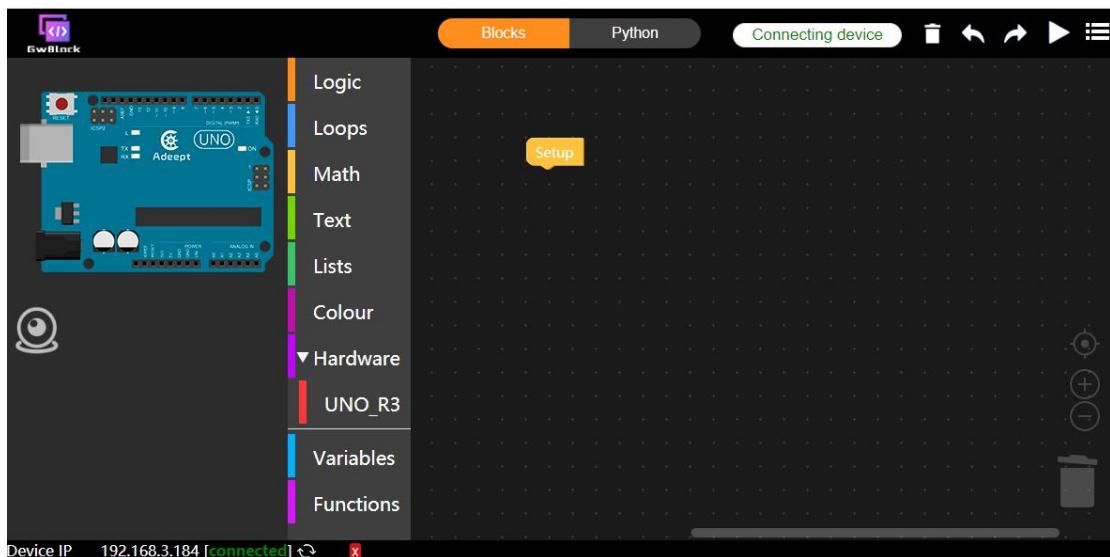
6. Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

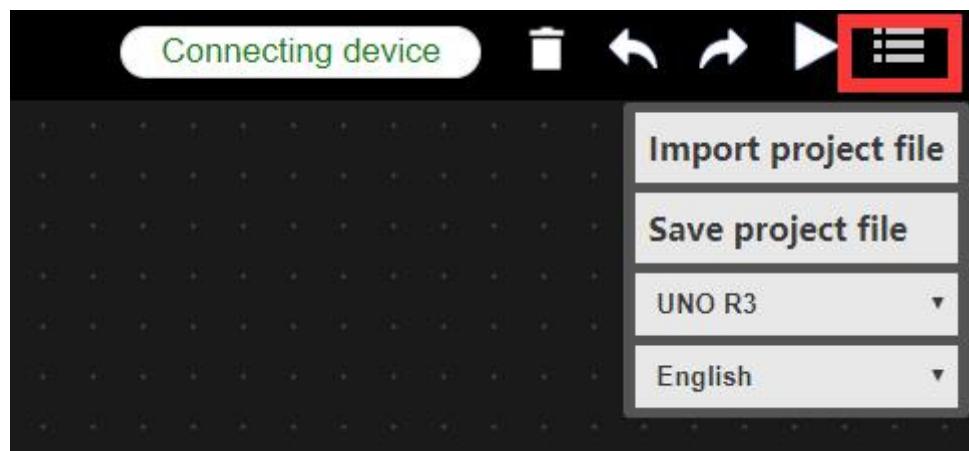


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

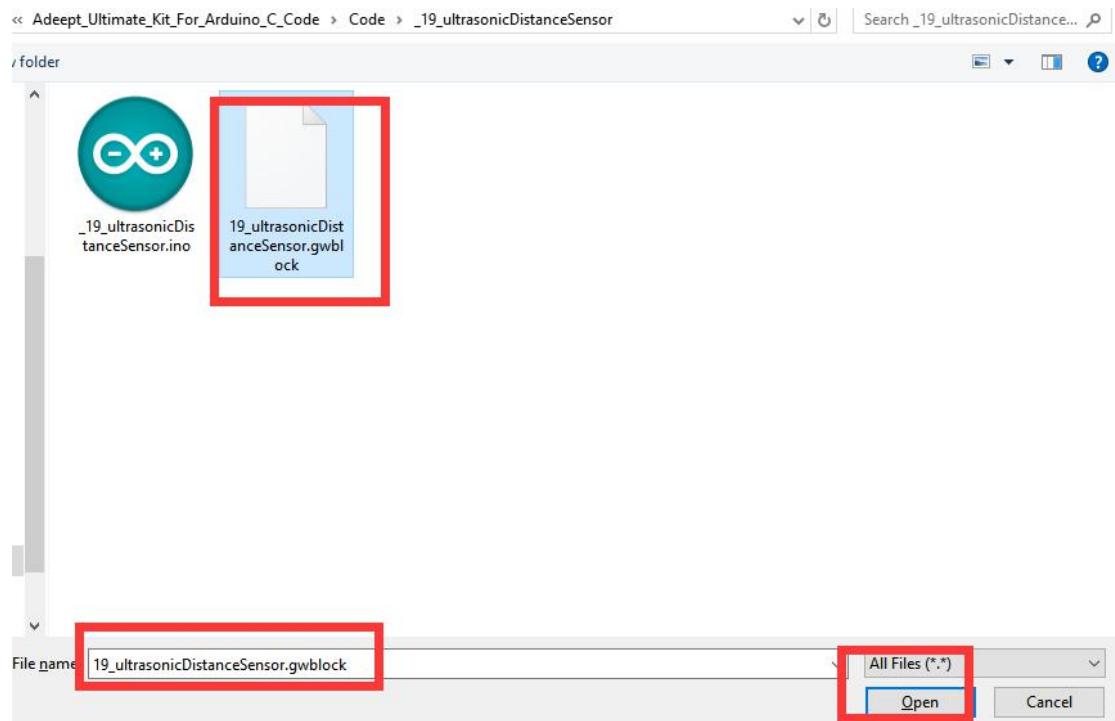
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_19_ultrasonicDistanceSensor

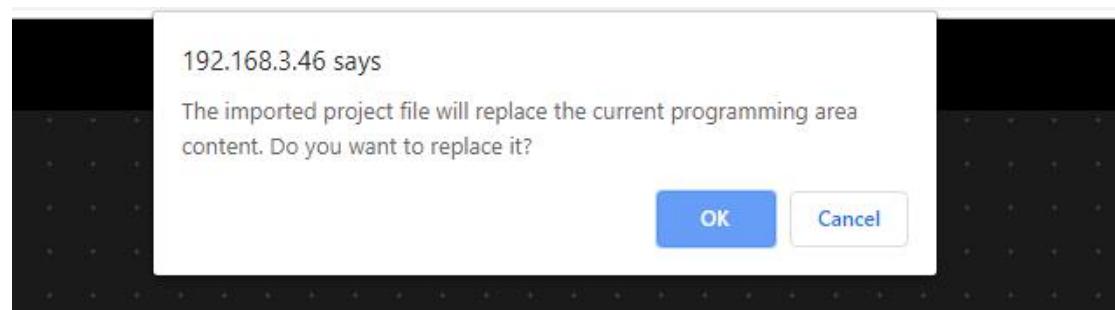
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

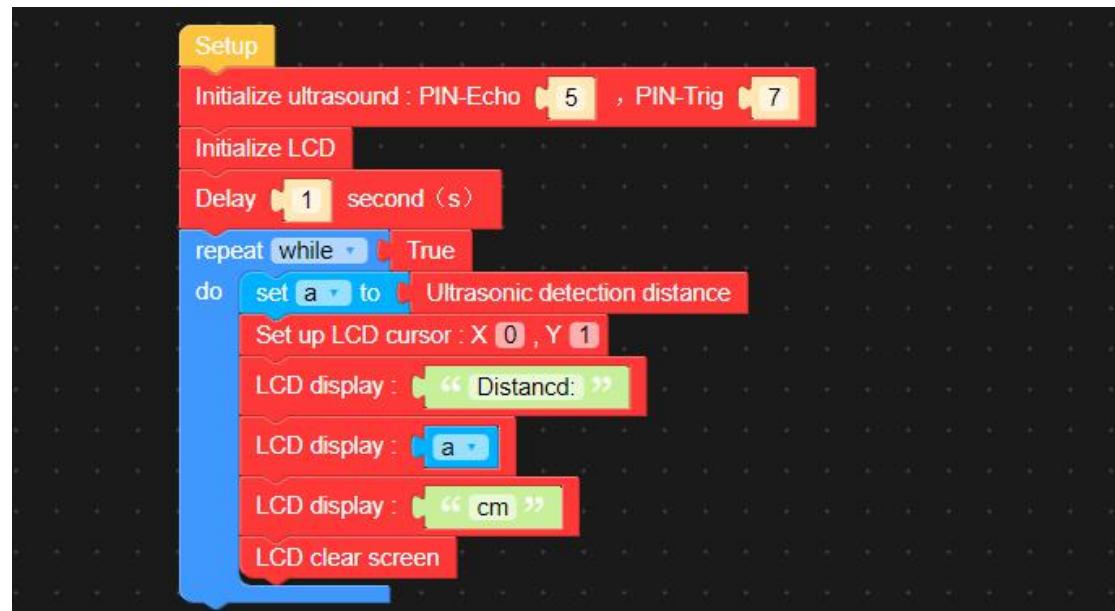
5. Select the "19_ultrasonicDistanceSensor.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



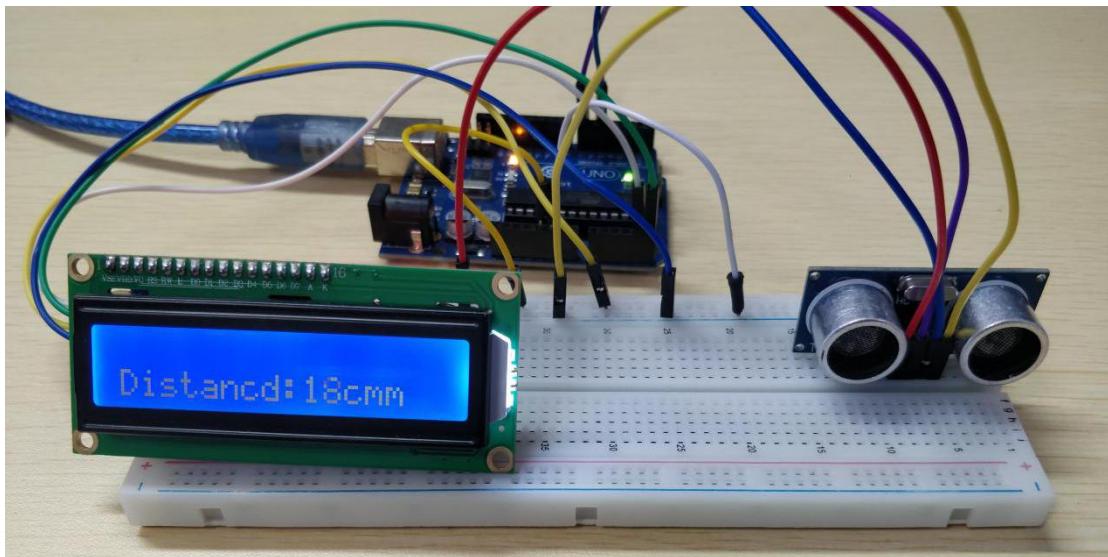
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



8.Click the button  on the upper right corner. After successfully running the program, we will observe the obstacle distance detected by the ultrasonic sensor displayed on the LCD1602 screen, indicating that our experimental test is successful. The physical connection diagram of the experiment is as below:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to read the distance of the obstacle detected by the ultrasonic distance sensor on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

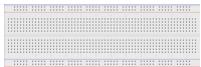
In the GwBlock graphical editor, all code programs are executed from **Setup**. Obtain the data of the ultrasonic ranging sensor through the instruction block **set a to Ultrasonic detection distance**, and display the acquired data on the LCD1602 screen through **LCD display : a**.



Lesson 20 Application of MPU6050

In this lesson, we will learn the application of MPU6050.

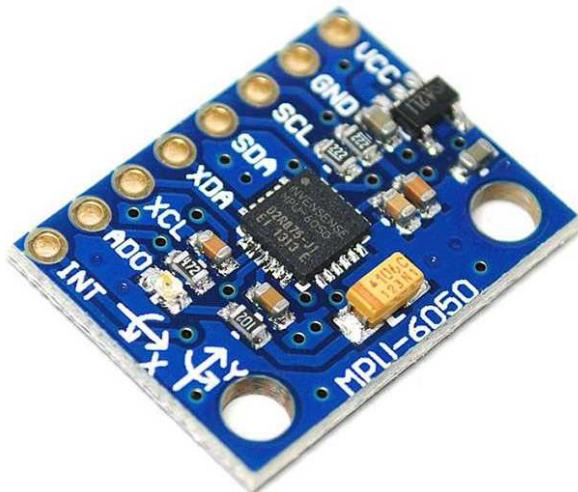
1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
IR Receiver HX1838	1	
Male to Female Jumper Wires	4	

2. The introduction of MPU6050

(1) MPU6050

The MPU6050 is an integrated 6-axis motion processing component. Compared with the multi-component solution, it eliminates the problem of the difference between the axes when combining the gyroscope and the accelerator, and reduces a lot of packaging space. The MPU6050 integrates a gyroscope, acceleration sensor, and temperature sensor. It can accept the input of an external 3-axis compass through the I2C bus and provide 9-axis output. It can be used to obtain the current acceleration component and angular velocity component of the device.



(2) Sensing range of MPU6050

The MPU-6050 uses three 16-bit ADCs for the gyroscope and accelerometer respectively. It converts the measured analog quantity into a digital quantity which can be outputted. In order to accurately track fast and slow movements, the measurement range of the sensor is user-controllable. The gyro's measurement range is $\pm 250, \pm 500, \pm 1000, \pm 2000^{\circ}/\text{SEC}$ (DPS). The accelerometer's measurement range is $\pm 2, \pm 4, \pm 8$, and $\pm 16\text{g}$. Mpu-6050 can operate at different voltages. VDD power supply voltage of $2.5\text{v} \pm 5\%$, $3.0\text{v} \pm 5\%$ or $3.3\text{v} \pm 5\%$. VDDIO power supply $1.8\text{v} \pm 5\%$. for logical interface.

2. Using graphical code blocks to program to read the value of MPU6050 on Arduino UNO

(1) Connecting to GwBlock graphical editor

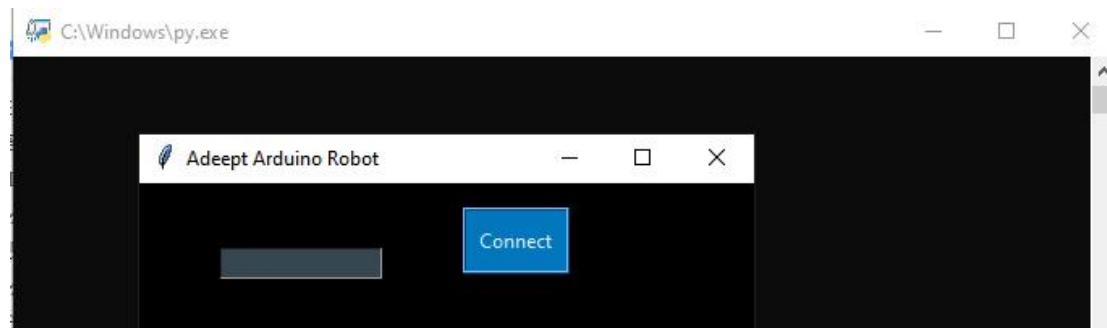
In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). [Note]: We need to modify #define MPU6050 0 in the opened block_py.ino file to #define MPU6050 1. Then click  and upload

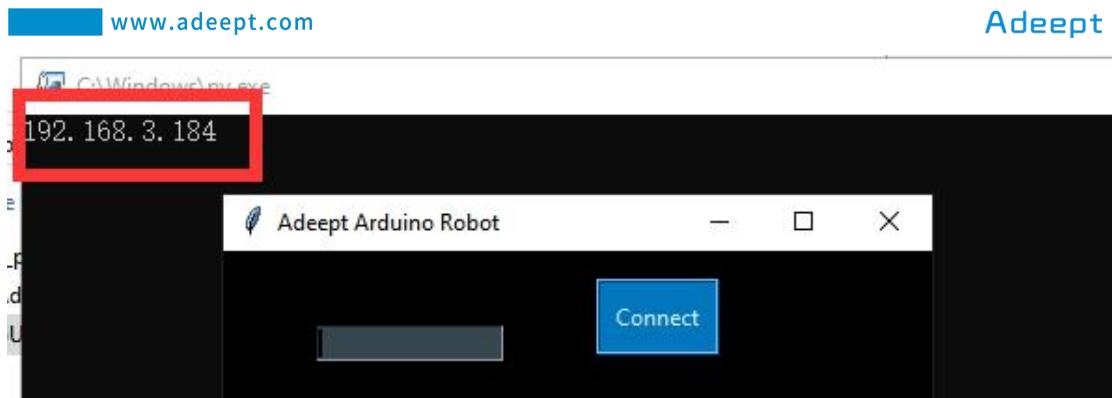
the program to the Arduino UNO. After successful Upload, it will show as below:



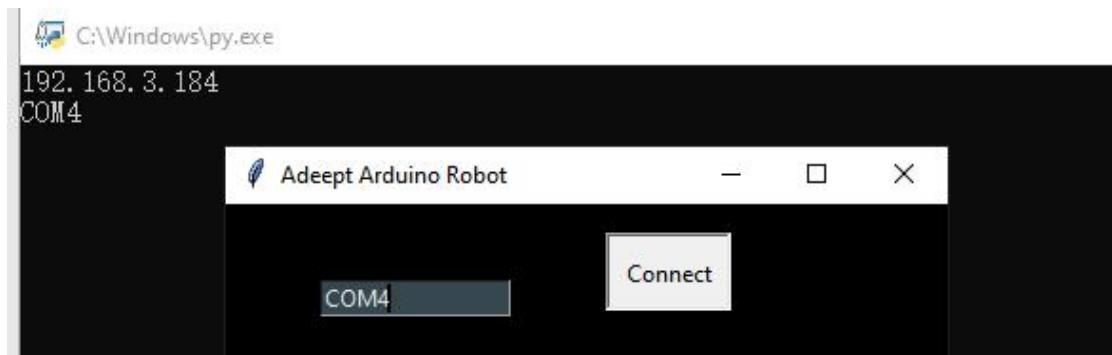
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3.Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



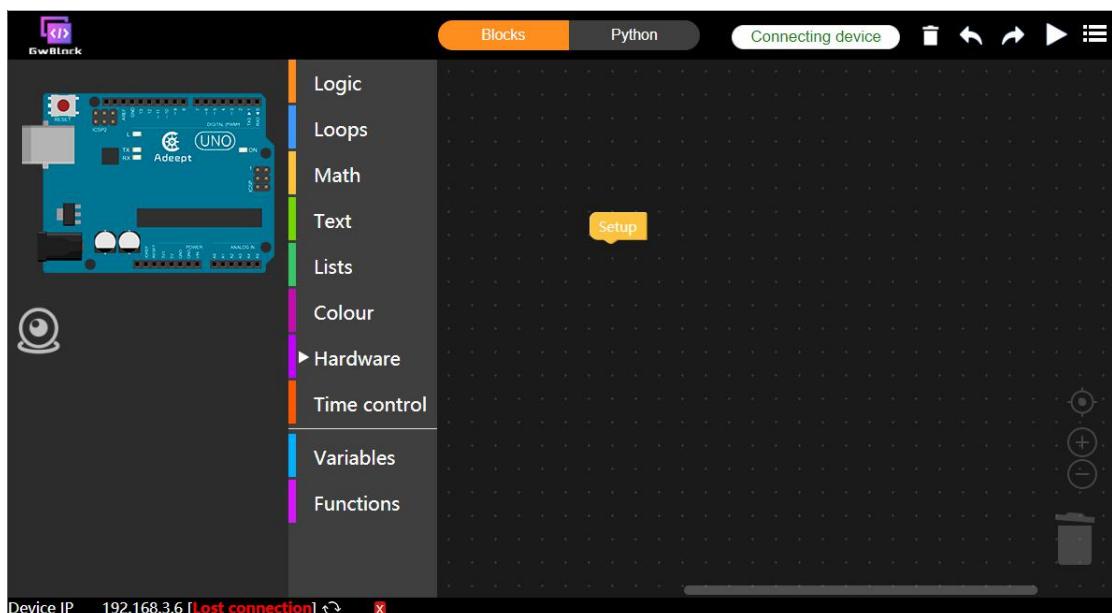
4. Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



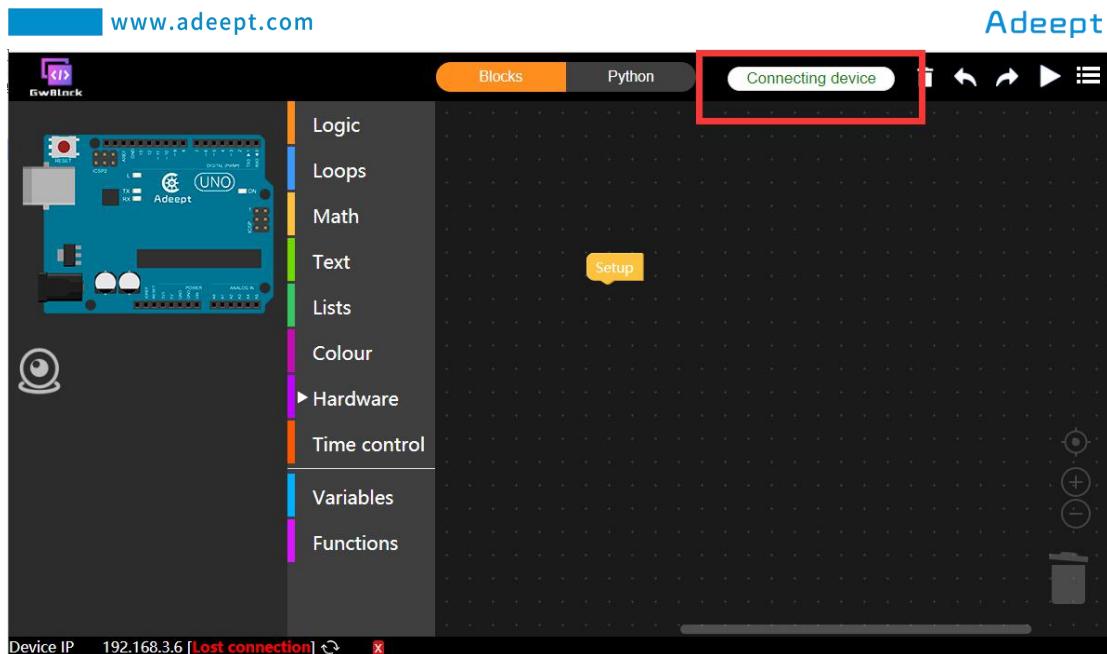
5. Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

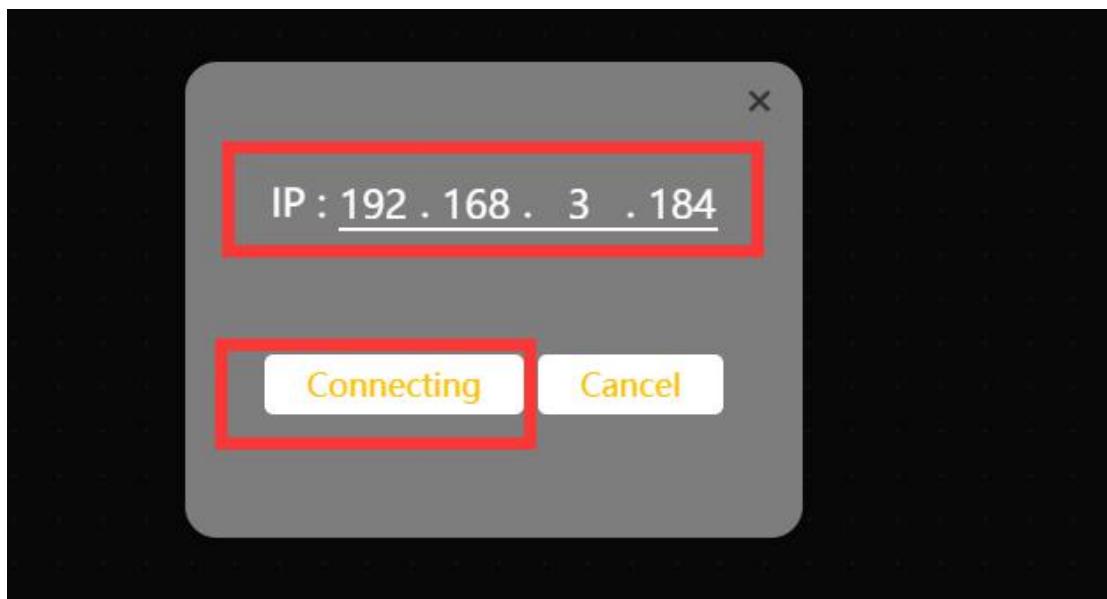
After successfully entering the website, the interface is as follows:



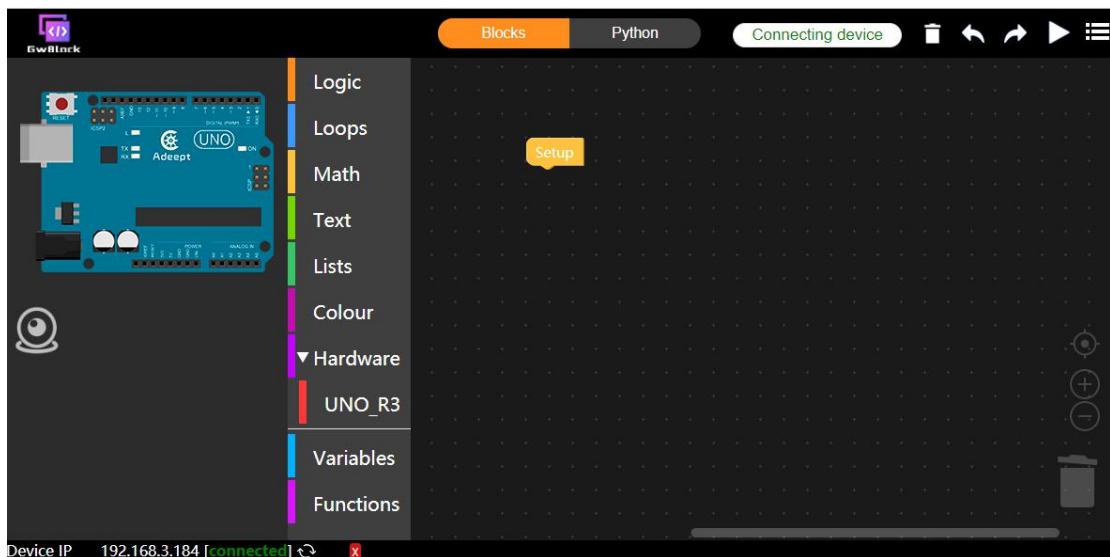
6. Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

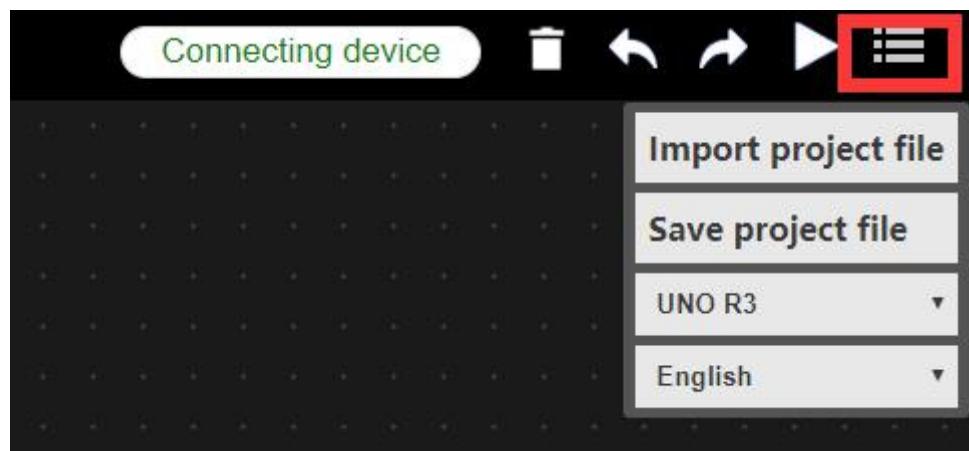


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

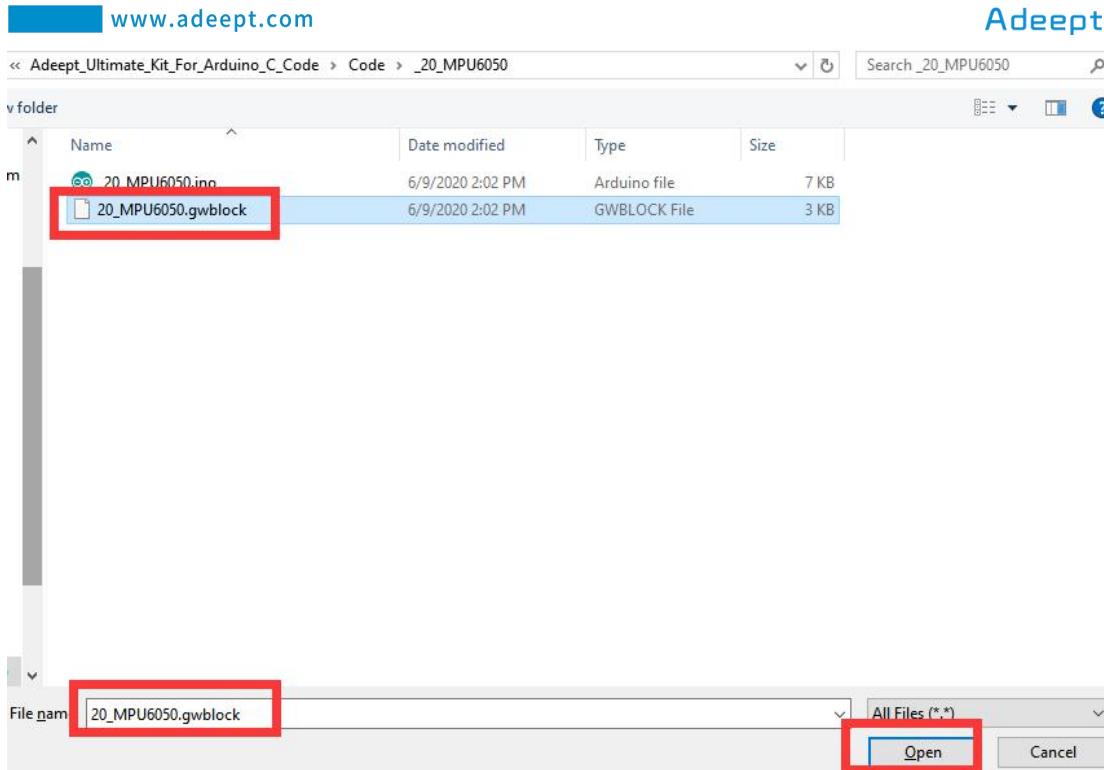
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_20_MP6050

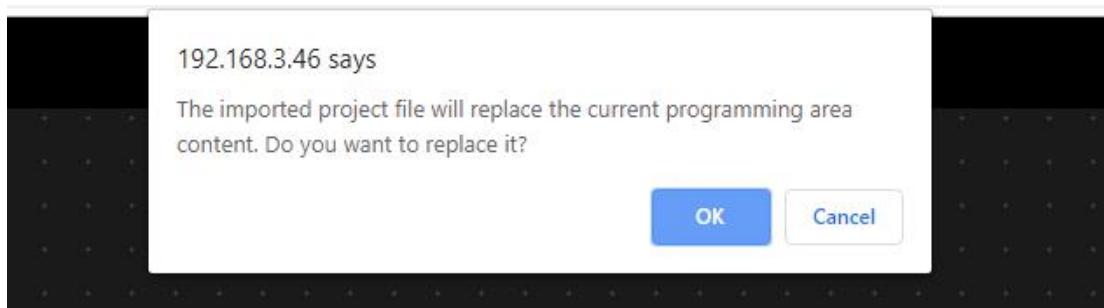
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

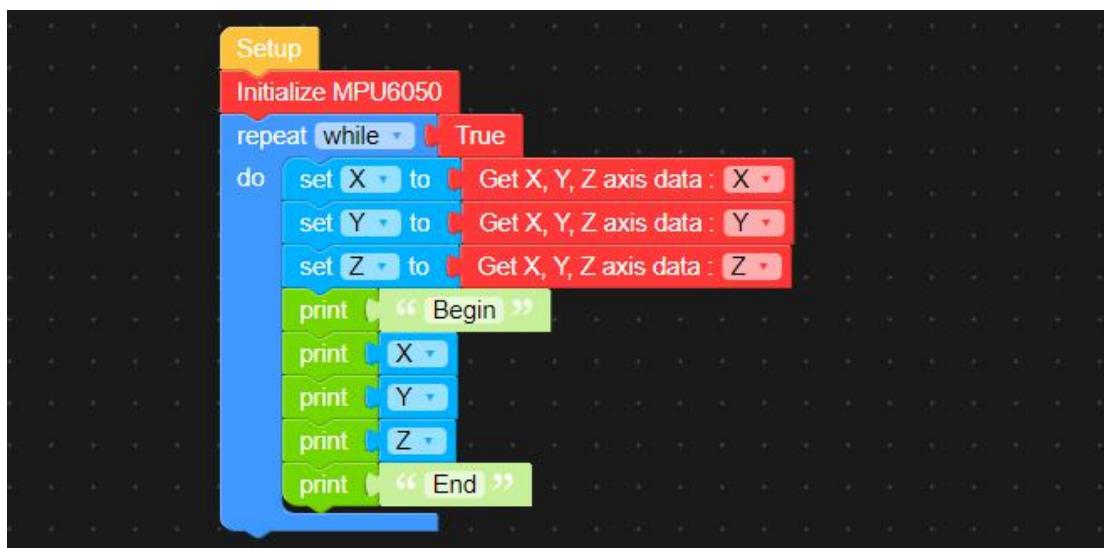
5. Select the "20_MP6050.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



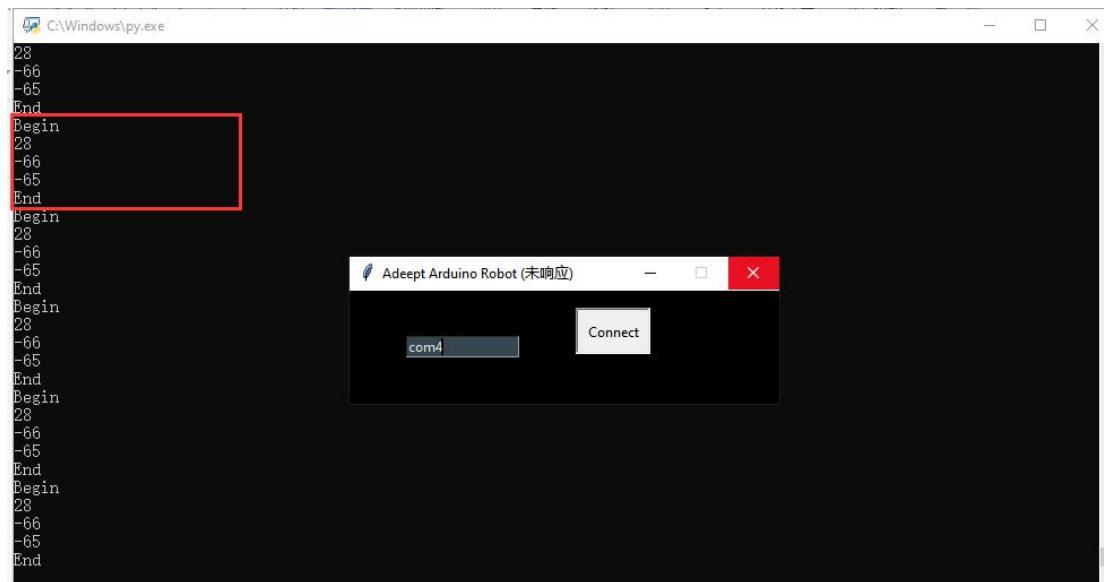
6.Click OK.It will show as below:



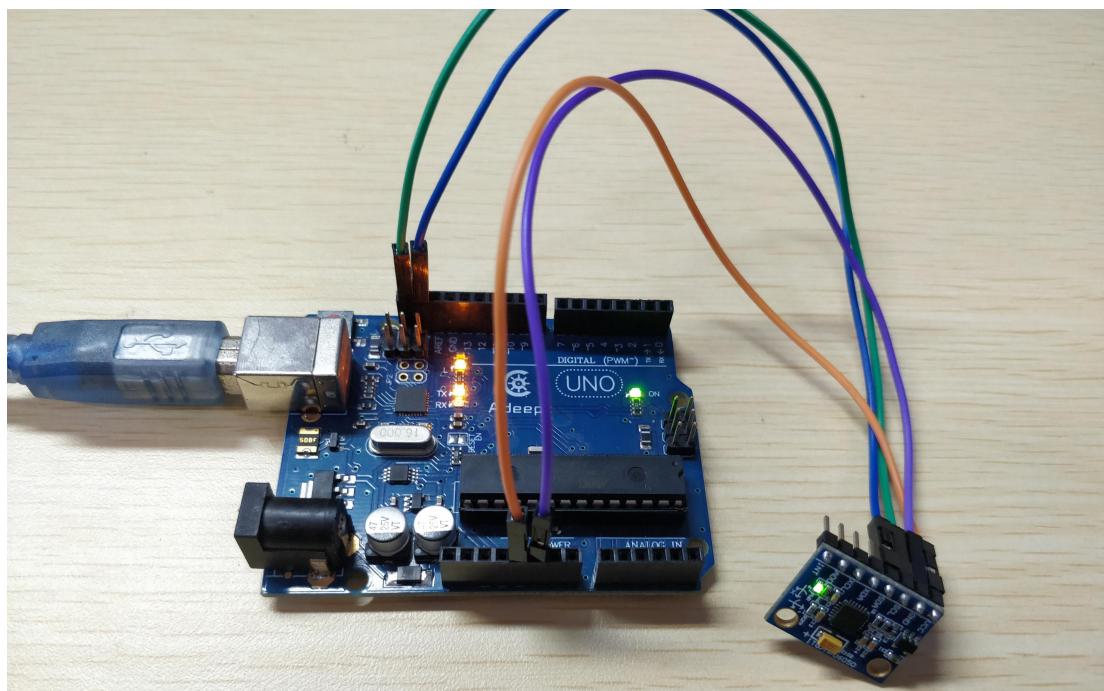
7.It will show as below after successfully opening:



5. Click the button  on the upper right corner. After successfully running the program, we open the GUI info v1.0.py window. It will display the X, Y, Z axis values of the detected MPU6050, indicating that our experimental test is successful.



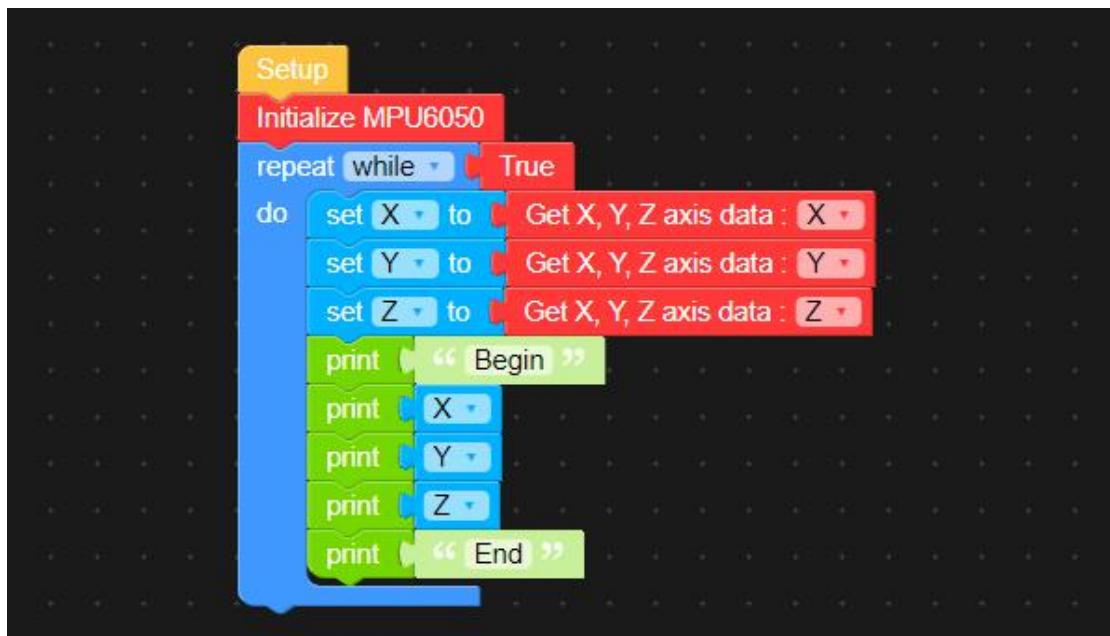
9. The physical connection of the experiment is as below:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to read the value of MPU6050 on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

In the GwBlock graphical editor, all code programs are executed from **Setup**. Initialize the MPU6050 sensor through **Initialize MPU6050**. Obtain the X, Y, and Z axis data of the MPU6050 through the instruction block **Get X, Y, Z axis data**. Finally print out the X, Y, and Z data through the instruction **print**.



(3) Features of MPU6050

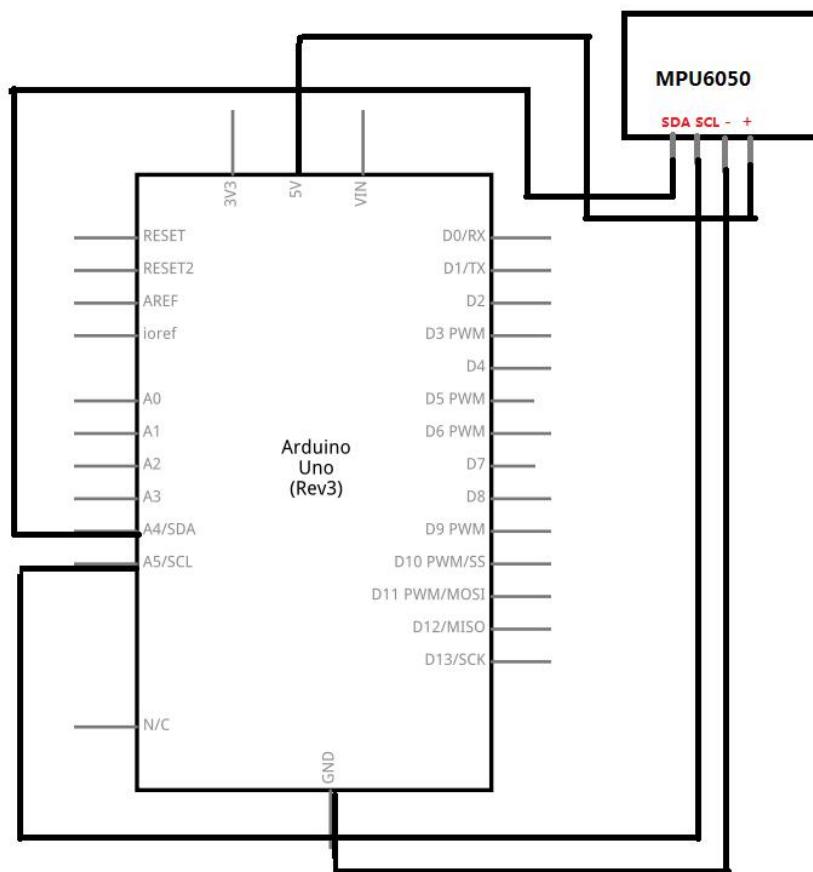
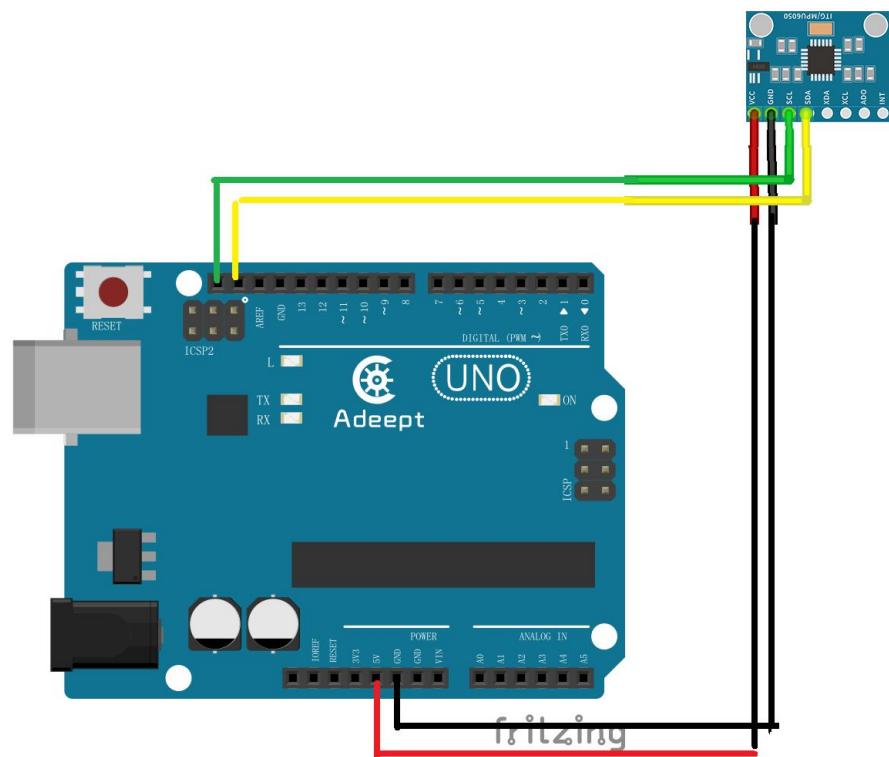
1. Digital output of 6-axis or 9-axis rotation matrix, quaternion (quaternion), Euler angle format (EulerAngleforma) fusion calculation data.
2. 3-axis angular velocity sensor (gyroscope) with 131 LSBs/ $^{\circ}$ /sec/sensitivity and full-scale sensing range of ± 250 , ± 500 , ± 1000 and ± 2000 $^{\circ}$ /sec.
3. Programmable control, and the range of program control is $\pm 2g$, $\pm 4g$, $\pm 8g$ and $\pm 16g$ 3-axis accelerator.
4. The sensitivity between the axis of the accelerator and the gyroscope can be removed to reduce the influence of the setting and the drift of the sensor.
5. Digital Motion Processing (DMP: Digital Motion Processing) engine can reduce the load of complex fusion calculation data, sensor synchronization, posture sensing,

etc. The motion processing database supports the built-in operation time deviation and magnetic sensor calibration calculation technology of Android, Linux and Windows, eliminating the need for customers to perform additional calibration.

6. Temperature sensor with digital output.
7. Syncpin with digital input supports video electronic video stabilization technology and GPS.
8. Programmable interrupt (interrupt) supports gesture recognition, panning, zooming in and out of the screen, scrolling, rapid descent interruption, high-G interruption, zero motion sensing, touch sensing, and shaking sensing functions.
9. VDD power supply voltage of $2.5V \pm 5\%$, $3.0V \pm 5\%$, $3.3V \pm 5\%$; VDDIO is $1.8V \pm 5\%$.
10. Gyroscope operating current: 5mA; gyroscope standby current: 5uA; accelerator operating current: 500uA; accelerator power saving mode current: 40uA@10Hz.
11. I2C in fast mode up to 400kHz, or SPI serial host interface up to 20MHz.
12. The built-in oscillator only has a frequency change of $\pm 1\%$ within the operating temperature range. Optional external clock input 32.768kHz or 19.2MHz.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:



4. Application of MPU6050

We provide two different methods to read the value of MPU6050. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

1. Using C language to program to read the value of MPU6050 on Arduino UNO

(1) Compile and run the code program of this course

1. Open the Arduino IDE software, as shown below:

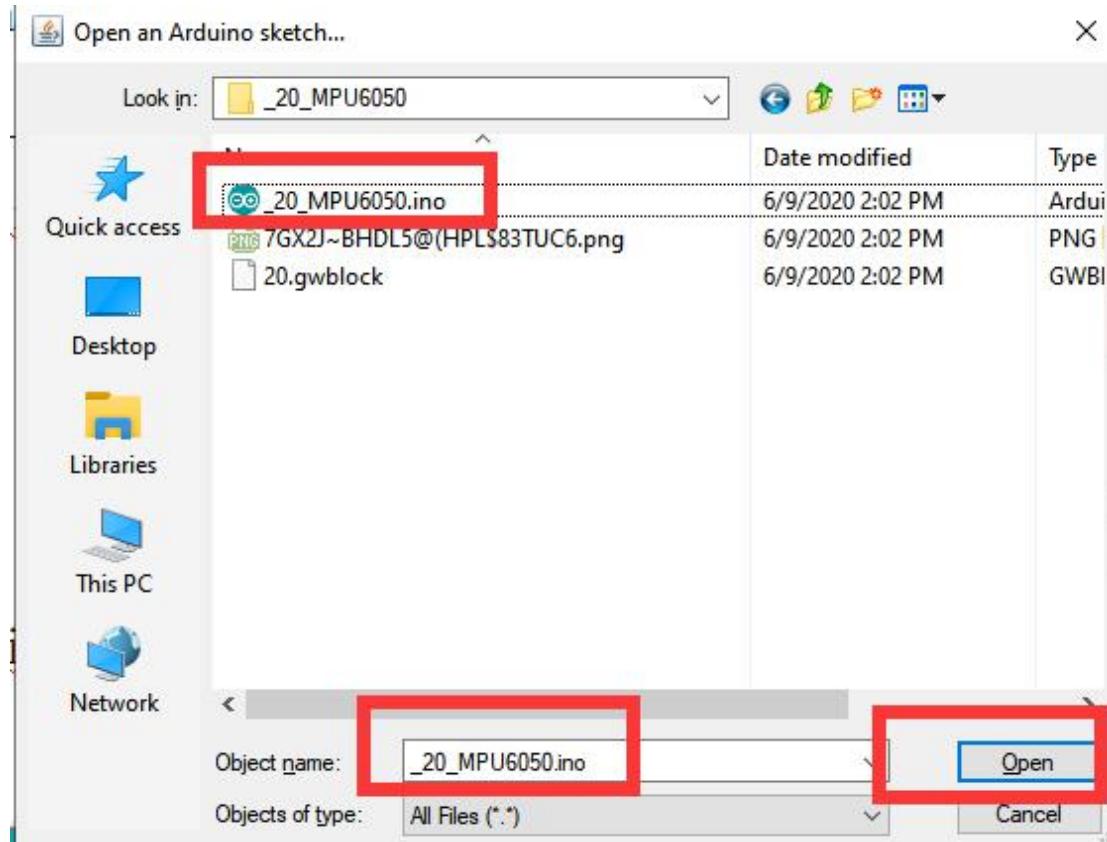


2. Click Open in the File drop-down menu:

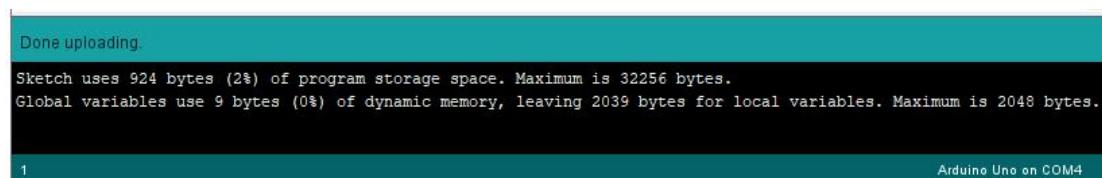


3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the

Code\20_MPU6050 directory. Select_20_MPU6050.ino. This file is the code program we need in this course. Then click Open.



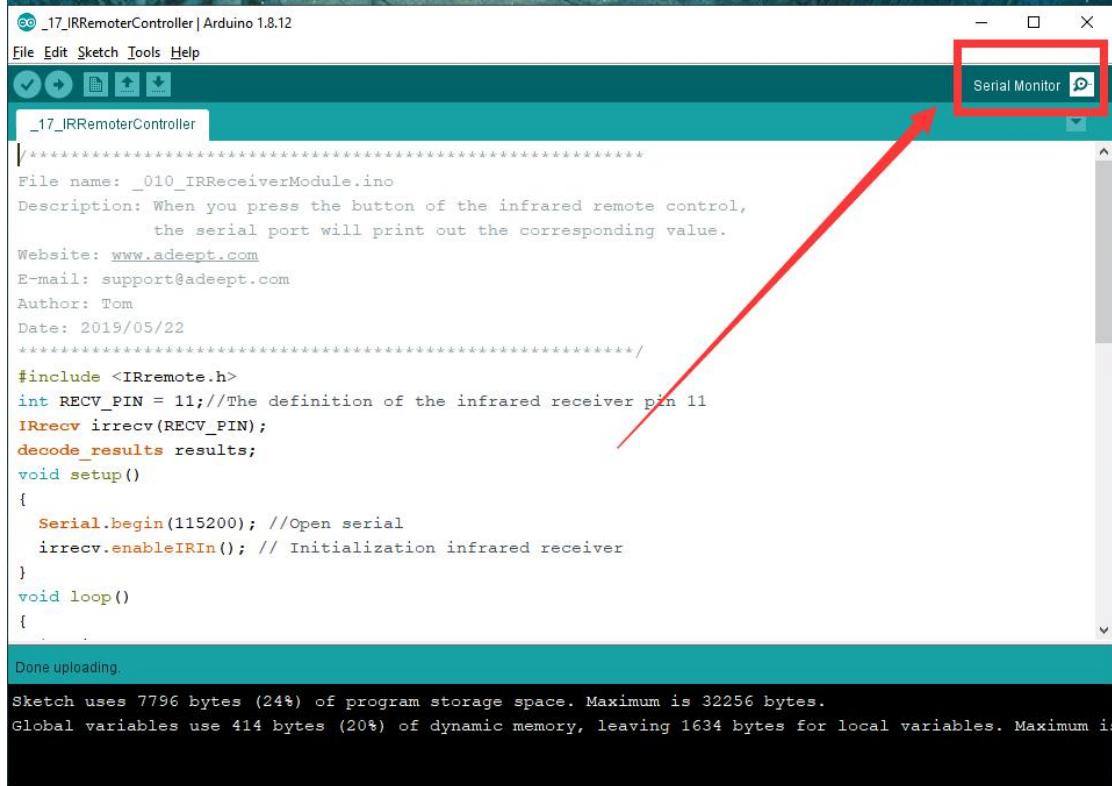
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.



5. After successfully running the program, we need to open the serial monitor on the Arduino software. When we press the swing MPU6050 sensor, the serial monitor will print out the corresponding X, Y, and Z axis values. How to open the serial monitor?

You need to click the "Serial Monitor"  button in the upper right corner, as shown below:

www.adeept.com



```

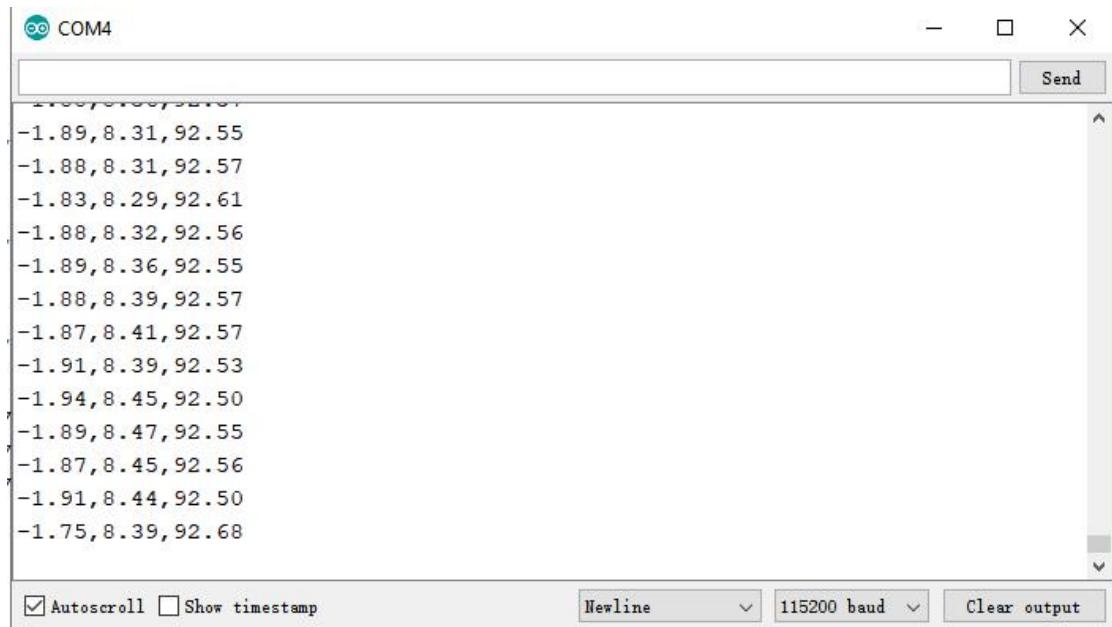
_17_IRRemoterController | Arduino 1.8.12
File Edit Sketch Tools Help
Serial Monitor
_17_IRRemoterController
/*
File name: _010_IRReceiverModule.ino
Description: When you press the button of the infrared remote control,
the serial port will print out the corresponding value.
Website: www.adeept.com
E-mail: support@adeept.com
Author: Tom
Date: 2019/05/22
*/
#include <IRremote.h>
int RECV_PIN = 11;//The definition of the infrared receiver pin 11
IRrecv irrecv(RECV_PIN);
decode_results results;
void setup()
{
    Serial.begin(115200); //Open serial
    irrecv.enableIRIn(); // Initialization infrared receiver
}
void loop()
{
}

Done uploading.

Sketch uses 7796 bytes (24%) of program storage space. Maximum is 32256 bytes.
Global variables use 414 bytes (20%) of dynamic memory, leaving 1634 bytes for local variables. Maximum is

```

6. After clicking, the serial monitor window will pop up, as shown below:



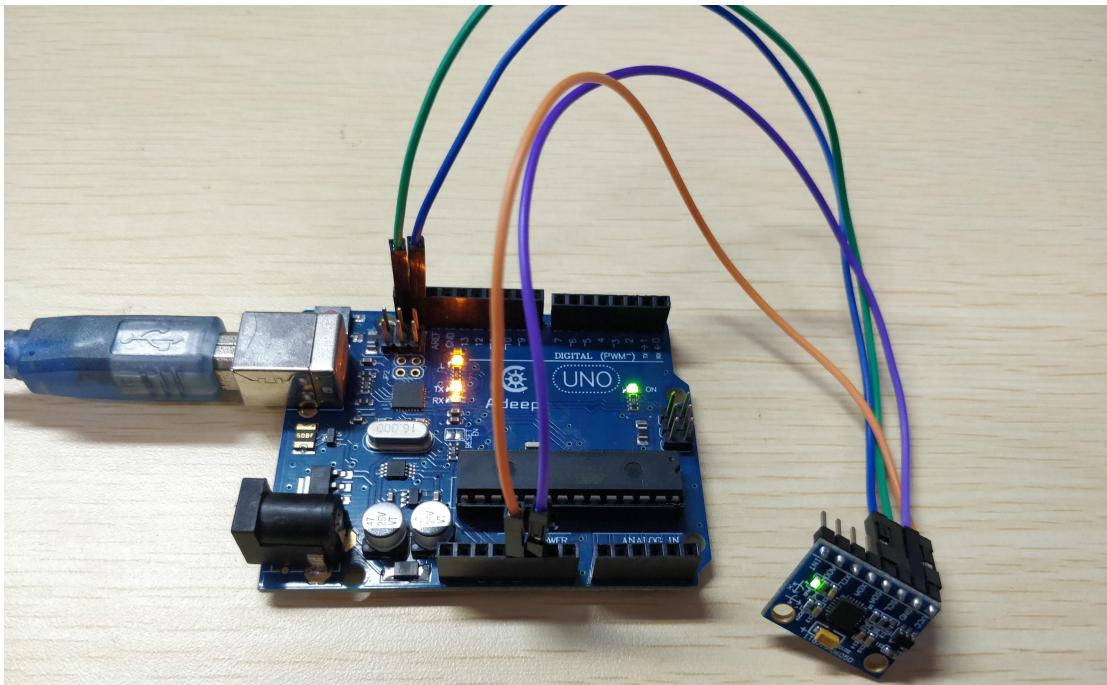
```

COM4
Send
-1.89,8.31,92.55
-1.88,8.31,92.57
-1.83,8.29,92.61
-1.88,8.32,92.56
-1.89,8.36,92.55
-1.88,8.39,92.57
-1.87,8.41,92.57
-1.91,8.39,92.53
-1.94,8.45,92.50
-1.89,8.47,92.55
-1.87,8.45,92.56
-1.91,8.44,92.50
-1.75,8.39,92.68

```

Autoscroll Show timestamp Newline 115200 baud Clear output

7. The physical connection diagram of the experiment is as below:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to read the value of MPU6050 on Arduino UNO. We will introduce how our core code can be achieved:

1.In the setup() function, the six-axis original value is read by the getMotion6() function in the loop statement, and the accelerometer shift amount axo and gyroscope shift amount gxo are calculated.

```

void setup()
{
    Wire.begin();
    Wire.beginTransmission(MPU_addr);
    Wire.write(0x6B); // PWR_MGMT_1 register
    Wire.write(0); // set to zero (wakes up the MPU-6050)
    Wire.endTransmission(true);

    Serial.begin(115200);

    for(int i=0;i<times;i++)
    {
        getMotion6(); //读取六轴原始数值
        axo += AcX; ayo += AcY; azo += AcZ;           //采样和
        gxo += GyX; gyo += GyY; gzo += GyZ;
    }
    axo /= times; ayo /= times; azo /= times; //计算加速度计偏移
    gxo /= times; gyo /= times; gzo /= times; //计算陀螺仪偏移
}

```

2.In the loop() function, print the angular acceleration values of agx, agy, agz through count6Axe().

```

void loop()
{
    count6Axe();
}
.
```

2. Using graphical code blocks to program to read the value of MPU6050 on Arduino UNO

(1)Connecting to GwBlock graphical editor

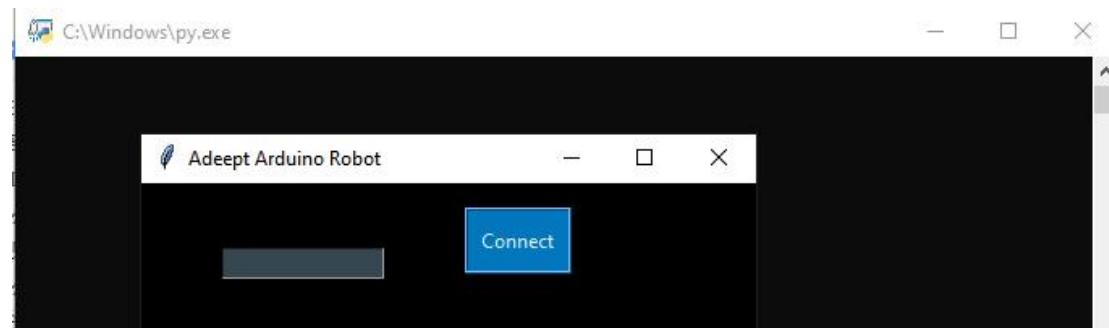
In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1.Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). [Note]: We need to modify #define MPU6050 0 in the opened block_py.ino file to #define MPU6050 1, Then click  and upload

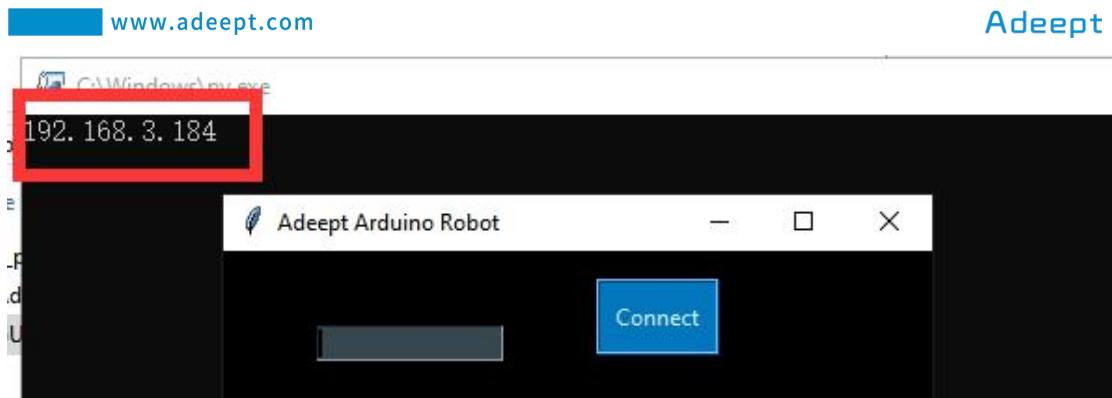
the program to the Arduino UNO. After successful Upload, it will show as below:



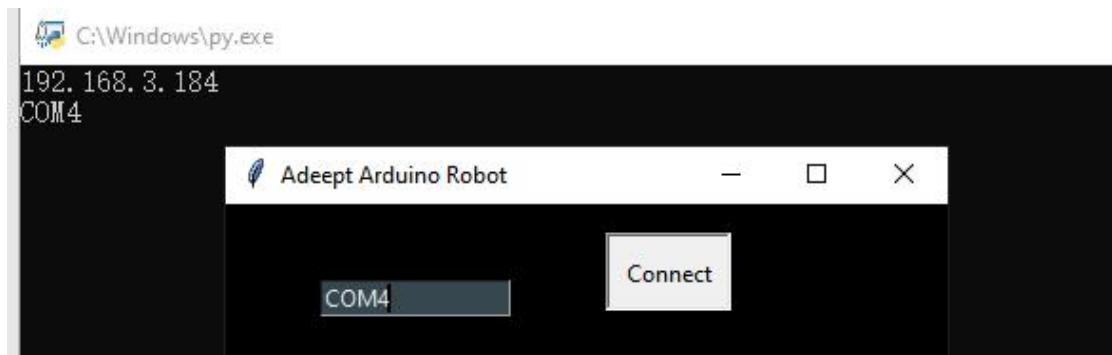
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



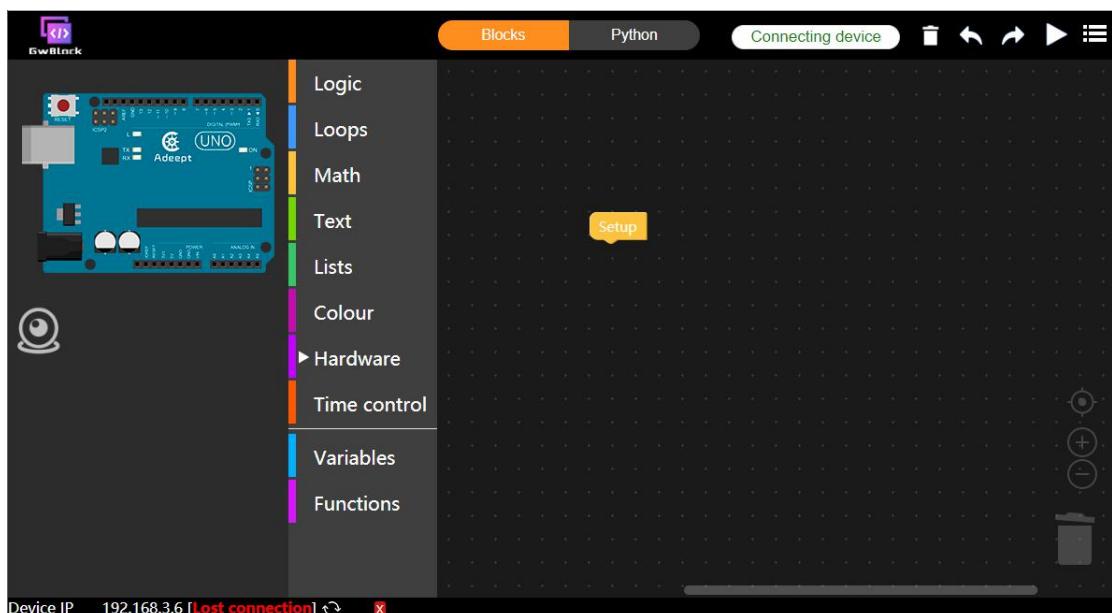
4. Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



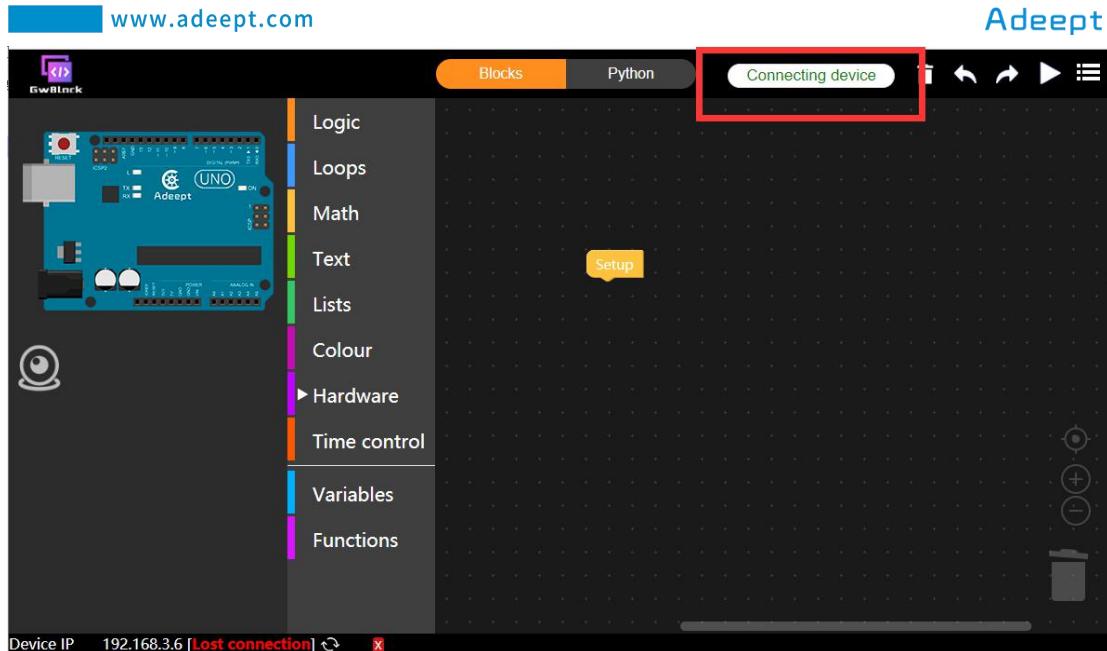
5. Enter the URL of the GwBlock graphical editor in the browser:

[http://www.adeept.com/gwblock/?hd_mo=uno_r3.](http://www.adeept.com/gwblock/?hd_mo=uno_r3)

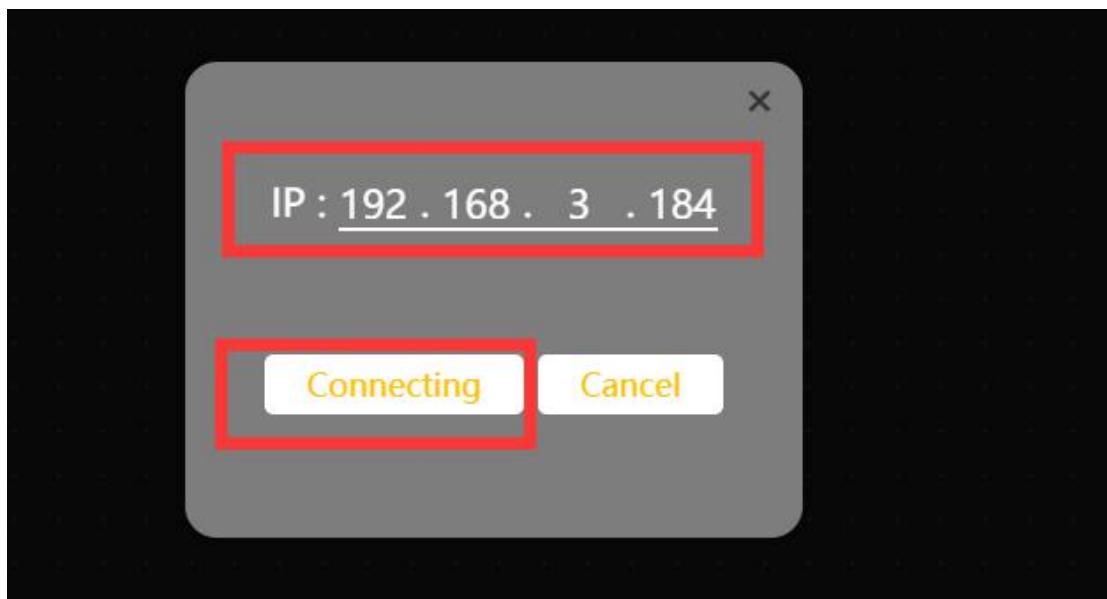
After successfully entering the website, the interface is as follows:



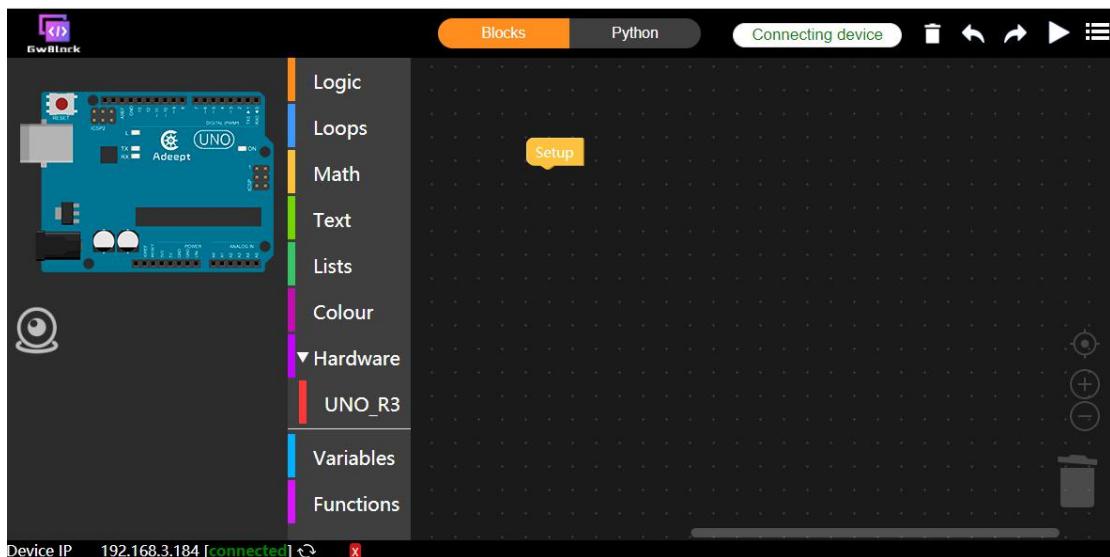
6. Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

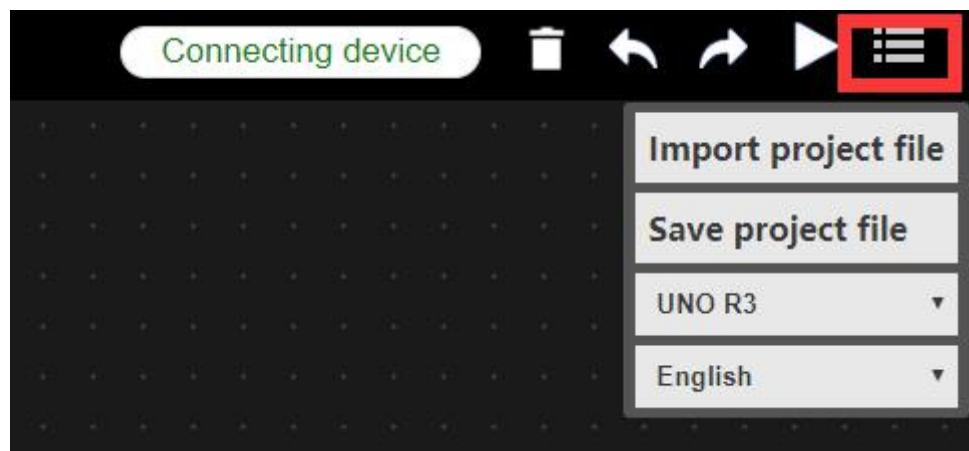


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

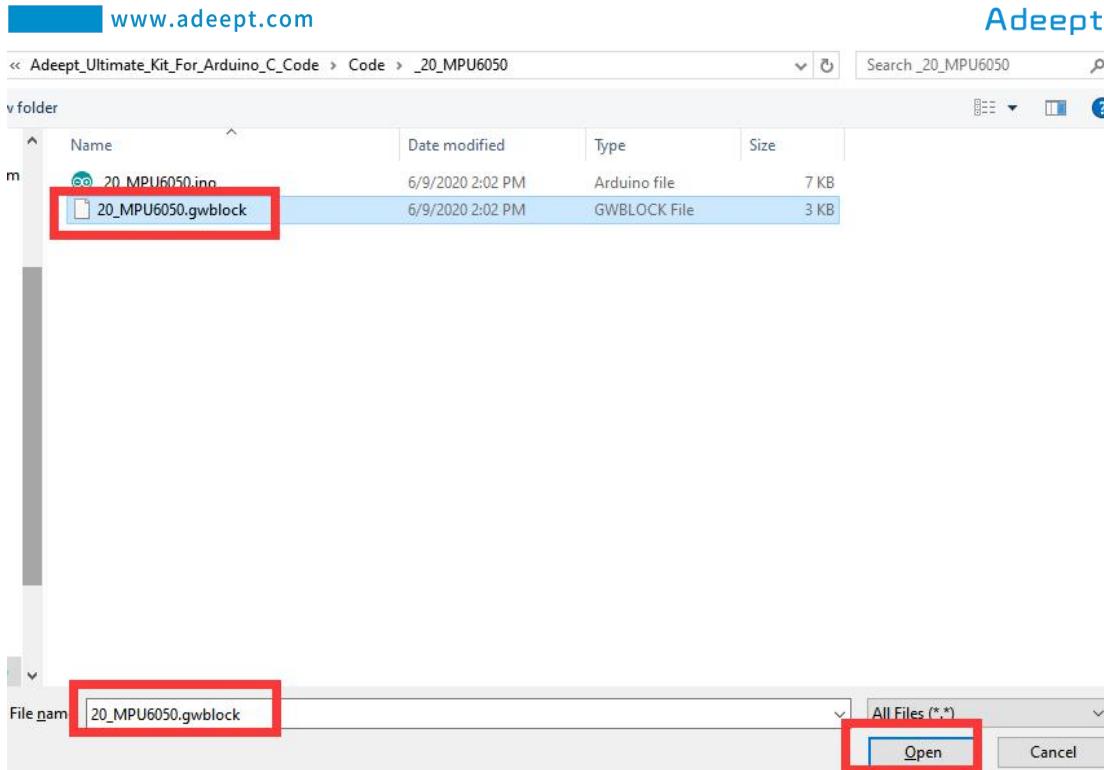
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_20_MP6050

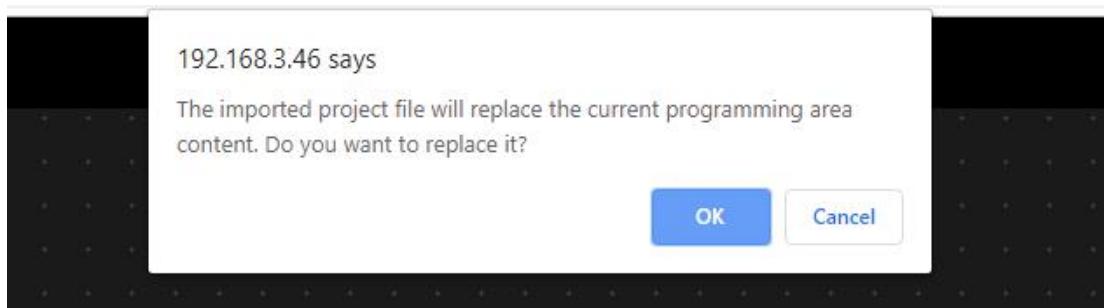
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

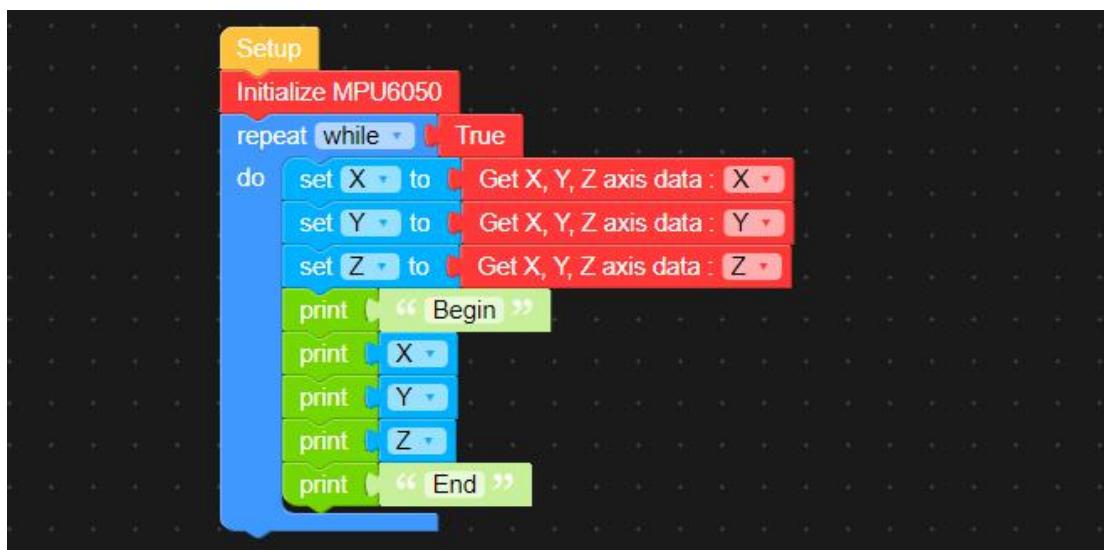
5. Select the "20_MP6050.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



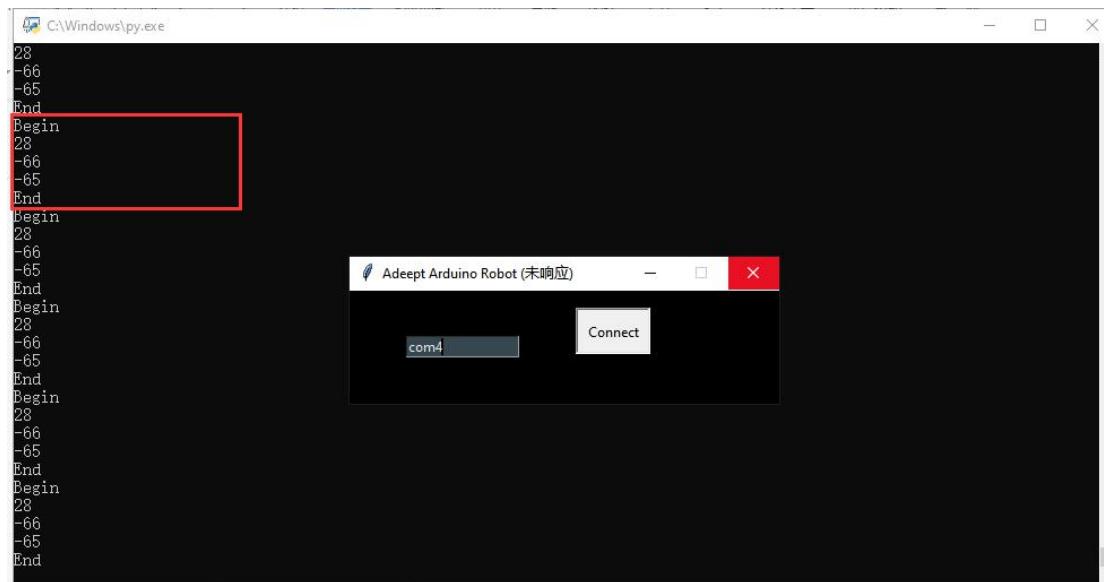
6.Click OK.It will show as below:



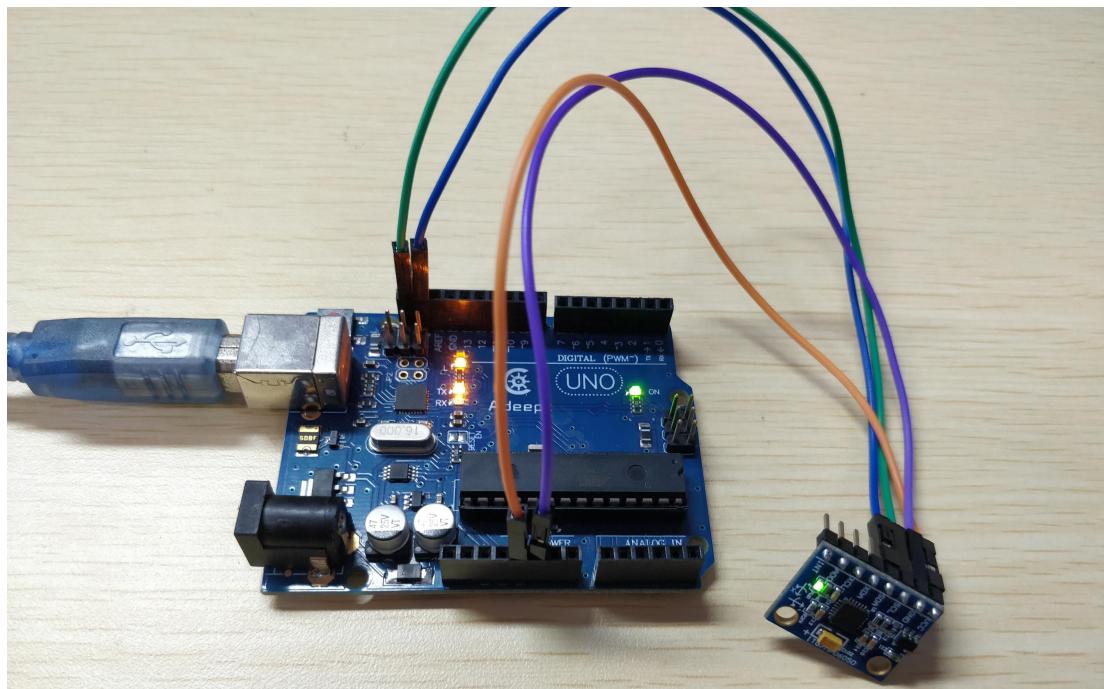
7.It will show as below after successfully opening:



8. Click the button  on the upper right corner. After successfully running the program, open the window of GUI info v1.0.py. It will display the X, Y, Z axis values of the detected MPU6050, indicating that our experimental test is successful.



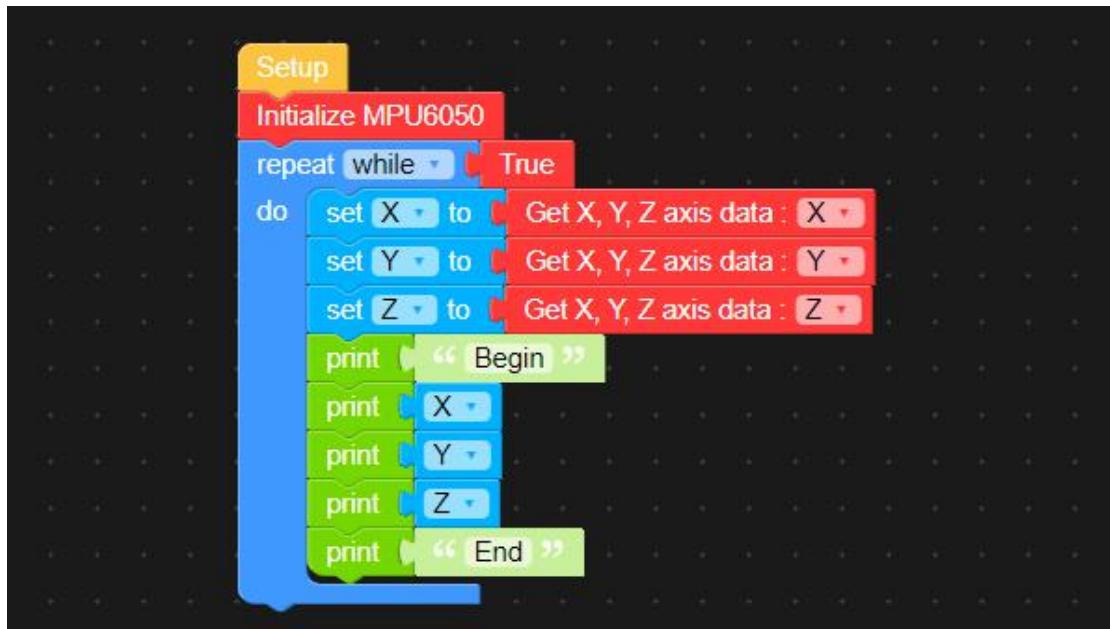
9. The physical connection diagram of the experiment is as follows:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to read the value of MPU6050 on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

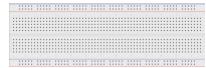
In the GwBlock graphical editor, all code programs are executed from **Setup**. Initialized MPU6050 through **Initialize MPU6050**. Obtain the X, Y, and Z axis data of the MPU6050 through the instruction block **Get X, Y, Z axis data**. Finally print out the X, Y, and Z data through the instruction **print**.



Lesson 21 The Application of the 4*4 Matrix Keyboard

In this lesson, we will learn about the application of the 4*4 Matrix Keyboard.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
4*4 Matrix Keyboard	1	

2. The introduction of 4*4 matrix keyboard

(1) The 4*4 Matrix Keyboard

The 4x4 matrix keyboard is a keyboard set arranged in a matrix used in Arduino external devices. In a 4*4 matrix keyboard, each horizontal and vertical line is not connected directly at the intersection, but is connected by a single key. In this way, a port (such as port P1) can form $4 \times 4 = 16$ keys, which is twice as much as using port line for keyboard directly. Moreover, the more lines, the more obvious the difference is. For example, one more line can form a 20-key keyboard, while using port line can only create one more key (9 keys). Thus it can be seen that it is reasonable to use the matrix method to make the keyboard when the number of keys needed is large.



(2) The working principle of the 4*4 Matrix Keyboard

The keys are arranged at the intersection of the row and column lines, and the row and column lines are connected to both ends of the key switch respectively. When the key is not pressed, all input terminals are high level, which means no key is pressed. The line output is low level. Once a key is pressed, the input line will be pulled down. In this way, you can know whether a key is pressed by reading the state of the input line. The line is connected to the +5V power supply through a pull-up resistor.

A "line scan method" is used to determine which key on the matrix keyboard is pressed

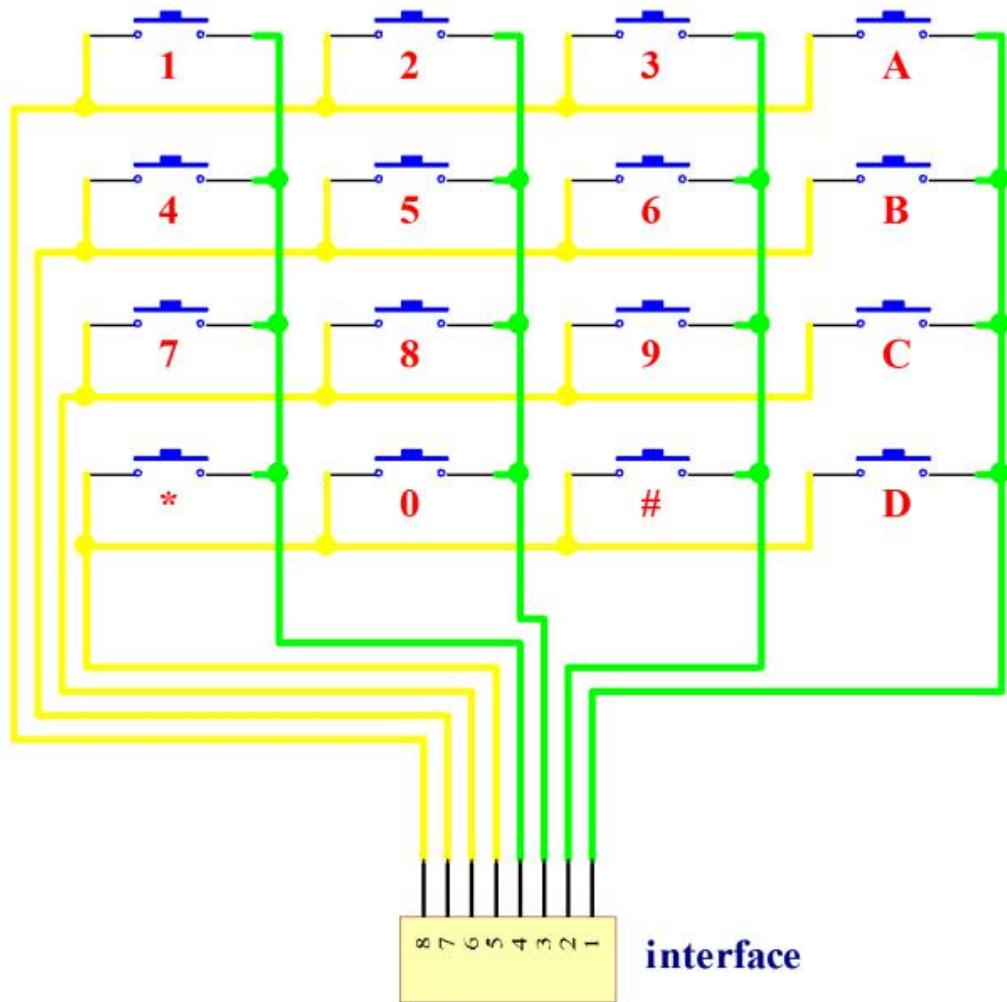
Step 1: make the line the input line of programming and the column line the output line, pull down all the column lines, and judge the change of the line. If there is a key pressed, the corresponding line pressed will be pulled down, otherwise all the

line lines will be at high level.

Step 2: after the first step is judged to have a key pressed, the mechanical jitter will be eliminated after a delay of 10ms, and the row value will be read again. If the line is still at a low level, the next step will be entered; otherwise, the first step will be returned to re-judge.

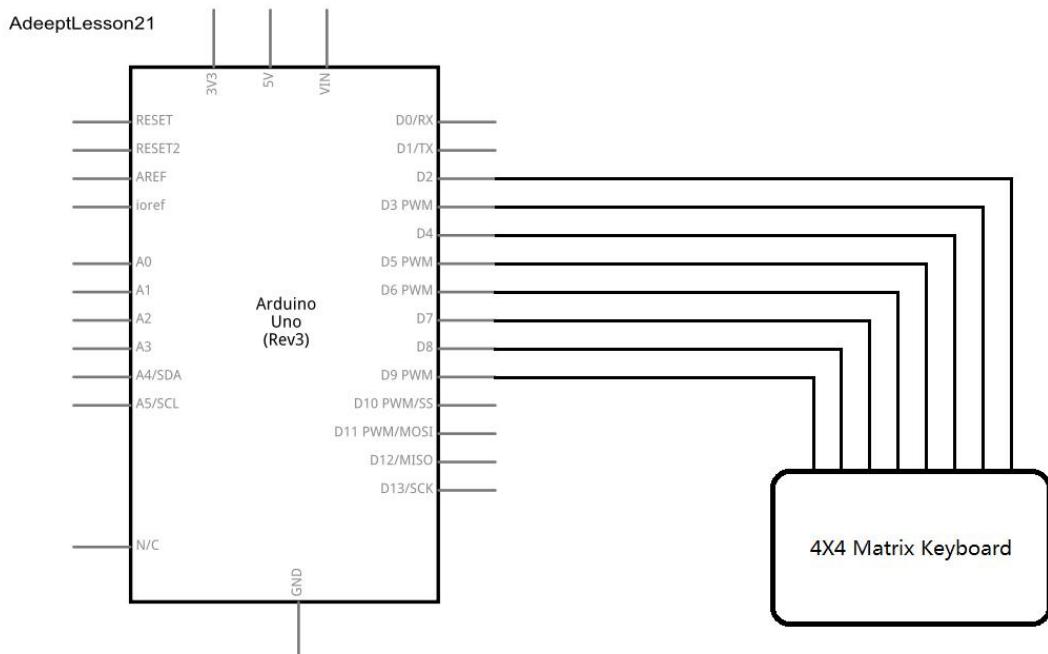
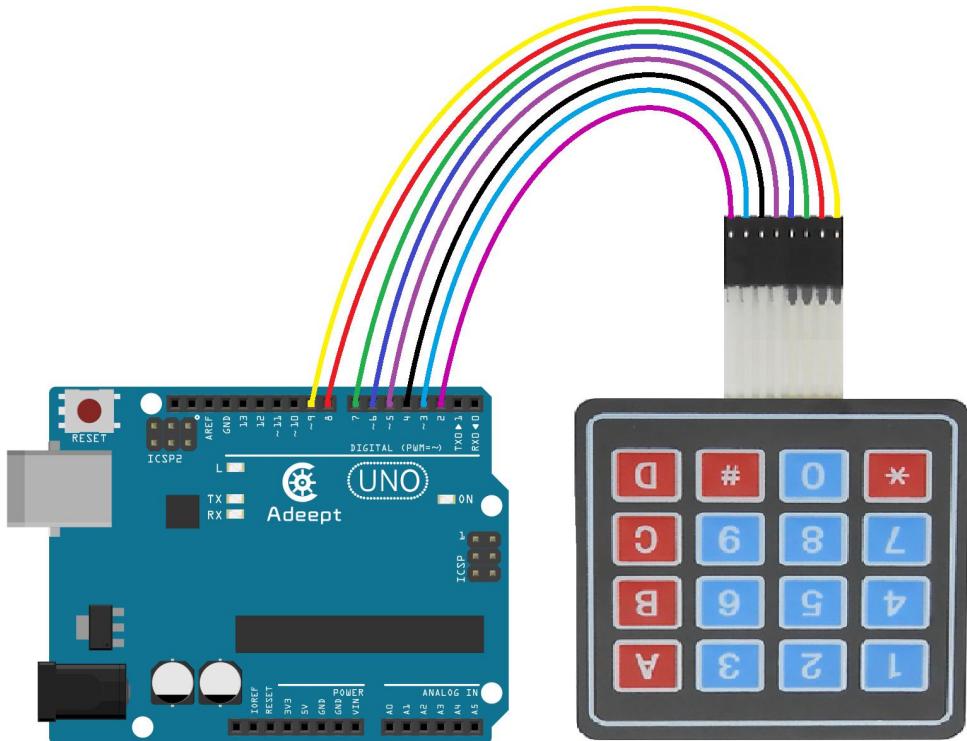
Step 3: start scanning the position of the key, and use line by line scanning. Every 1ms interval, pull down the first column, the second column, the third column and the fourth column respectively. No matter which column you lower, the other three columns are at high level. Read the row value to find the location of the key and store the row and column values in it.

Step 4: find the row value and the column value from the register and combine them to get the key value, which is coded from the first row to the fourth row from the first row to the fourth row, from "0000" to "1111", and then display Decode. Finally, the key number is displayed. Principle of dynamic scanning of digital tube: the 7 segments and the decimal point of the digital tube are all composed of LED blocks, and the display mode is divided into static display and dynamic display. When the digital tube is displayed in static mode, the bit selection signals of its total positive tube are all low level, and the common segment selection lines of the four digital tubes a, b, c, d, e, f, g and dp are respectively connected with the 8 I/O port lines of CPLD. When the digital tube is displayed, only the low level should be sent to the corresponding segment selection line. Dynamic display of digital tube in the way, there can be only one at a time to light the digital tube display digital, the rest is in a state of the gate, a selected code port signal changes, port signal segment code also should make corresponding change. The stay time of each Character is usually 1-5 ms, using visual inertia, one eye can see fairly stable on the digital tube digital display.



3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. Use $220\ \Omega$ resistor. As shown in the following figure:



4. The Application of the 4*4 Matrix Keyboard

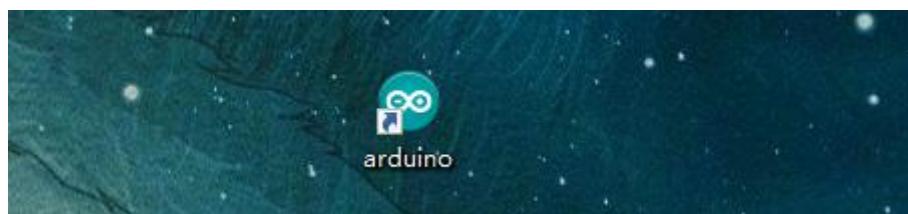
We provide two different methods to read the value of 4*4 matrix keyboard. One

is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

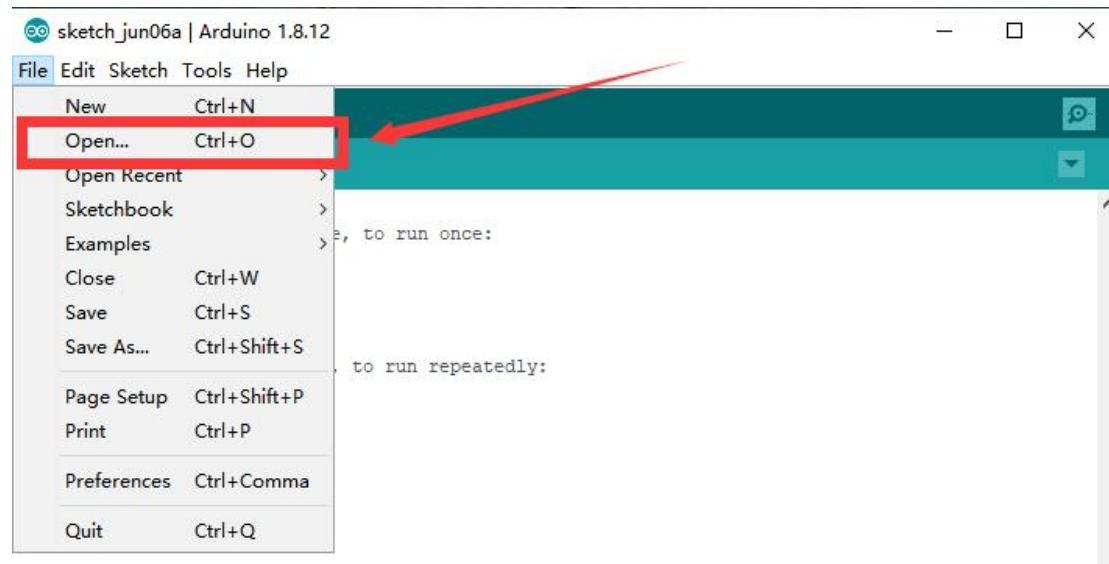
1.Using C language to program to read the value of 4*4 matrix keyboard on Arduino UNO

(1)Compile and run the code program of this course

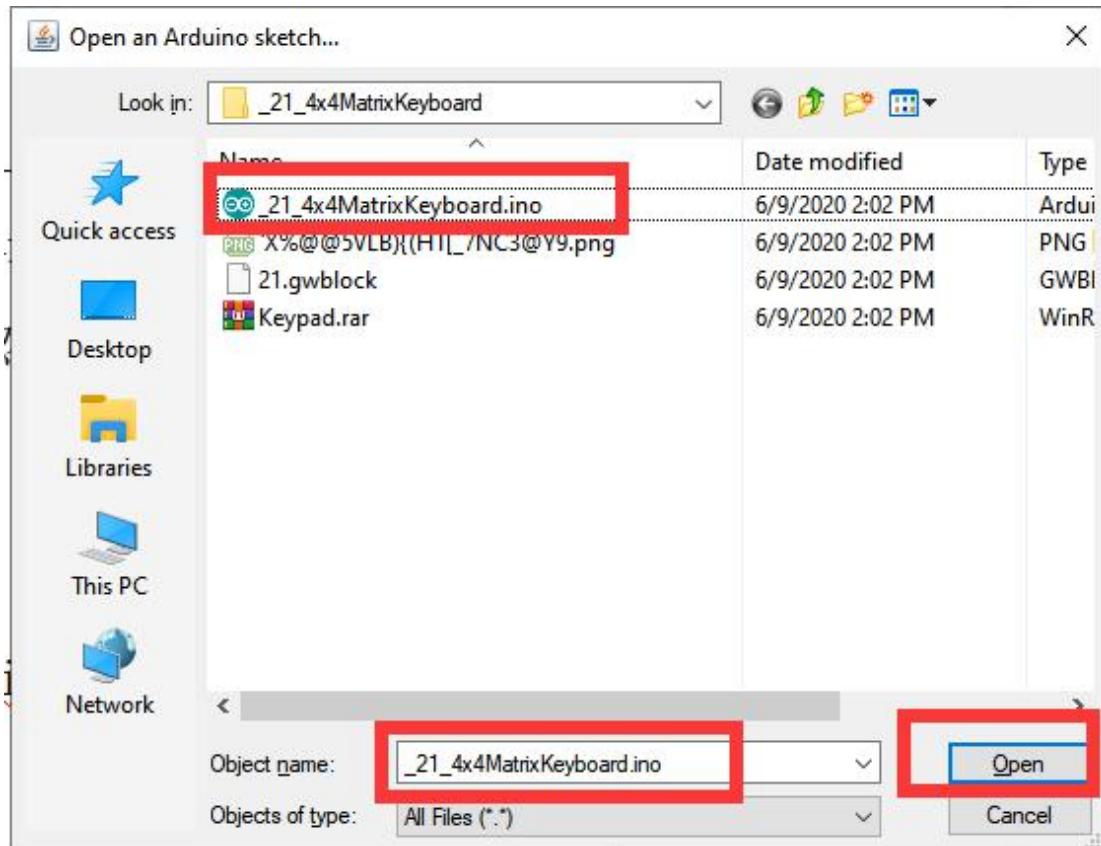
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\21_4x4MatrixKeyboard directory. Select 21_4x4MatrixKeyboard.ino. This file is the code program we need in this course. Then click Open.



4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

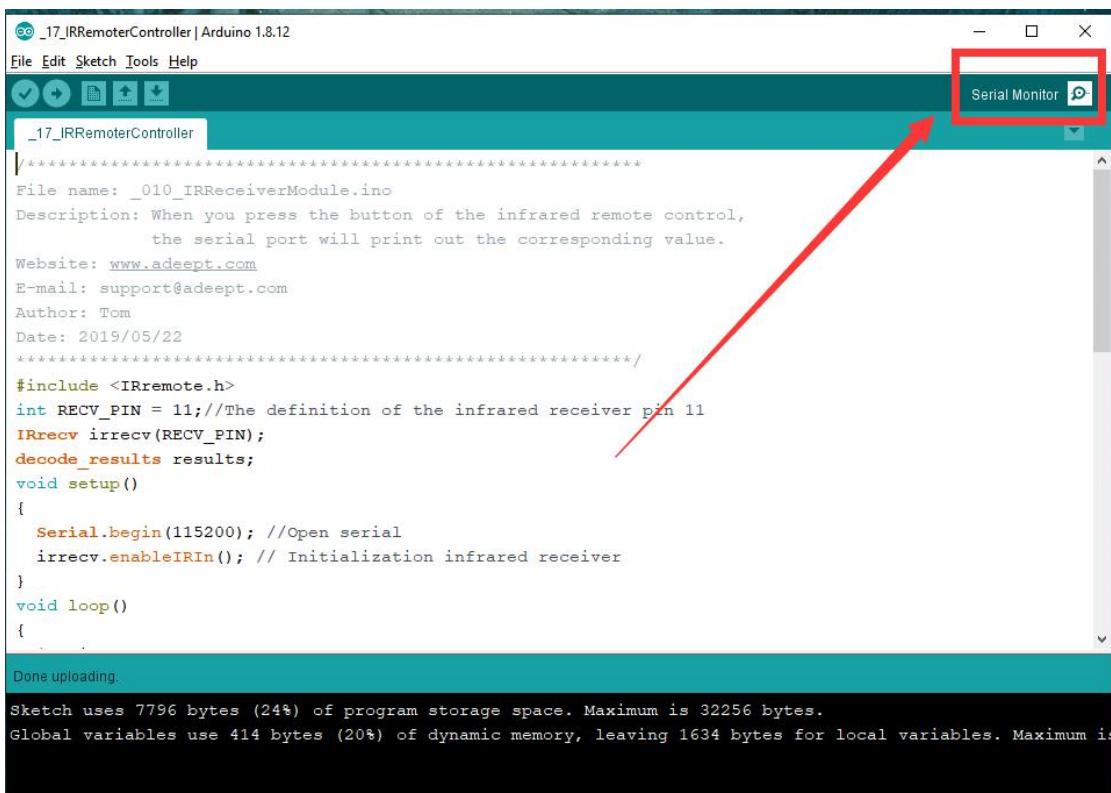
```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After successfully running the program, we need to open the serial monitor on the Arduino software. When we press the number on the 4x4 matrix button module, the serial monitor will print the corresponding value. How to open the serial monitor?

You need to click the "Serial Monitor"  in the upper right corner, as shown below:



The screenshot shows the Arduino IDE interface. The title bar reads '_17_IRRemoterController | Arduino 1.8.12'. The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with various icons. On the right side of the interface, there is a 'Serial Monitor' button with a small icon, which is highlighted with a red rectangular box and an arrow pointing to it from the bottom right. The main area displays the code for '_17_IRRemoterController' and some descriptive text about the sketch.

```

_17_IRRemoterController | Arduino 1.8.12
File Edit Sketch Tools Help
Serial Monitor
_17_IRRemoterController
/*
File name: _010_IRReceiverModule.ino
Description: When you press the button of the infrared remote control,
the serial port will print out the corresponding value.
Website: www.adeept.com
E-mail: support@adeept.com
Author: Tom
Date: 2019/05/22
*/
#include <IRremote.h>
int RECV_PIN = 11;//The definition of the infrared receiver pin 11
IRrecv irrecv(RECV_PIN);
decode_results results;
void setup()
{
    Serial.begin(115200); //Open serial
    irrecv.enableIRIn(); // Initialization infrared receiver
}
void loop()
{
}

Done uploading.

Sketch uses 7796 bytes (24%) of program storage space. Maximum is 32256 bytes.
Global variables use 414 bytes (20%) of dynamic memory, leaving 1634 bytes for local variables. Maximum is

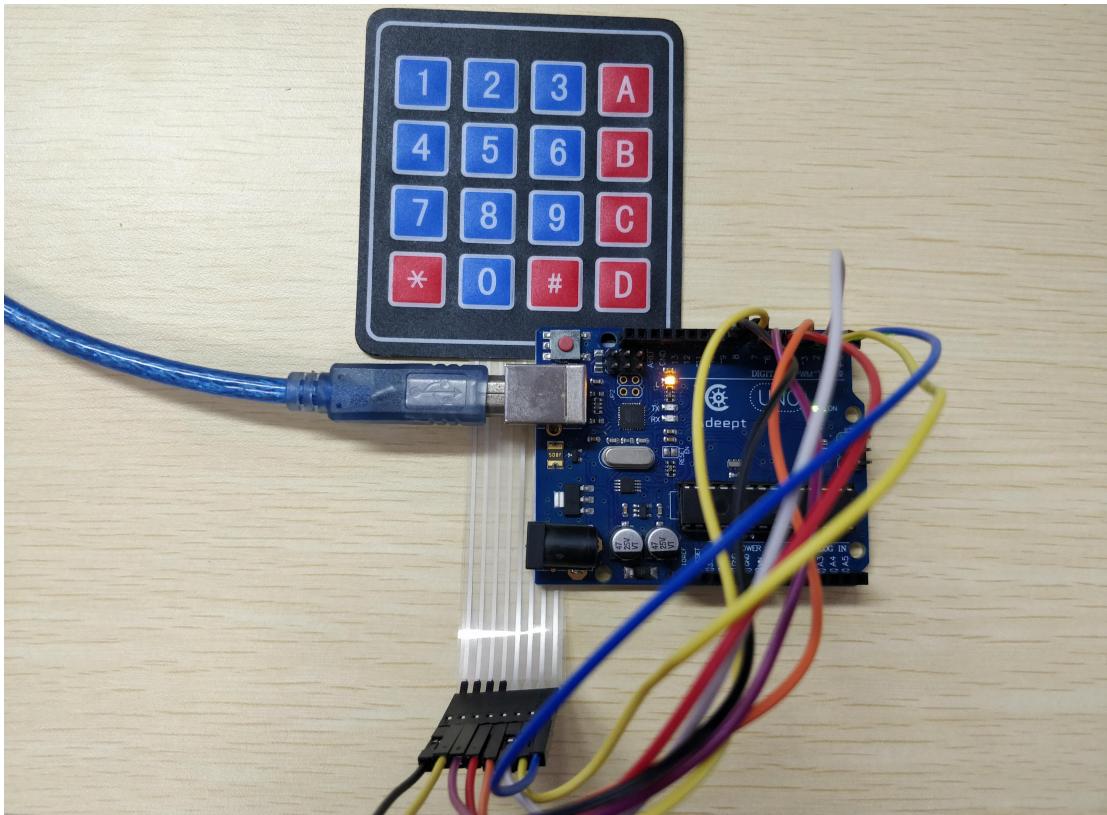
```

6. After clicking , the serial monitor window will pop up. Press the number on the 4x4 matrix button module, the serial monitor will print out the corresponding value. As shown below:



The screenshot shows the Serial Monitor window. The title bar says 'COM4'. The main area displays the following sequence of numbers: 1, 2, 3, 4, 5, 6, 7, 8, 9, 0. At the bottom of the window, there are several controls: 'Autoscroll' (checked), 'Show timestamp' (unchecked), 'Newline' (dropdown menu), '115200 baud' (dropdown menu), and 'Clear output' (button).

7. The physical connection diagram of the experiment is as follows:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to read the value of 4*4 matrix keyboard on Arduino UNO. We will introduce how our core code can be achieved:

1. In the setup() function, open the serial port with Serial.begin(115200).

```
void setup() {
    Serial.begin(115200); //Open serial
}
```

2. In the loop() function, read the key value of the 4x4 matrix key via customKeypad.getKey(). Output the key value on the serial port with Serial.println(customKey).

```
void loop() {
    char customKey = customKeypad.getKey(); //Read Key data
    if (customKey) {
        Serial.println(customKey); //send the key data by serial port (UART)
    }
}
```

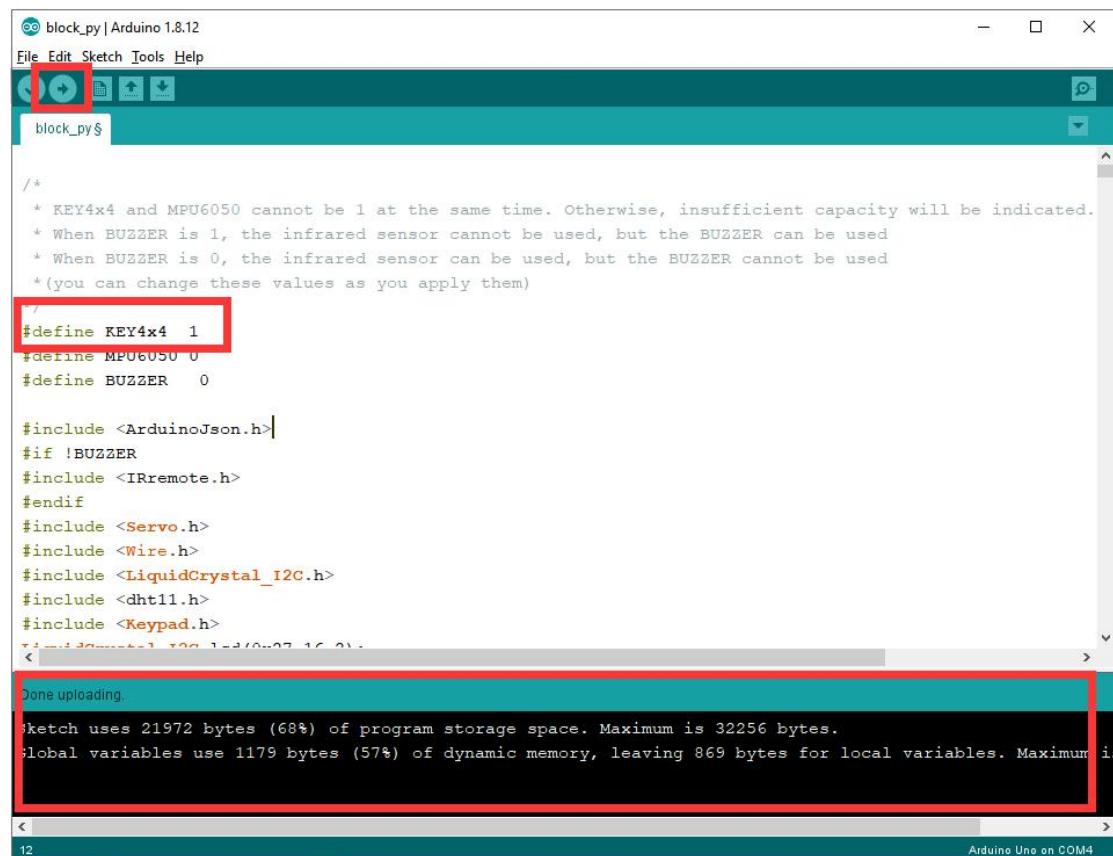
2. Programming to read the value of 4*4 matrix keyboard on Arduino UNO with graphical code blocks

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1.Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). [Note]: We need to modify #define KEY4x4 0 in the opened block_py.ino file to #define KEY4x4 1. Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:



```
block_py | Arduino 1.8.12
File Edit Sketch Tools Help
block_py §

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 1
#define MPU6050 0
#define BUZZER 0

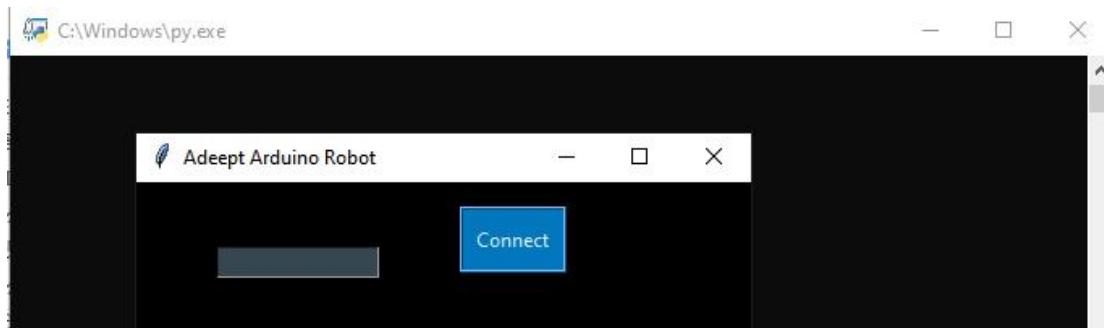
#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <dht11.h>
#include <Keypad.h>
<DHT.h>
<OneWire.h>

Done uploading.

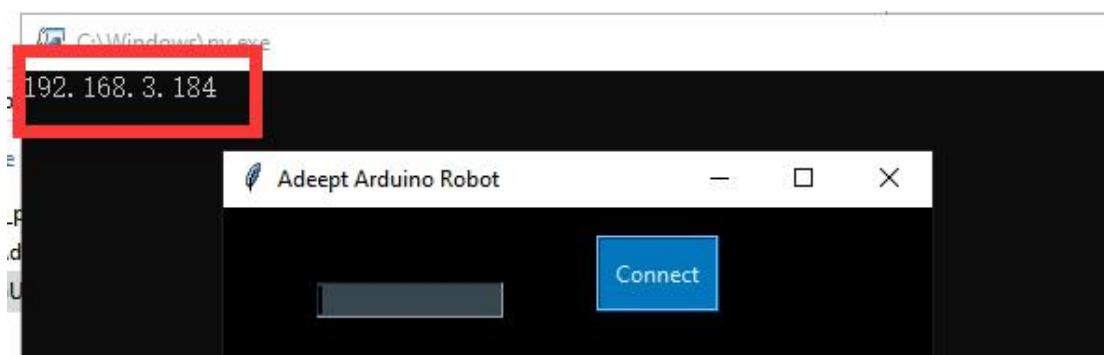
Sketch uses 21972 bytes (68%) of program storage space. Maximum is 32256 bytes.
Global variables use 1179 bytes (57%) of dynamic memory, leaving 869 bytes for local variables. Maximum is
```

2.Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again.Then open the websocket folder. Double-click to open the GUI info v1.0.py file.It will show as

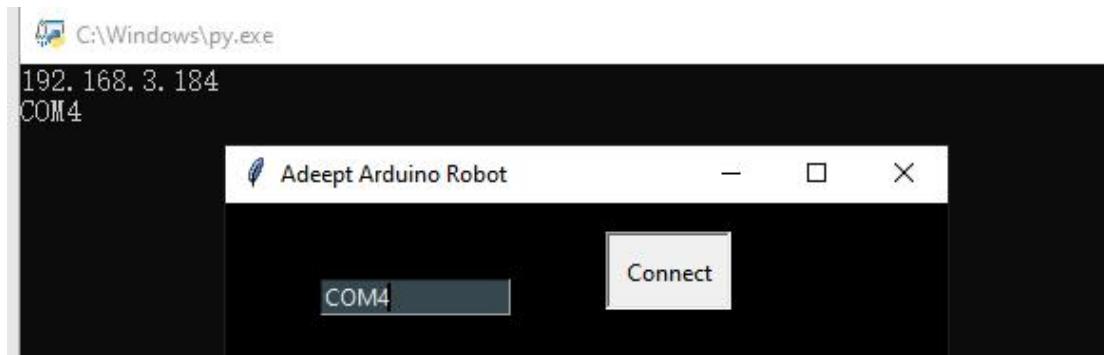
below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



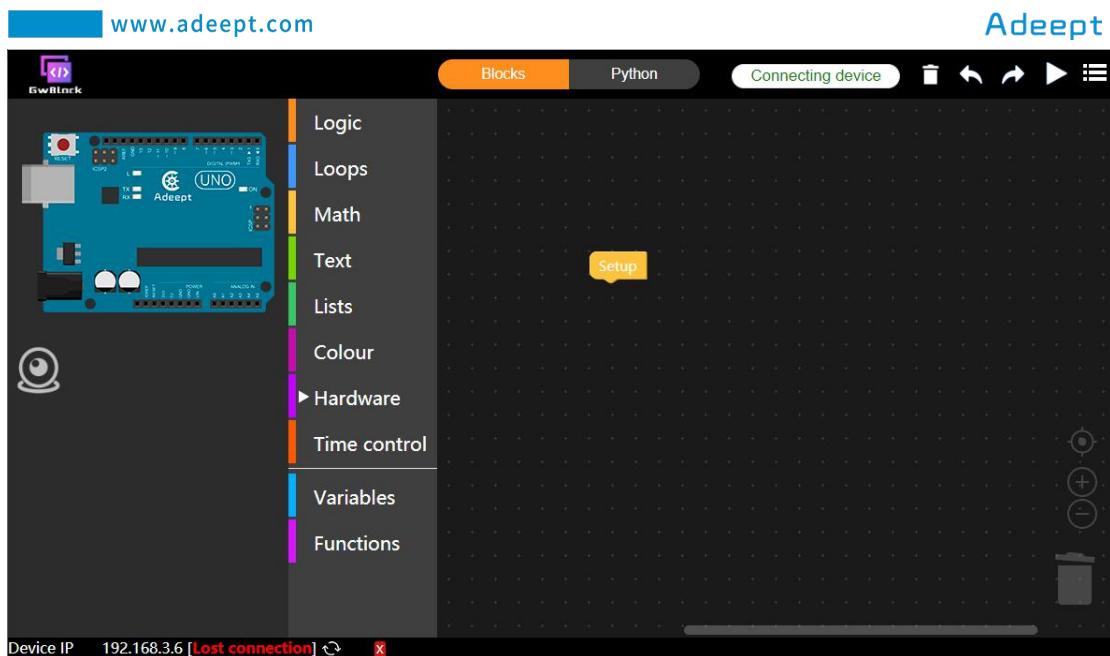
4. Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



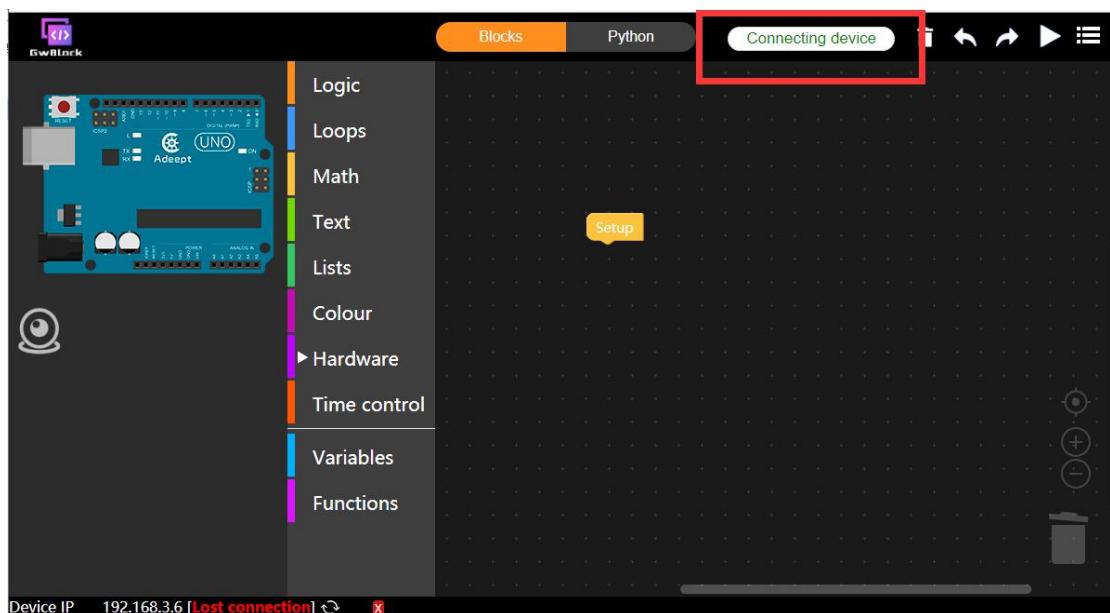
5. Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

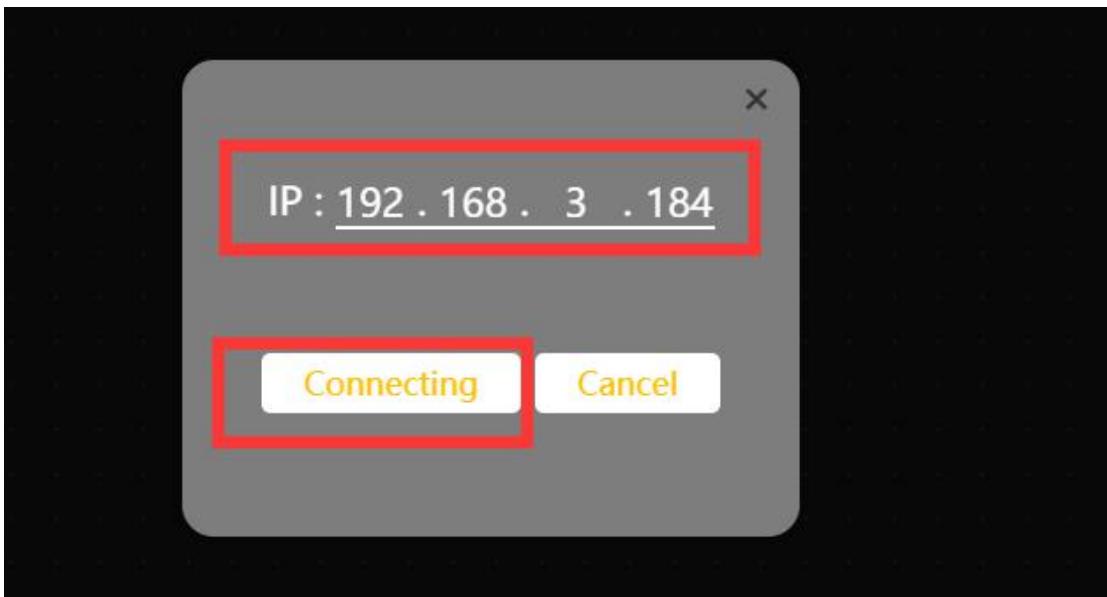
After successfully entering the website, the interface is as follows:



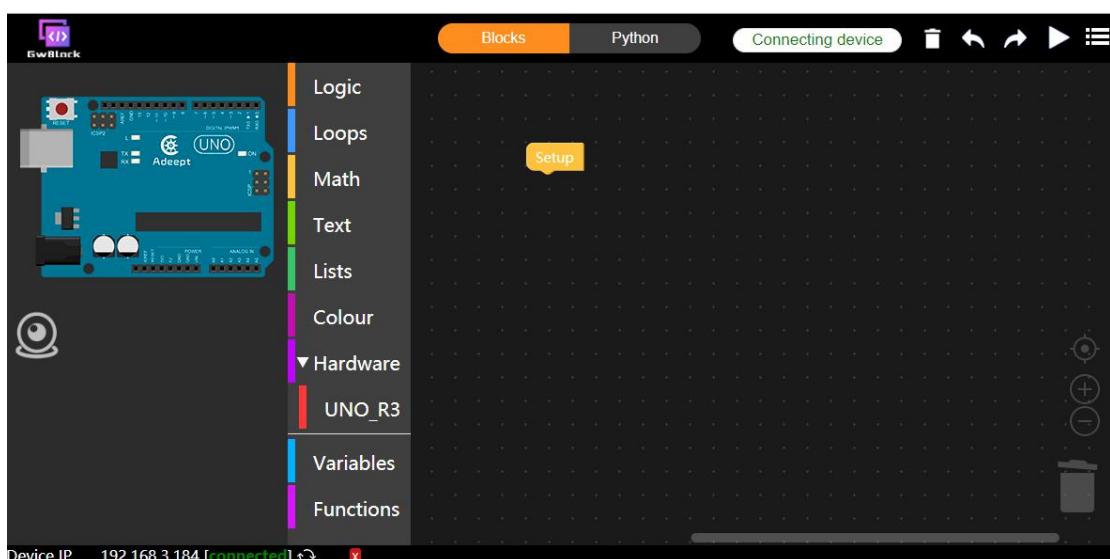
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8. After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:



(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

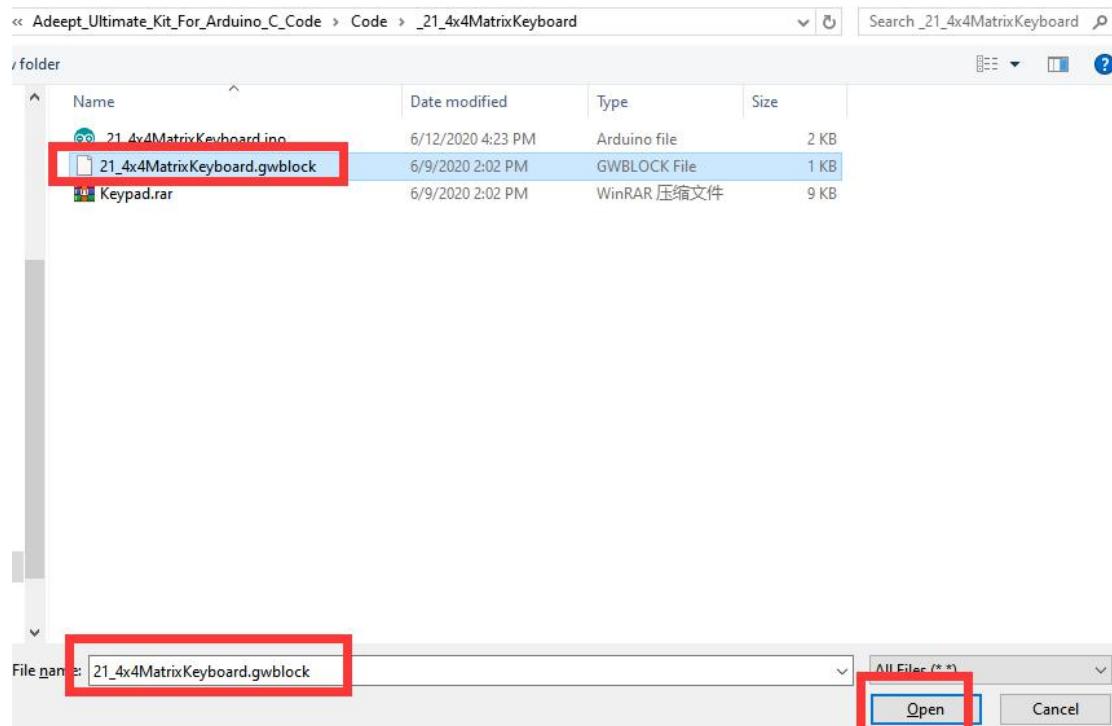
Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_21_4x4MatrixKeyboard

It will show as below:

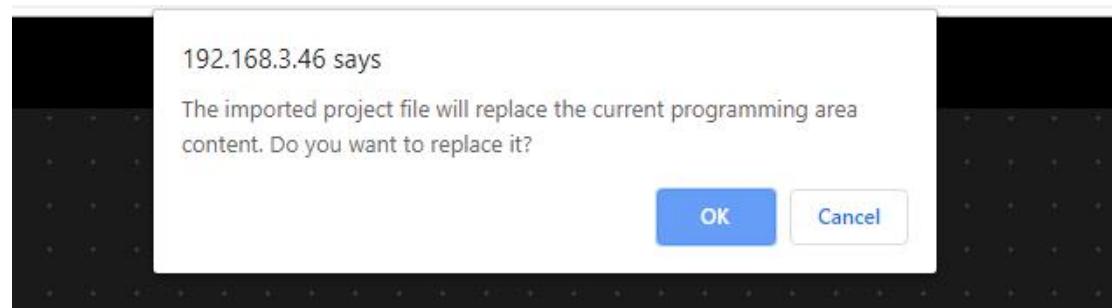
Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

5. Select the "21_4x4MatrixKeyboard.gwblock" file. This file is the graphical code

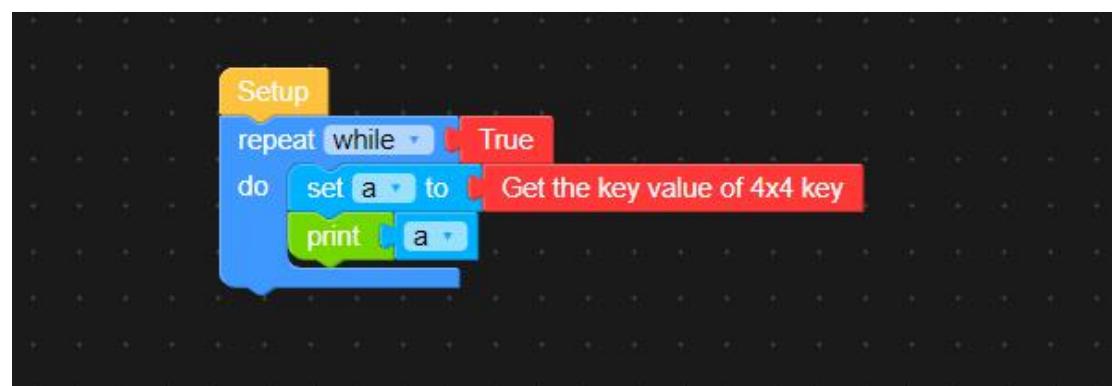
program for our lesson. Click "Open" in the lower right corner. It will show as below:



6. Click OK. It will show as below:



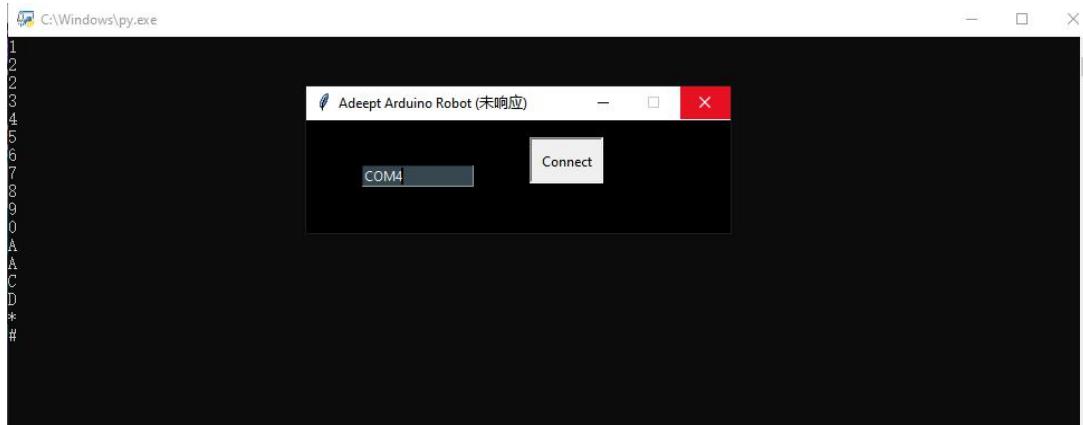
7. It will show as below after successfully opening:



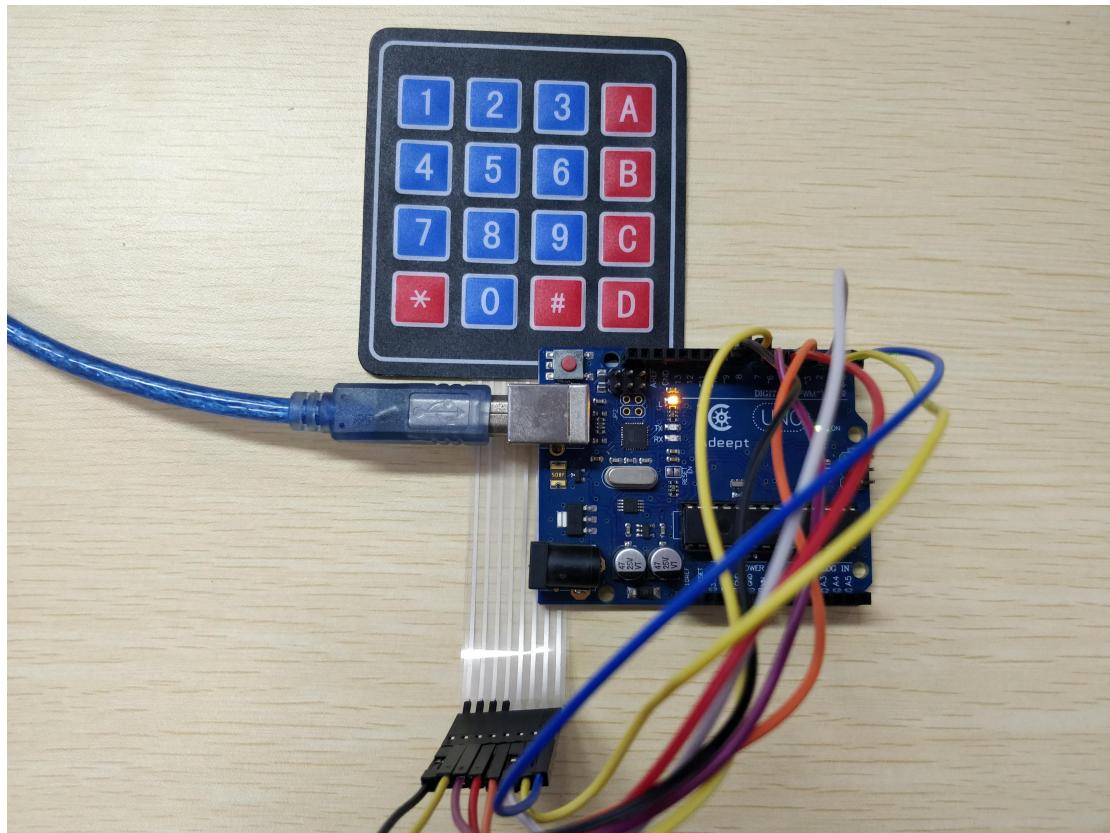
8. Click the button  on the upper right corner. Open the GUI info v1.0.py window.

When you press the key numbers on the 4x4 matrix keyboard, you will find that the

corresponding key values will also be displayed on the window, indicating that our experimental test is successful.



9. The physical connection diagram of the experiment is as follows:

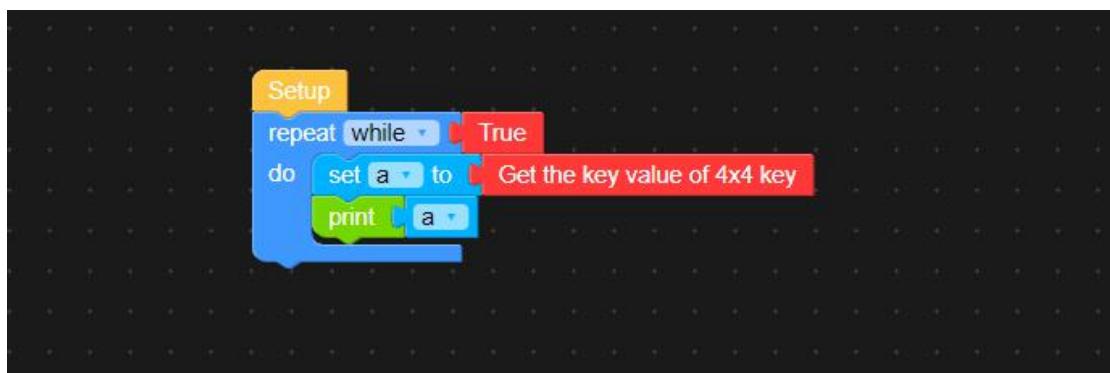


(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to read the value of 4*4 matrix keyboard on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

In the GwBlock graphical editor, all code programs are executed from **Setup**. Get

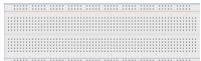
the key value of the 4x4 matrix keyboard through the instruction **set [a] to [Get the key value of 4x4 key]**. Print out the key value with **print [a]**.



Lesson 22 Controlling the DC Motor

In this lesson, we will learn how to control the DC Motor.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Adeept DC Motor Module	1	
Male to Female Jumper Wires	4	

2. The introduction of the DC Motor

(1) The DC Motor

A DC motor is a motor that converts DC electrical energy into mechanical energy. Because of its good speed regulation performance, it is widely used in electric towing. DC motors are divided into three categories: permanent magnet, other excitation, and self-excitation according to the excitation mode. Self-excitation is divided into three types: parallel excitation, series excitation, and compound excitation.

When DC power is supplied to the armature winding through the brush, the N-pole conductor on the surface of the armature can flow current in the same direction. According to the left-hand rule, the conductor will be subjected to

counterclockwise torque; a part of the conductor under the S pole of the armature surface also flow current in the same direction, and the conductor will also be subjected to a counterclockwise moment according to the left-hand rule. In this way, the entire armature winding, that is, the rotor will rotate counterclockwise, and the input DC energy is converted into mechanical energy output on the rotor shaft. It is composed of stator and rotor, stator: base, main magnetic pole, commutation pole, and brush device, etc. rotor (armature): armature core, armature winding, commutator, rotating shaft and fan, etc.



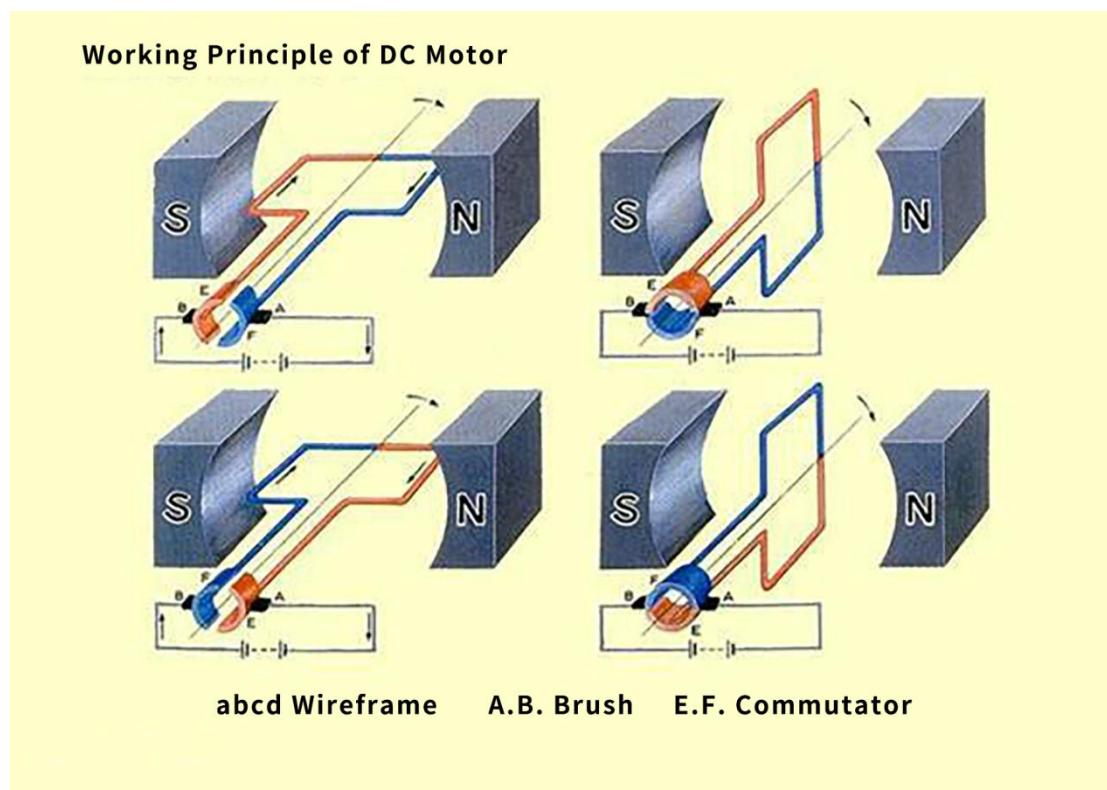
(2)Working principle of the DC Motor

1. The DC power current flows along the positive pole of the power supply to the left brush. The brush and the commutator rub against each other. The current flows into the coil through the left commutator, flows out from the right side of the coil, and passes through the right commutator And the brush on the right flows back to the negative pole of the power supply, forming a closed loop.
2. Because the coil is in the magnetic field of the main magnetic pole, the coil will be subjected to the electromagnetic force. The two sides of the coil have different current directions (the current on the left flows inward and the right flows outward). Electromagnetic forces of the same magnitude and opposite directions, these two electromagnetic forces just form an electromagnetic torque. Under the pulling of the electromagnetic torque, the coil starts to rotate. In the DC motor, the coil is embedded

in the rotor slot, and the motor starts to rotate.

3. The left and right commutator segments follow the shaft, and the brush is stationary. After one turn, the right coil reaches the left and the left coil reaches the right, but due to the presence of the commutator, it is now in the left coil. The direction of the current flows in the same direction as the current changed by the coil on the left, so the direction of the electromagnetic force received does not change, and the same is true on the right. Therefore, from the perspective of space, the direction of the electromagnetic force received by the coil at the same position is always the same, which ensures the cyclic rotation of the motor.

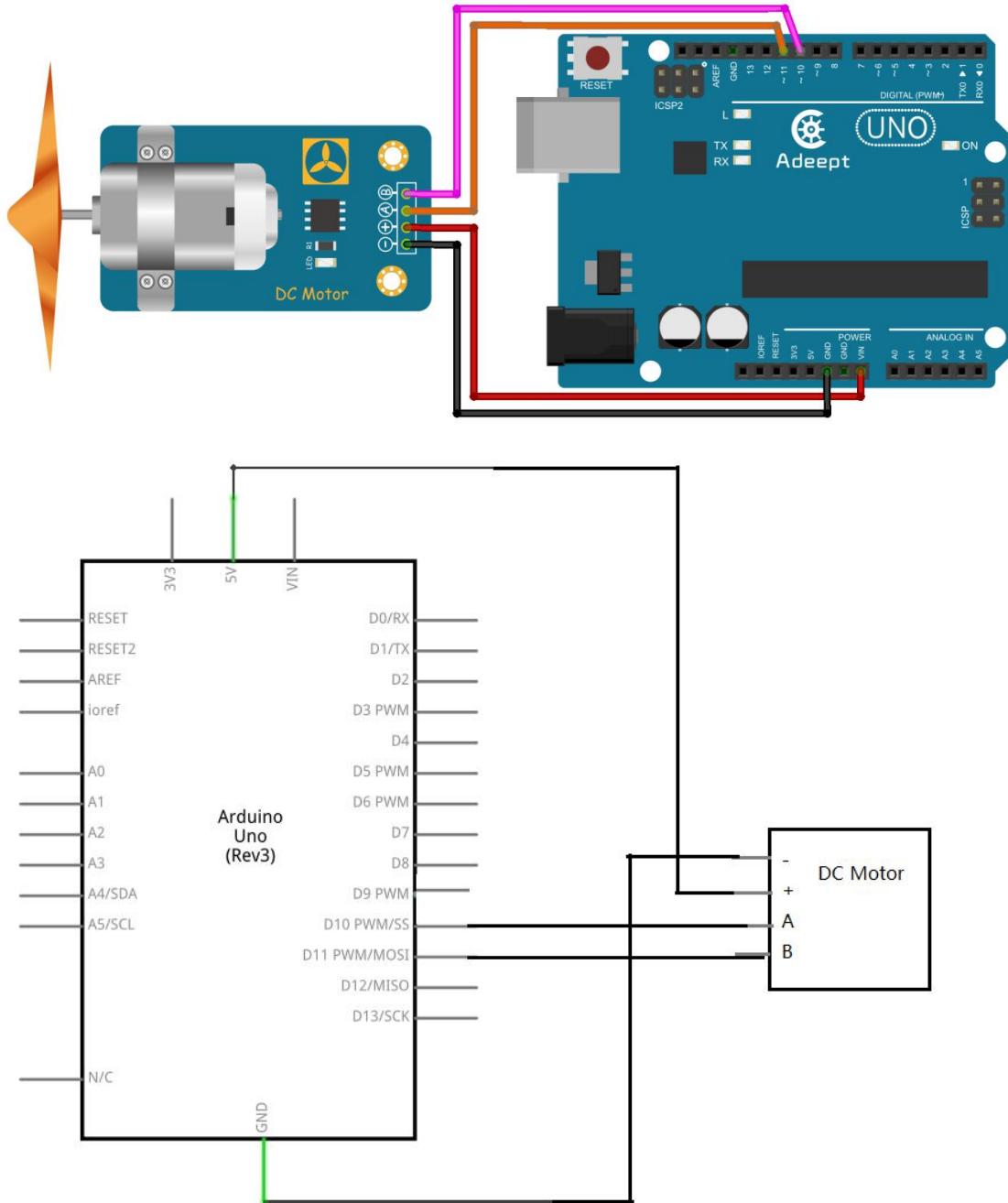
4. But for a coil, because the magnetic field is different when the coil is turned to different positions, the electromagnetic force received by the coil is also constantly changing, so the coil turns unstable, suddenly and slowly. Therefore, you can install more coils to ensure that the coils are evenly and stably stressed.



3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following

figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:



4. Controlling MPU6050

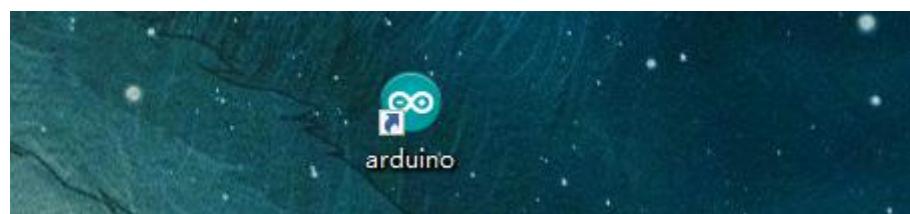
We provide two different methods to control the DC motor. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code

block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

1. Using C language to program to control DC Motor on Arduino UNO

(1) Compile and run the code program of this course

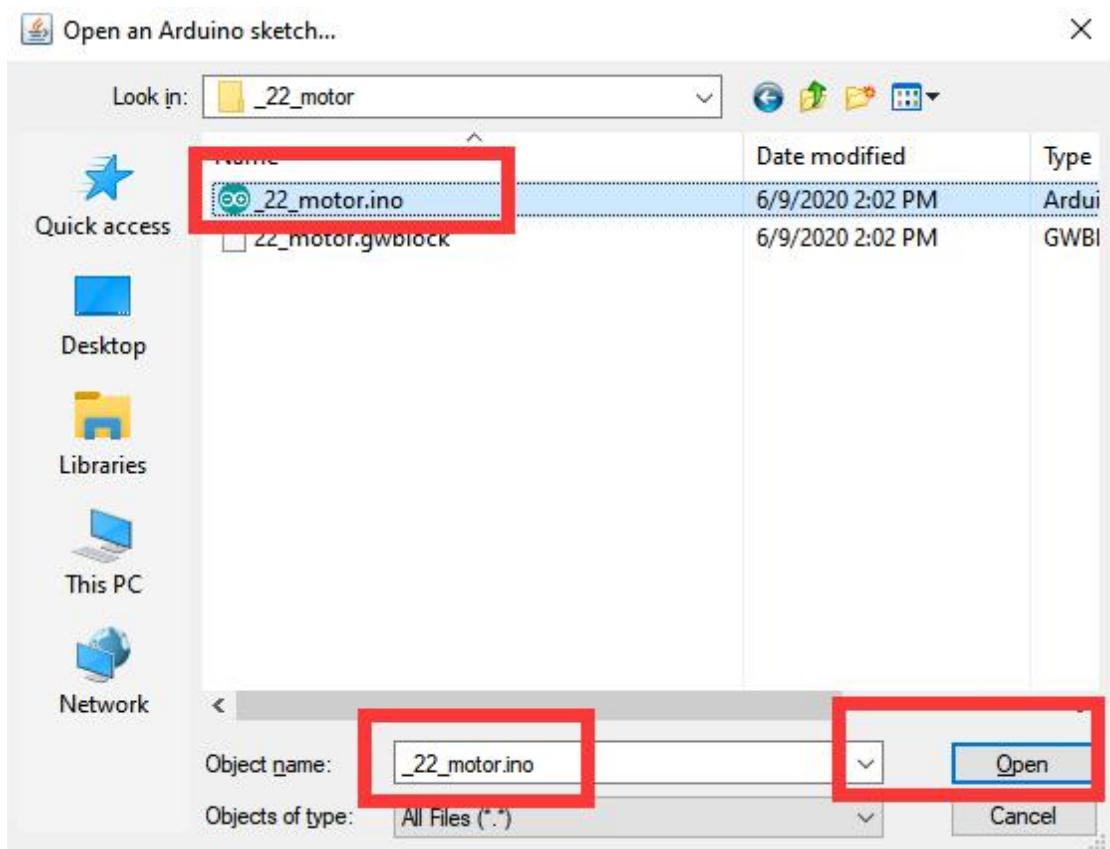
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\22_motor directory. Select _22_motor.ino. This file is the code program we need in this course. Then click Open.



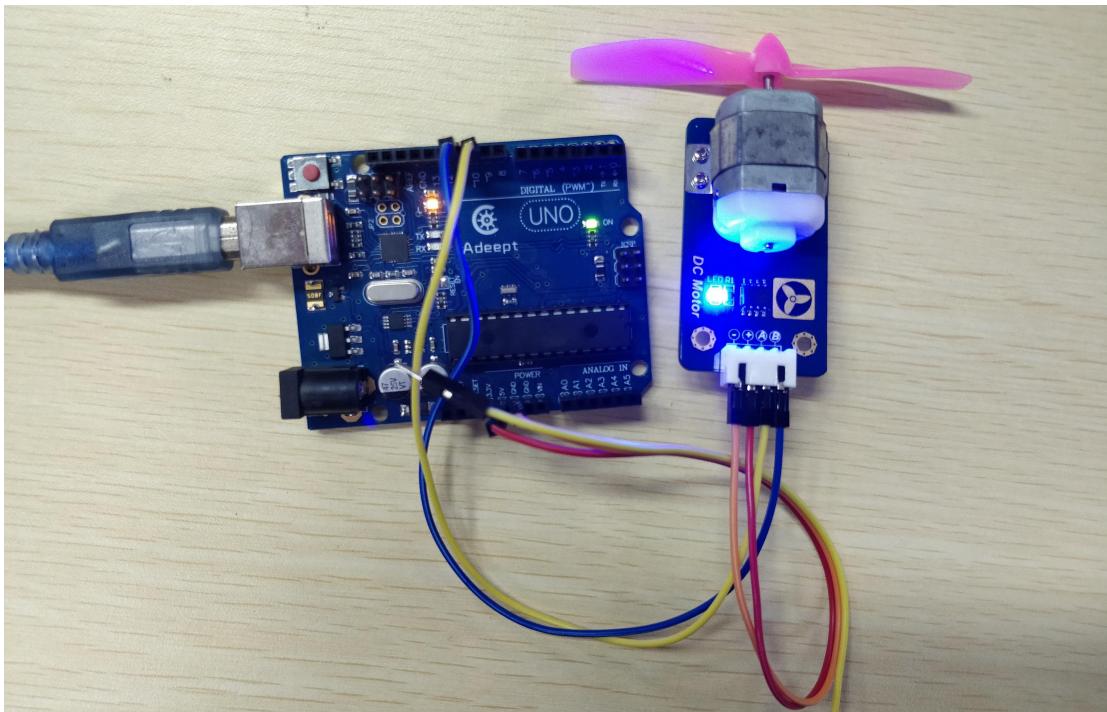
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                                         Arduino Uno on COM4
```

5. After successfully running the program, the DC motor will turn up, indicating that our course experiment is successful. The physical connection diagram of the experiment is as follows:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to control the DC motor on Arduino UNO. We will introduce how our core code can be achieved:

1. In the `setup()` function, set `motorIn1` and `motorIn2` to OUTPUT mode with `pinMode()` respectively. The `clockwise(200)` function can control the DC motor to rotate at a speed of 200.

```
void setup()
{
    pinMode(motorIn1,OUTPUT);      //initialize the motorIn1 pin as output
    pinMode(motorIn2,OUTPUT);      //initialize the motorIn2 pin as output

    clockwise(200);
}
```

2. The `clockwise()` function can control the DC motor to rotate at a specific speed.

```

void clockwise(int Speed)
{
    analogWrite(motorIn1,Speed); //set the speed of motor
    analogWrite(motorIn2,0); //stop the motorIn2 pin of motor
}

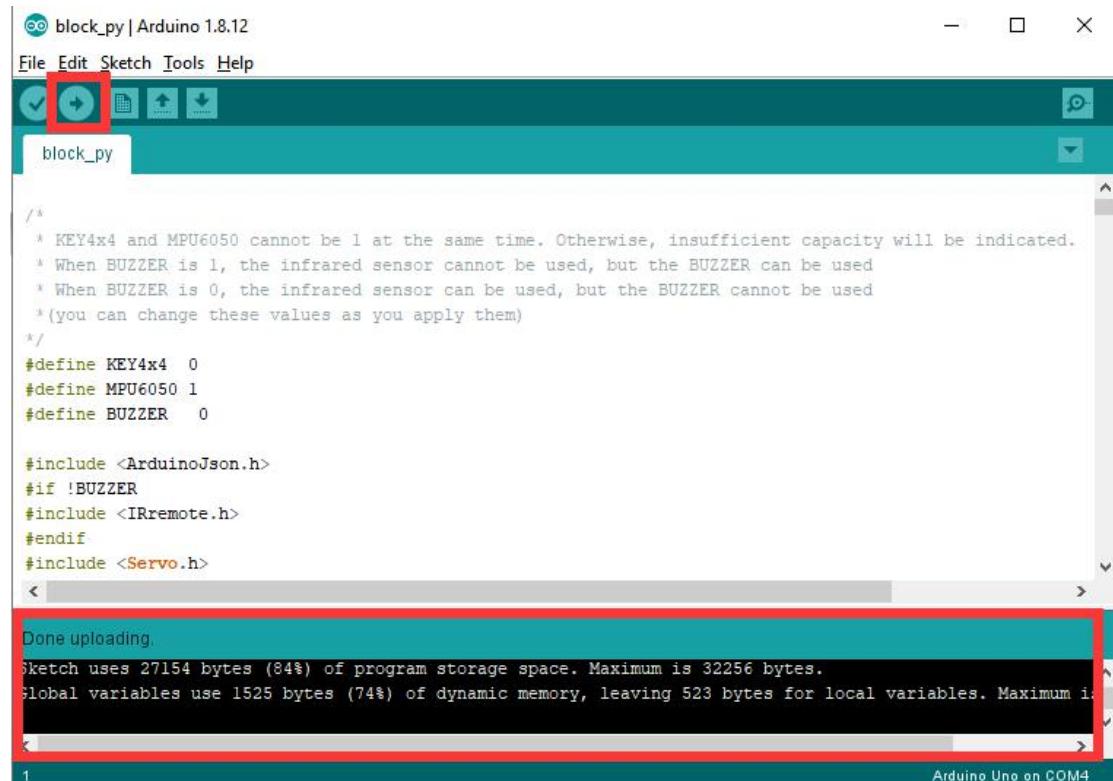
```

2.Using graphical code blocks to program to control the DC Motor on Arduino UNO

(1)Connecting to GwBlock graphical editor

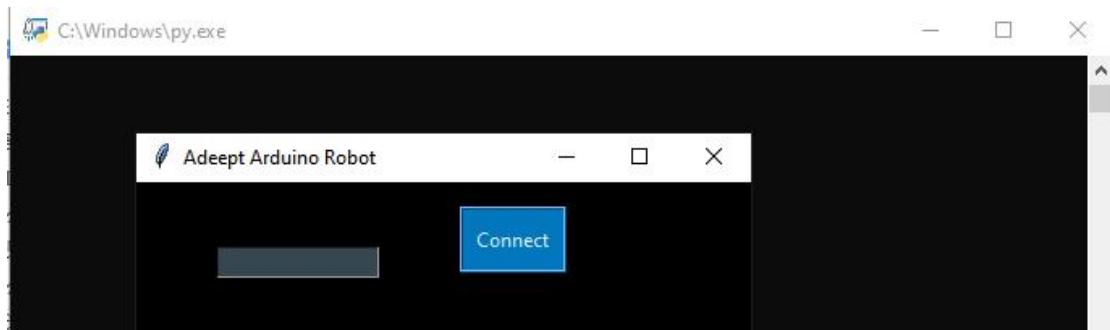
In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1.Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

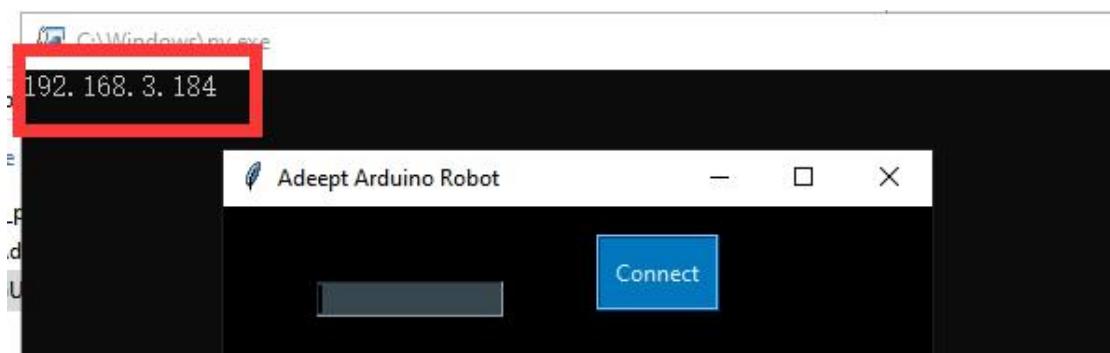


2.Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again.Then open the

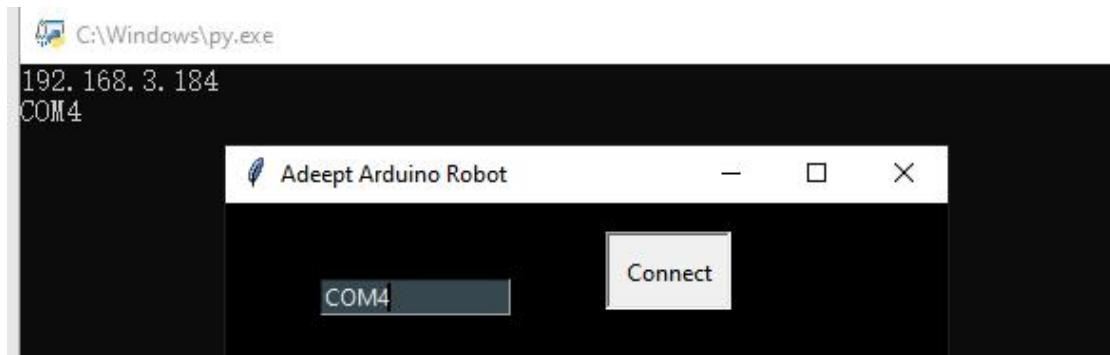
websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



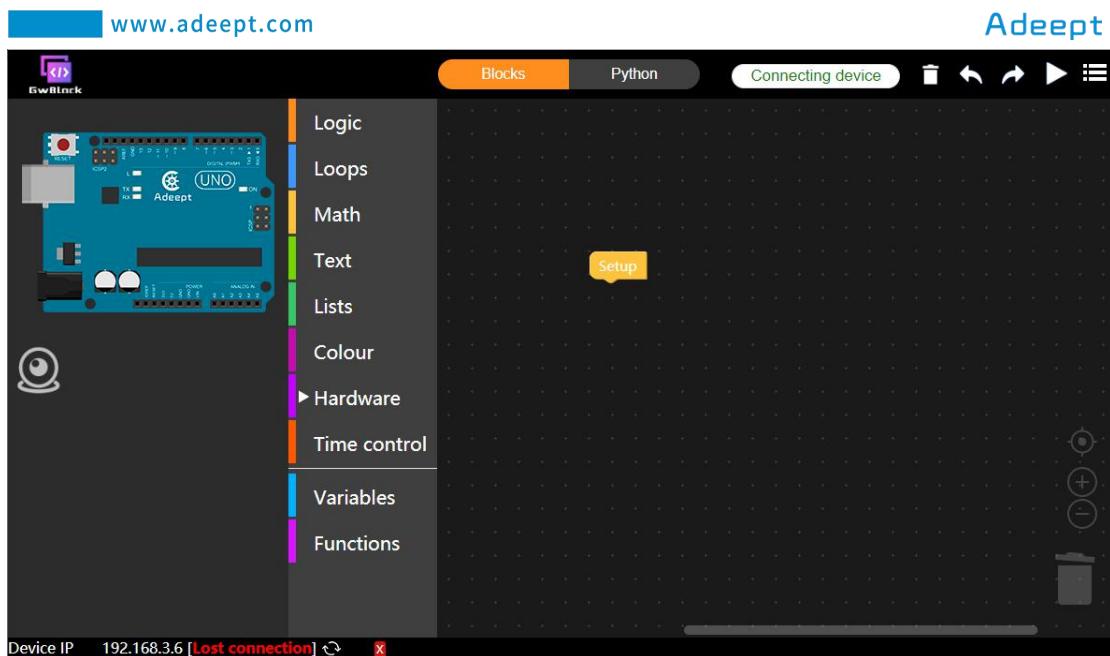
4. Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



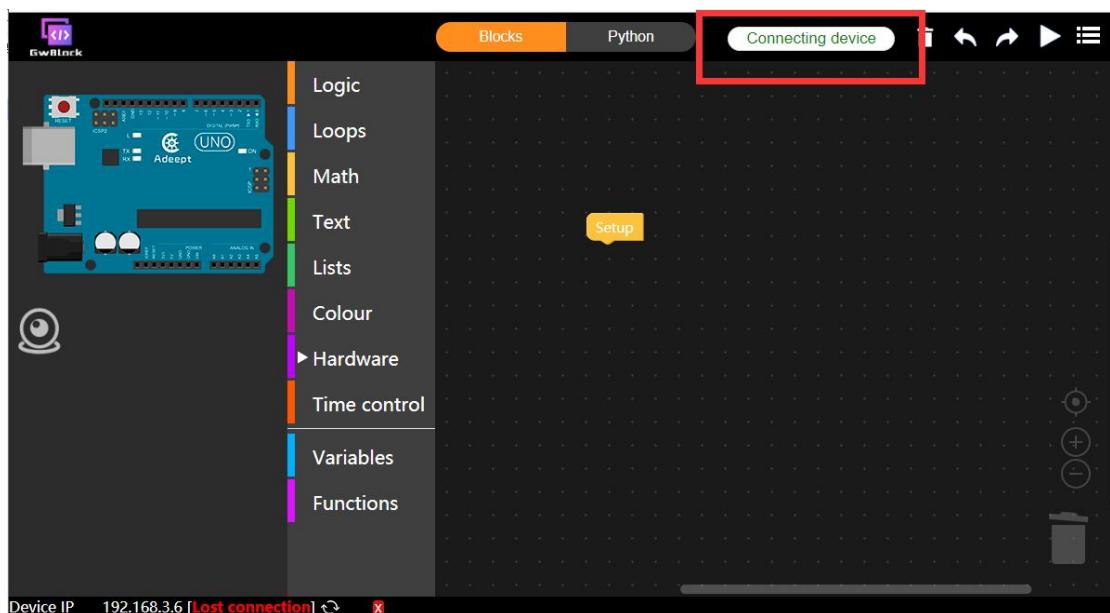
5. Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

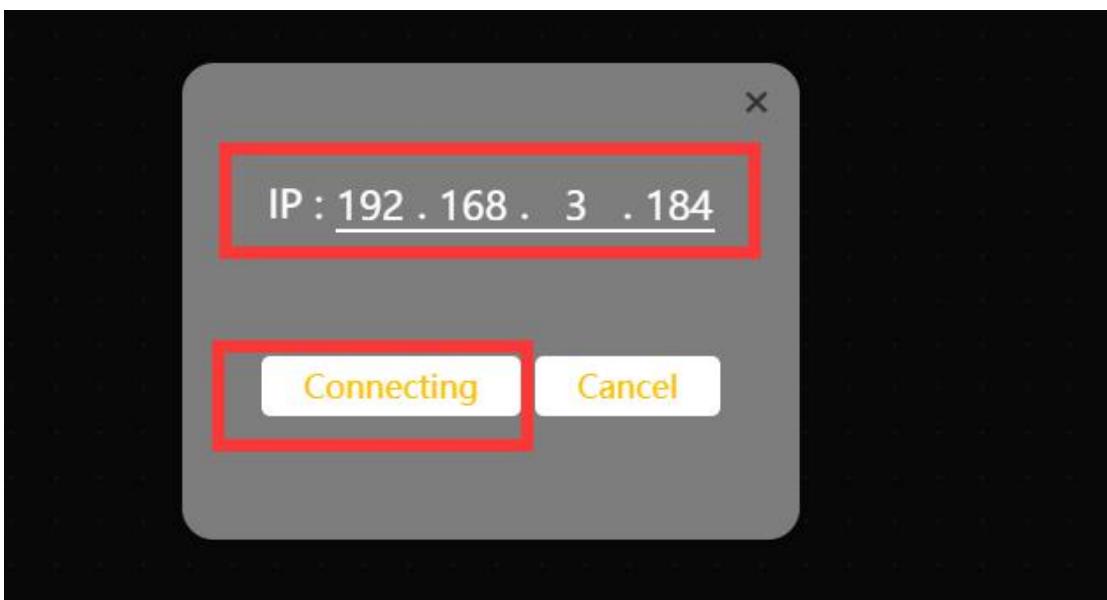
After successfully entering the website, the interface is as follows:



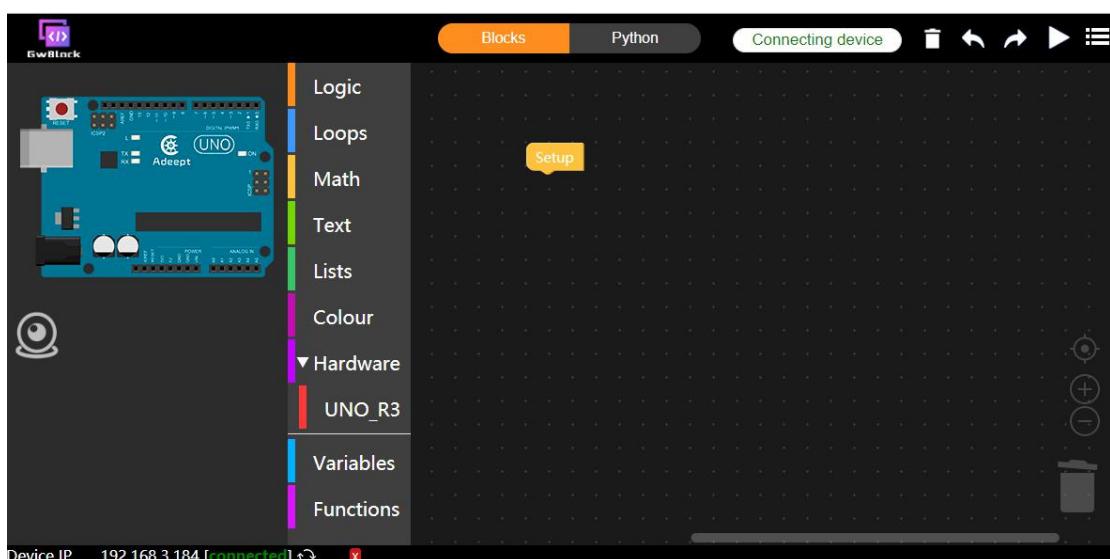
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8. After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

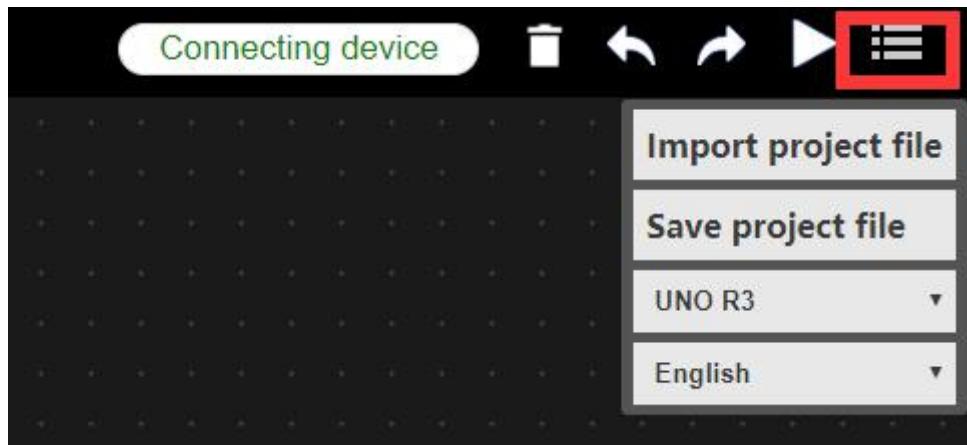


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

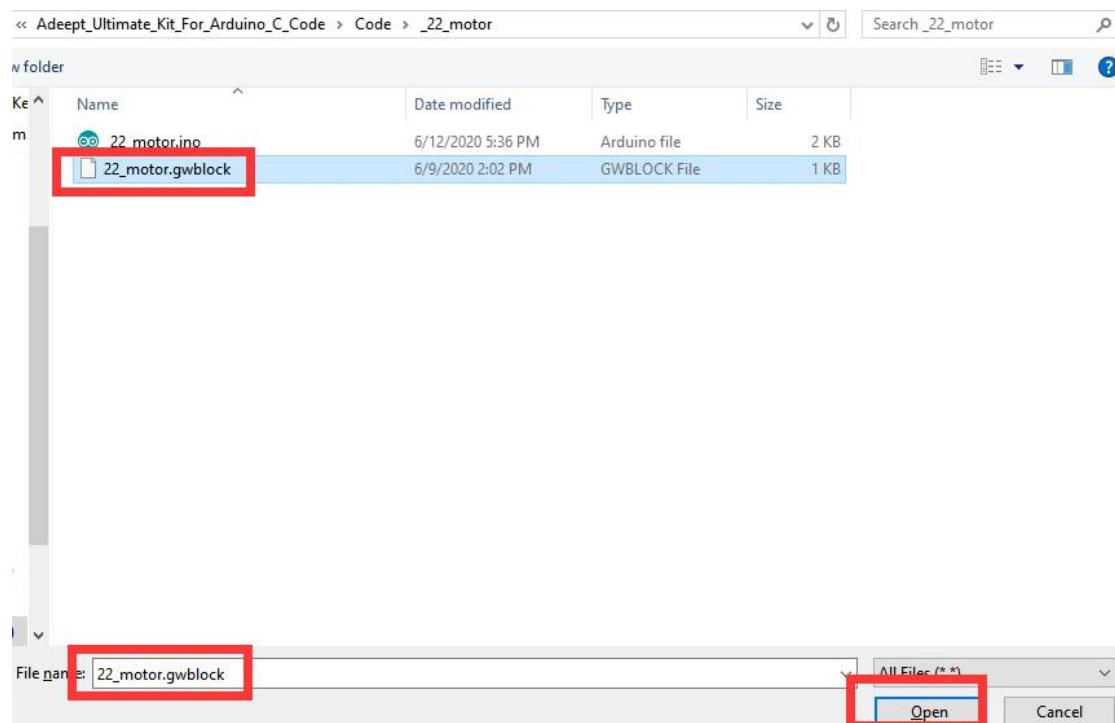
Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_22_motor

It will show as below:

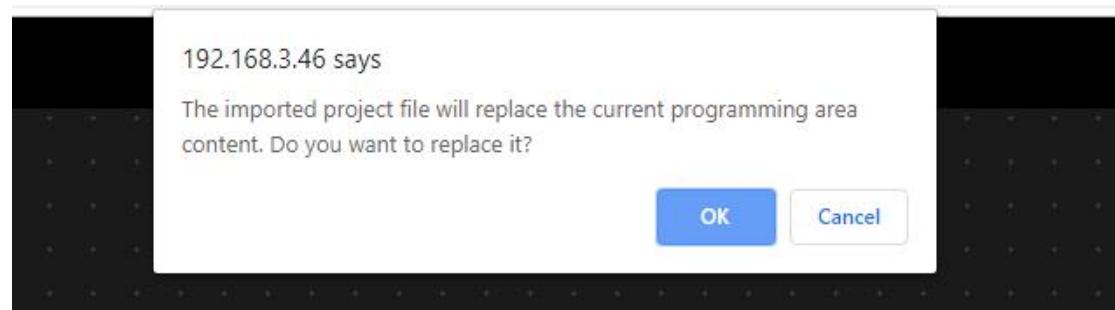
Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

5. Select the "22_motor.gwblock" file. This file is the graphical code program for our

lesson. Click "Open" in the lower right corner. It will show as below:



6. Click OK. It will show as below:

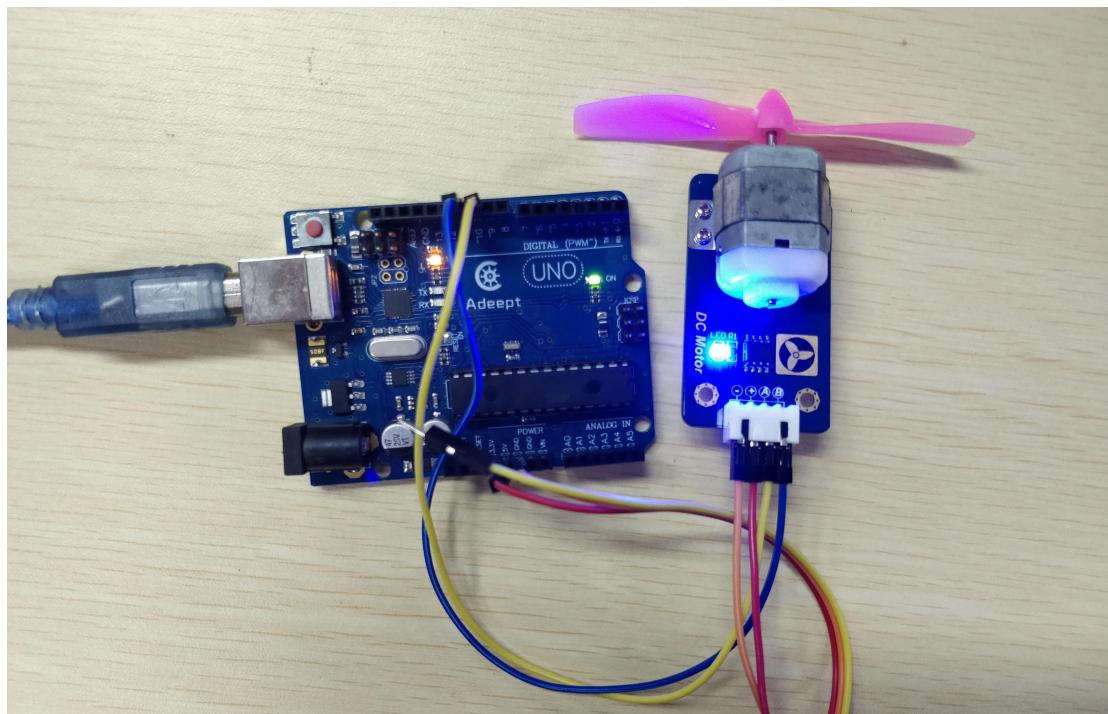


7. It will show as below after successfully opening:



8. Click the button  on the upper right corner. After successfully running the program, the DC motor will rotate, indicating that our experimental test is successful.

The physical connection diagram of the experiment is as follows:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to control the DCmotor on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

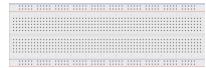
In the GwBlock graphical editor, all code programs are executed from **Setup**. Control the DC motor to rotate at a speed of 200 with the command block **Motor rotation : direction Reversal speed 200**.



Lesson 23 The Application of the PS2 Joystick Module

In this lesson, we will learn the application of PS2 Joystick Module.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
LCD1602	1	
Adeept IIC Module	1	
Male to Female Jumper Wires	Several	
Adeept PS2 Joystick Module	1	

2. The introduction of PS2 Joystick Module

(1)PS2 Joystick Module

The PS2 joystick module is an input device consisting of a joystick that can rotate on the base and report its angle or direction to the device it controls. Joysticks are commonly used to control video games and robots. We use the joystick PS2. It consists of two sliding rheostats and a button. When the rocker is turned, the resistance of the sliding rheostat will change, and the corresponding X/Y voltage

value will also change. If you press the rocker hard, the button will be triggered. When pressed, the corresponding SW signal will go low.

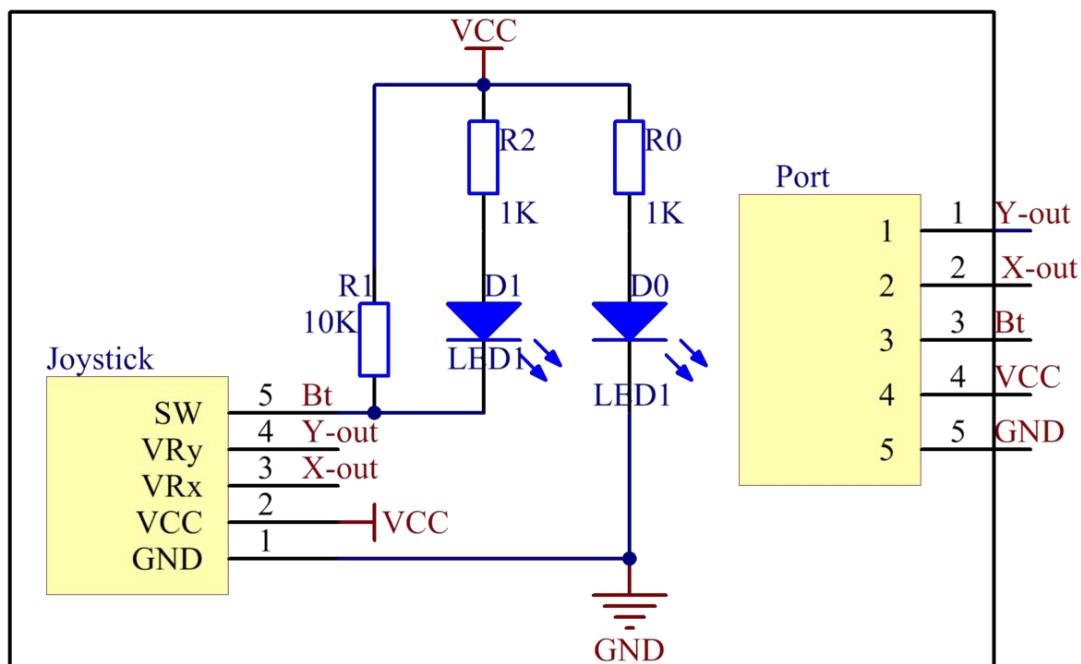


(2)Working principle of the PS2 Joystick Module

The PS2 joystick module consists of two sliding rheostats and a button. When the joystick is turned, the resistance of the sliding rheostat changes, and the corresponding X/Y voltage value also changes. If you press the joystick, the button will be pressed and the corresponding SW signal goes low. This module uses 5V power supply. In the original state, the voltage read by X and Y is about 2.5V. When pressed in the direction of the arrow, the voltage value increases with the maximum value of 5V; when pressed in the opposite direction of the arrow, the voltage value is reduced to a minimum of 0V.

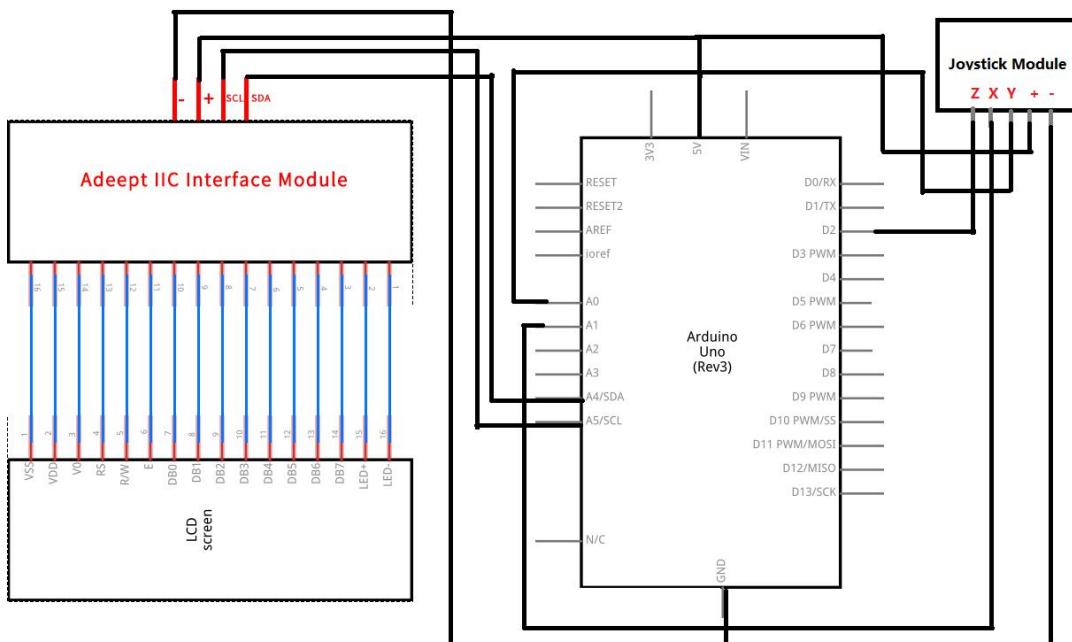
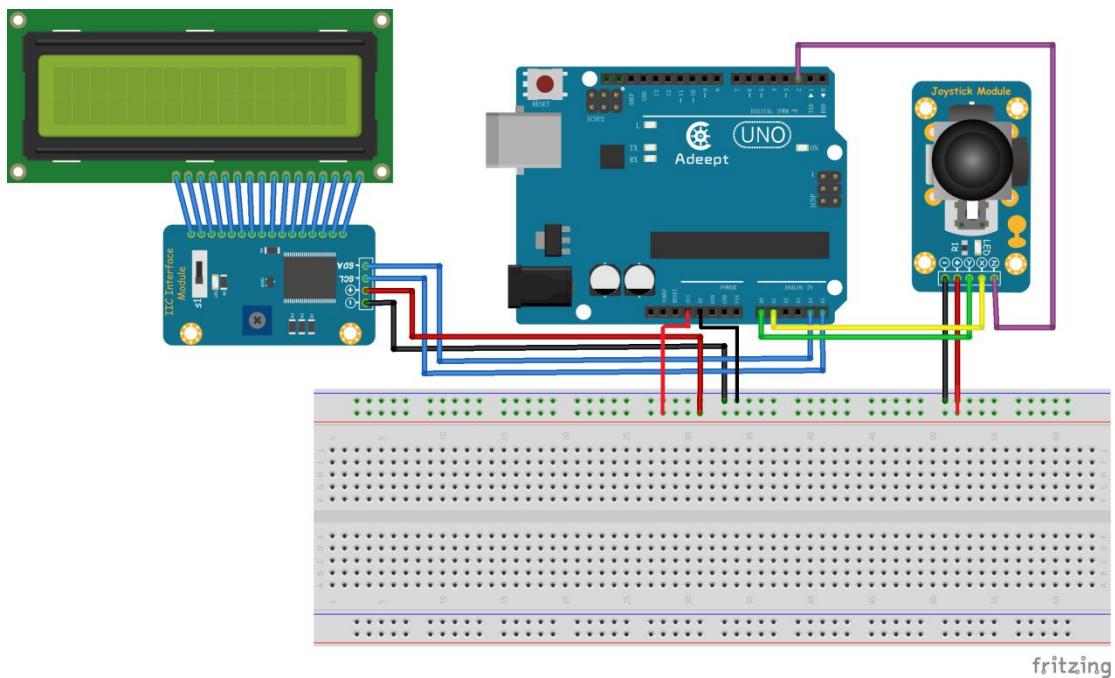
The PS2 joystick module has two analog outputs and one digital output interface. The output values correspond to the (X, Y) dual-axis offsets. The type is analog; the key indicates whether the user presses on the Z axis. It is a digital switch. The

integrated power indicator of the module can display the working status; the coordinate identifier is clear, concise and accurate positioning. In order to more easily cooperate with standard interfaces such as expansion boards, the X and Y axis circuits are led separately in the design to control the input of the joystick module's X, Y values and to achieve a certain function under a specific value.



3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:



4. Reading the value of PS2 Joystick Module

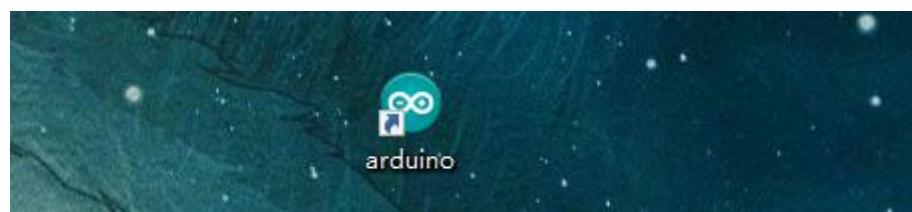
We provide two different methods to read the value of PS2 Joystick Module. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for

beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

1.Using C language to program to read the value of PS2 Joystick Module on Arduino UNO

(1)Compile and run the code program of this course

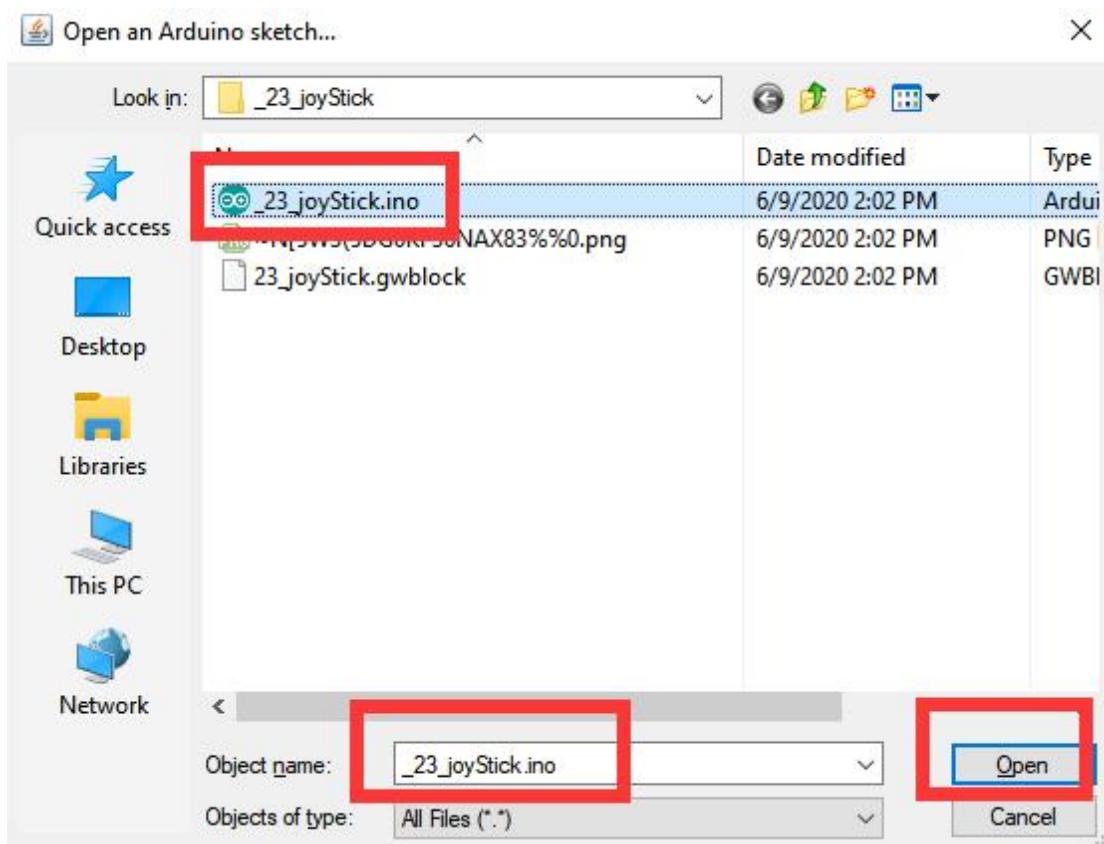
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\23_joyStick directory. Select _23_joyStick.ino. This file is the code program we need in this course. Then click Open.



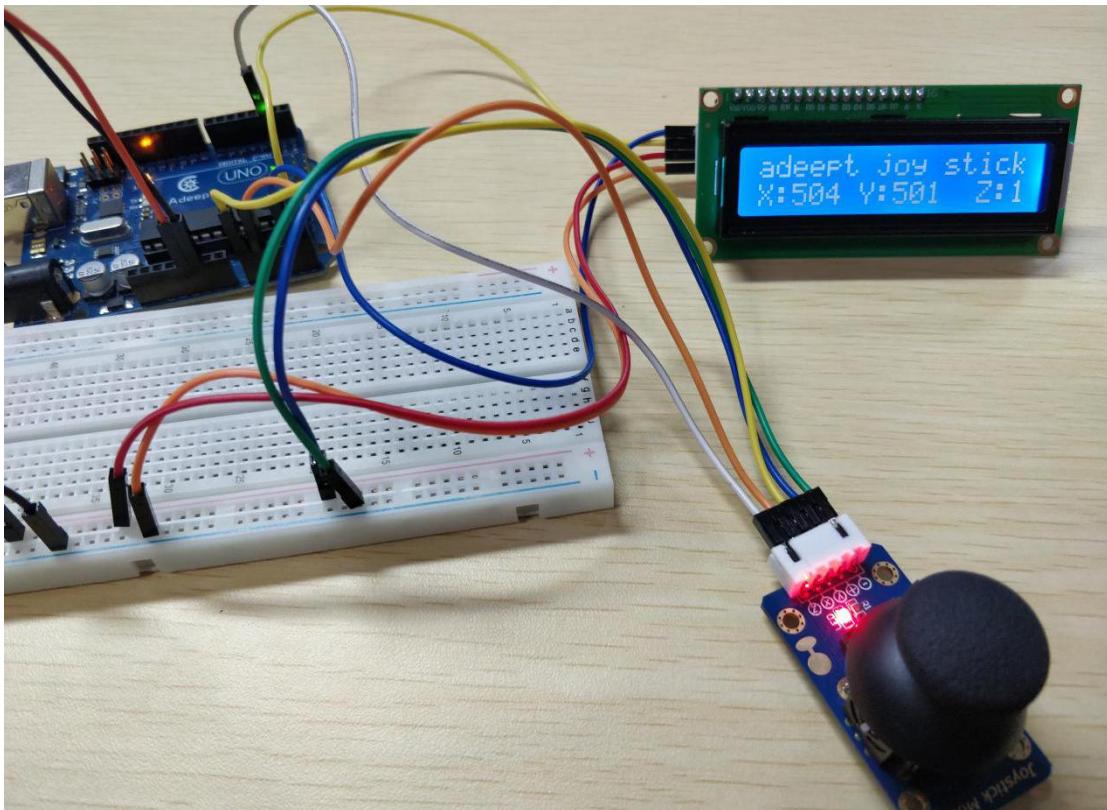
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After successfully running the program, you can toggle the PS2 joystick, and the corresponding coordinate value on the LCD1602 screen will also change. The physical connection diagram of the experiment is as follows:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to read the value of PS2 Joystick Module on Arduino UNO. We will introduce how our core code can be achieved:

1. In the setup() function, set JoyStick_Z to INPUT_PULLUP mode with pinMode(JoyStick_Z, INPUT_PULLUP).

```
void setup(void)
{
    lcd.init(); //initialize the lcd
    lcd.backlight(); //turn on the backlight
    pinMode(JoyStick_Z, INPUT_PULLUP); //Z axis is defined as an input PS2
}
```

2. In the loop() function, read the X, Y, Z axis data on the PS2 joystick with the analogRead() function. Display the X, Y, Z axis data on the LCD1602 screen with lcd.print().

```

void loop(void)
{
    int x,y,z;
    x=analogRead(JoyStick_X);
    y=analogRead(JoyStick_Y);
    z=digitalRead(JoyStick_Z);
    |

    lcd.setCursor(0, 0); // set the cursor to column 0, line 0
    lcd.print("adeept joy stick");// Print a message of "Temp: "to the LCD.

    lcd.setCursor(0, 1); // set the cursor to column 0, line 0
    lcd.print("X:");// Print a message of "Temp: "to the LCD.
    lcd.print(x);// Print a centigrade temperature to the LCD.
    lcd.print(" ");// Print a message of "Temp: "to the LCD
    lcd.setCursor(6, 1); // set the cursor to column 0, line 0
    lcd.print("Y:"); // Print the unit of the centigrade temperature to the LCD.
    lcd.print(y);// Print a centigrade temperature to the LCD
    lcd.print(" ");// Print a message of "Temp: "to the LCD
    lcd.setCursor(13, 1); // set the cursor to column 0, line 0
    lcd.print("Z:"); // Print the unit of the centigrade temperature to the LCD.
    lcd.print(z);// Print a centigrade temperature to the LCD
    delay(500);
}

```

2.Using graphical code blocks to program to read the value of PS2 Joystick Module on Arduino UNO

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

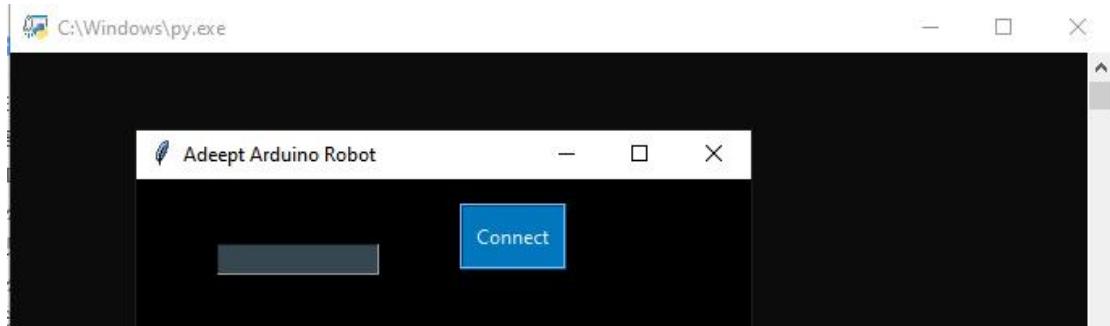
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

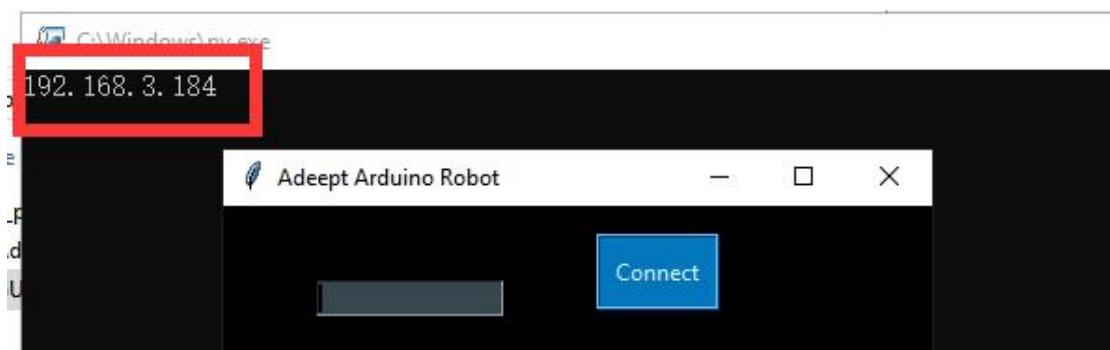
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

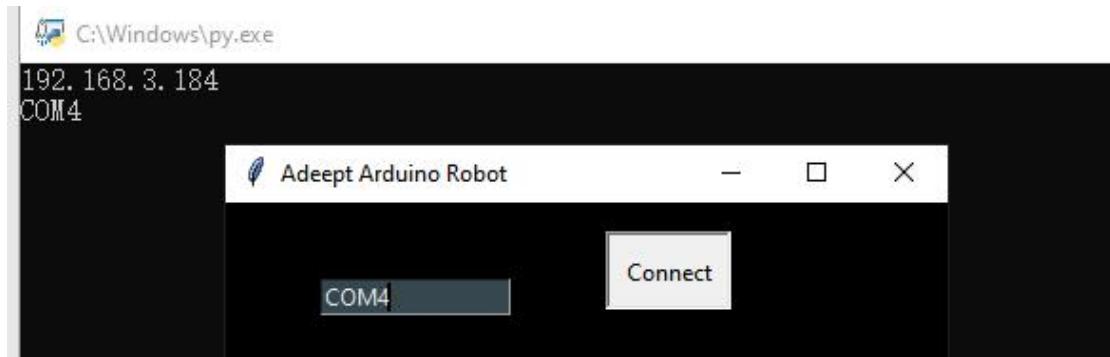
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



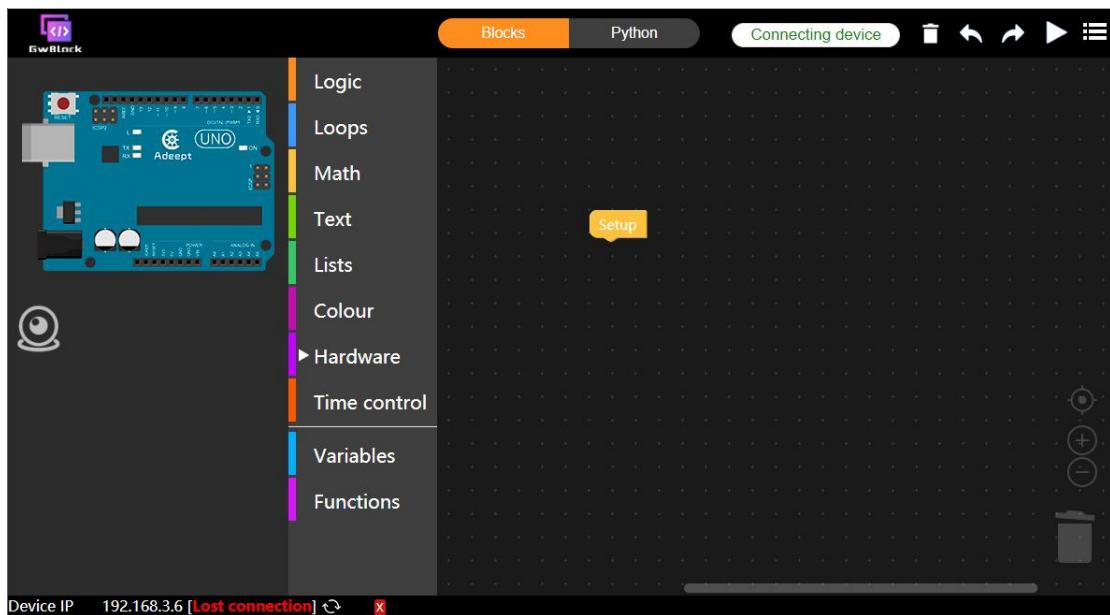
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



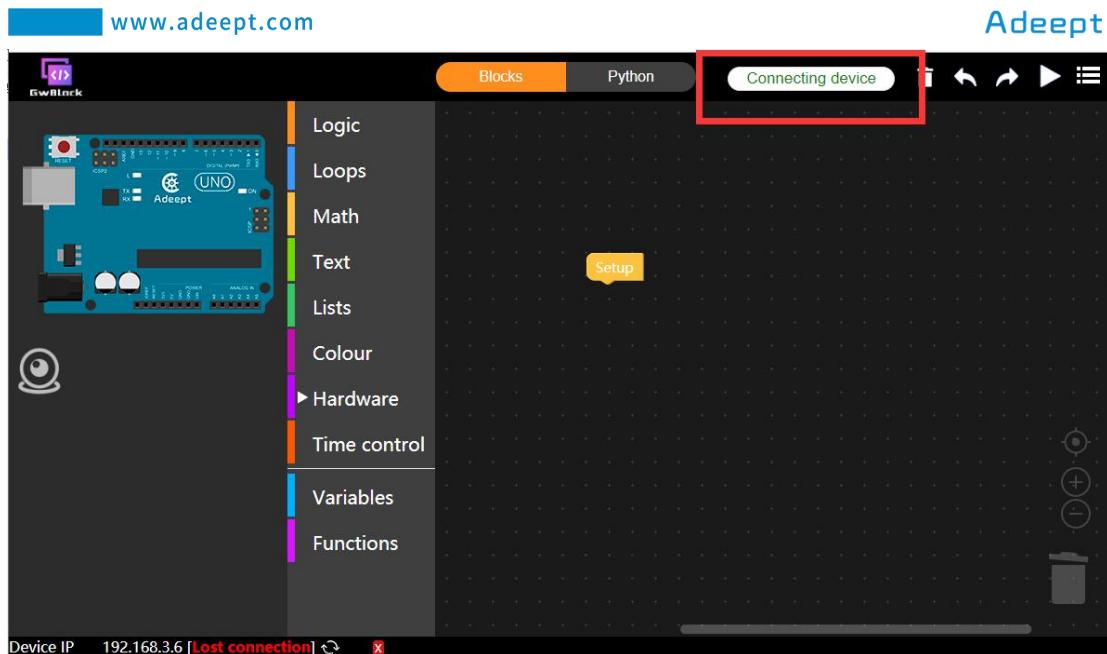
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

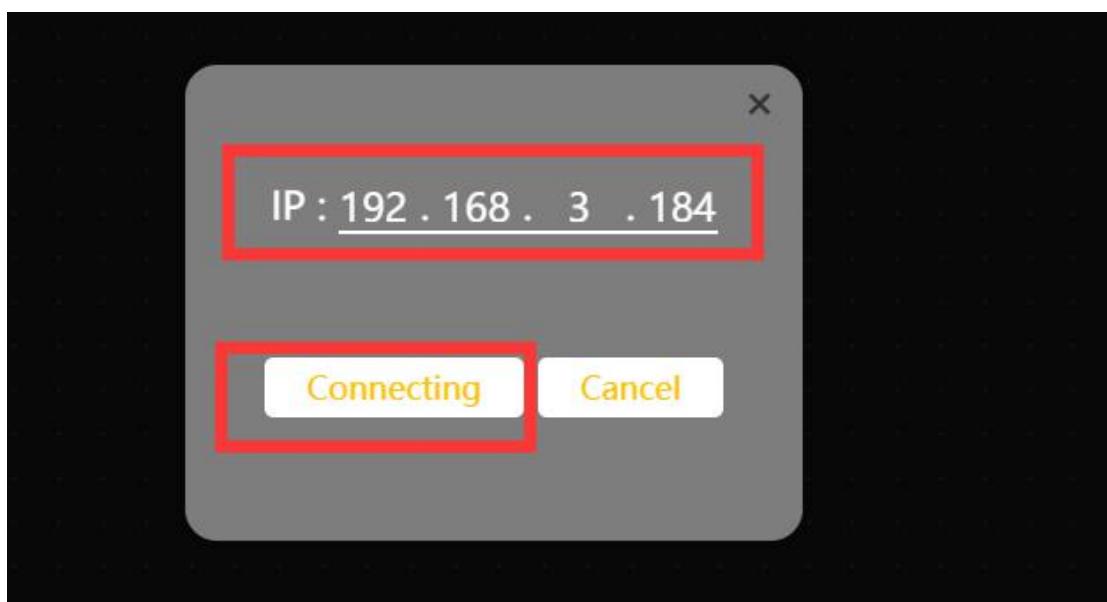
After successfully entering the website, the interface is as follows:



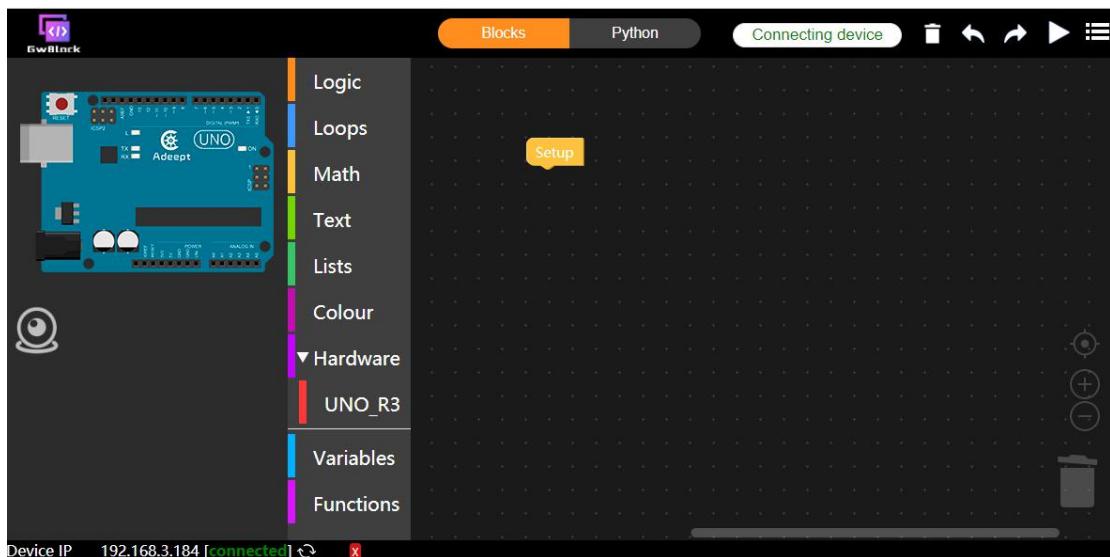
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

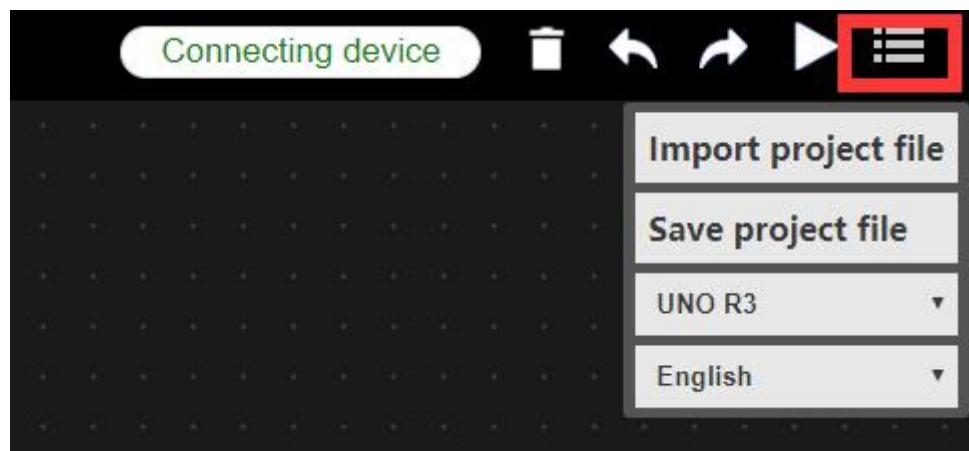


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

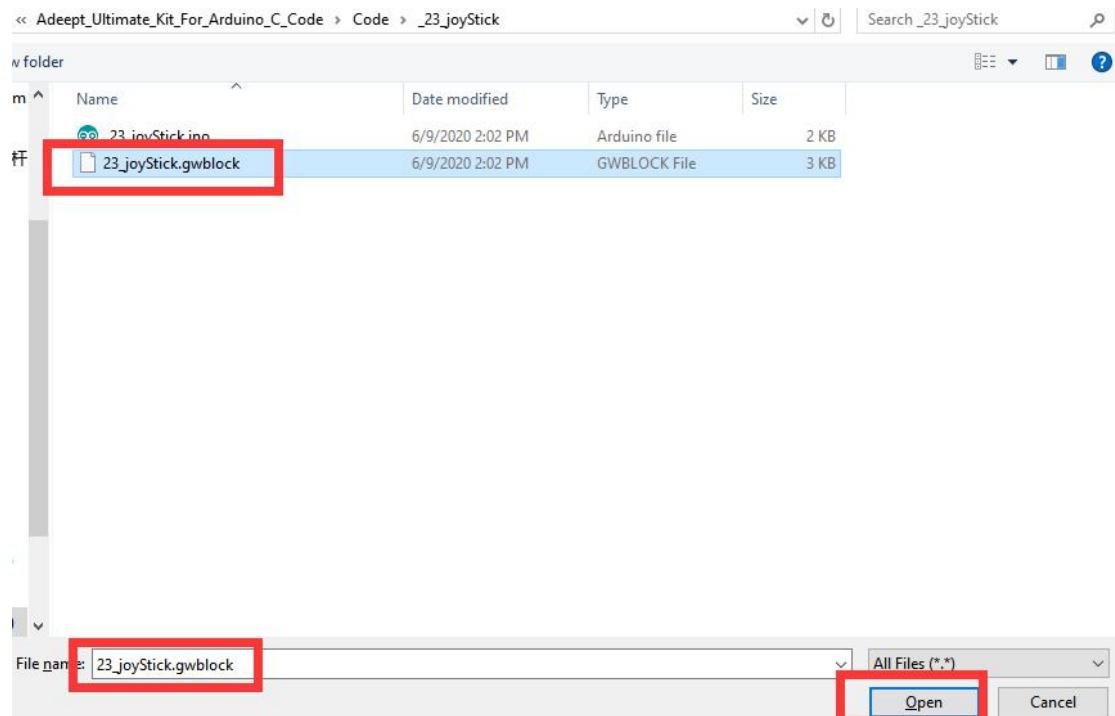
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_23_joyStick

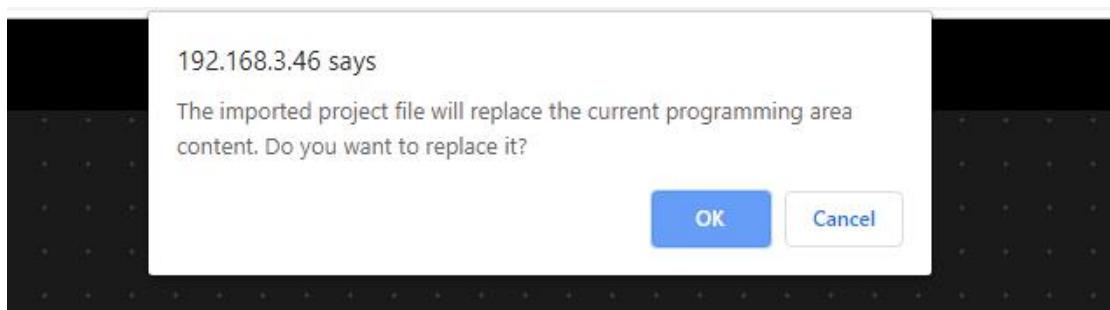
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

5. Select the "23_joyStick.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



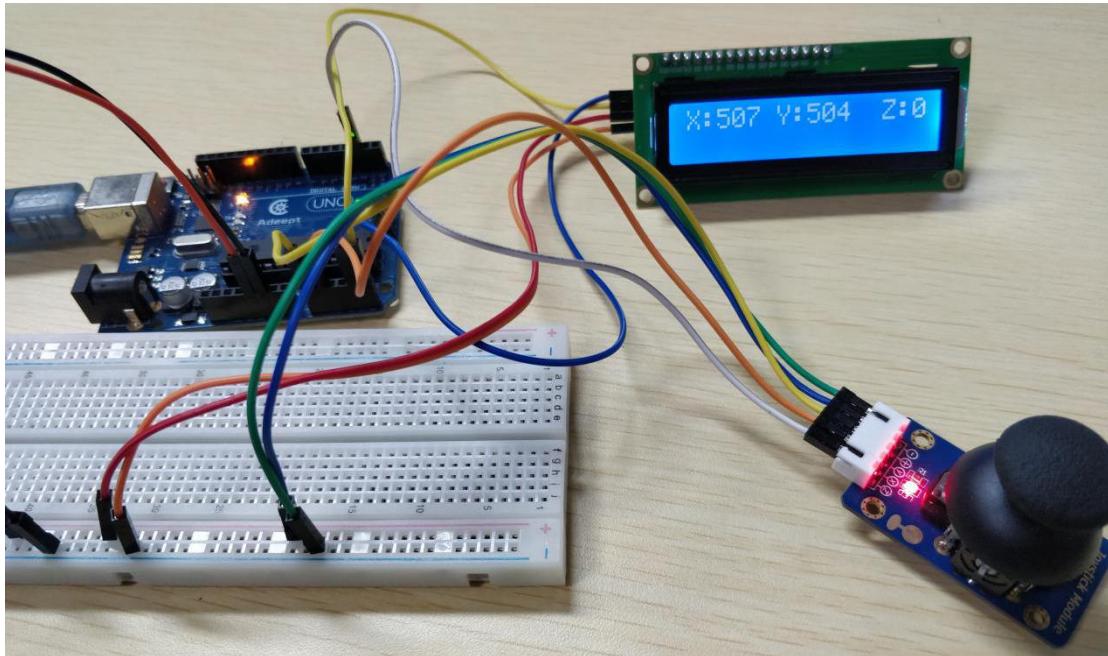
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



8. Click the button  on the upper right corner. After successfully running the program, you can toggle the PS2 joystick, and the corresponding coordinate values on the LCD1602 screen will also change. The physical connection diagram of the experiment is as follows:



(3)Core code program

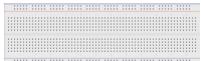
After the above hands-on operation, you must be very interested to know how we program to read the value of PS2 Joystick Module on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

In the GwBlock graphical editor, all code programs are executed from **Setup**. Use the instruction **Initialize LCD** to initialize the LCD1602. Read the X and Y axis values with the **set X to Read analog value : PIN 0** and **set Y to Read analog value : PIN 1** block. Then use the **LCD display : X** and **LCD display : Y** instructions to display the X and Y axis data.

Lesson 24 Controlling LED with Tilt Switch

In this lesson, we will learn how to use tilt switch to control LED.

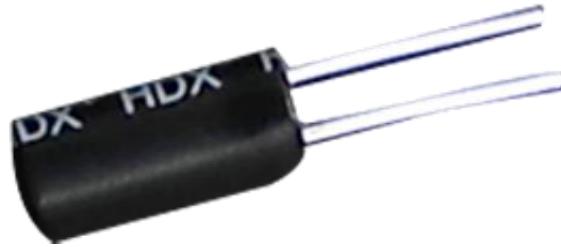
1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	1	
LED	1	
Tilt Switch	1	

2.The introduction of tilt switch

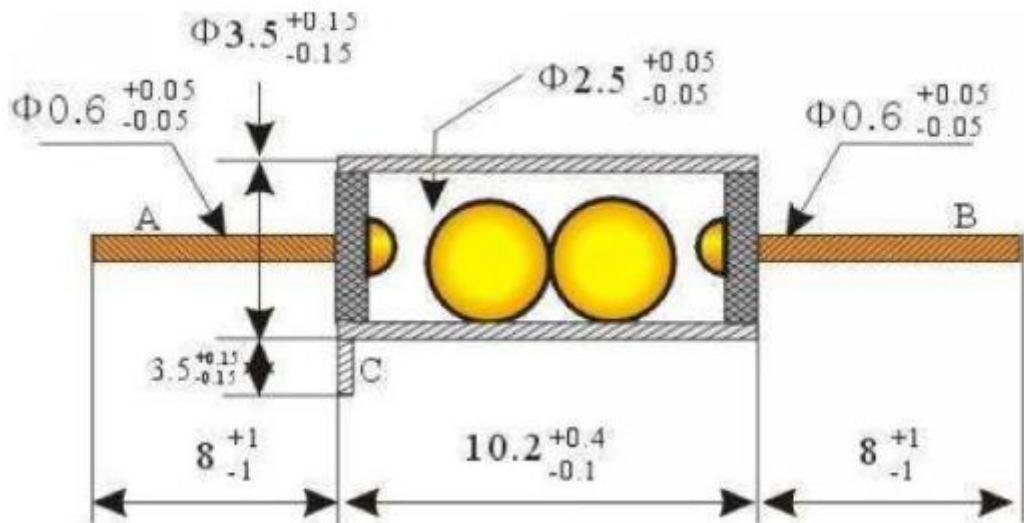
(1)tilt switch

The tilt switch is also called a ball switch, which controls the connection or connection of the circuit with the principle of the ball rolling contact with the guide pin. The tilt switch is also called the ball switch. When the switch is tilted in the appropriate direction, the contacts will be connected, tilting the switch the opposite direction causes the metallic ball to move away from that set of contacts, thus breaking that circuit.



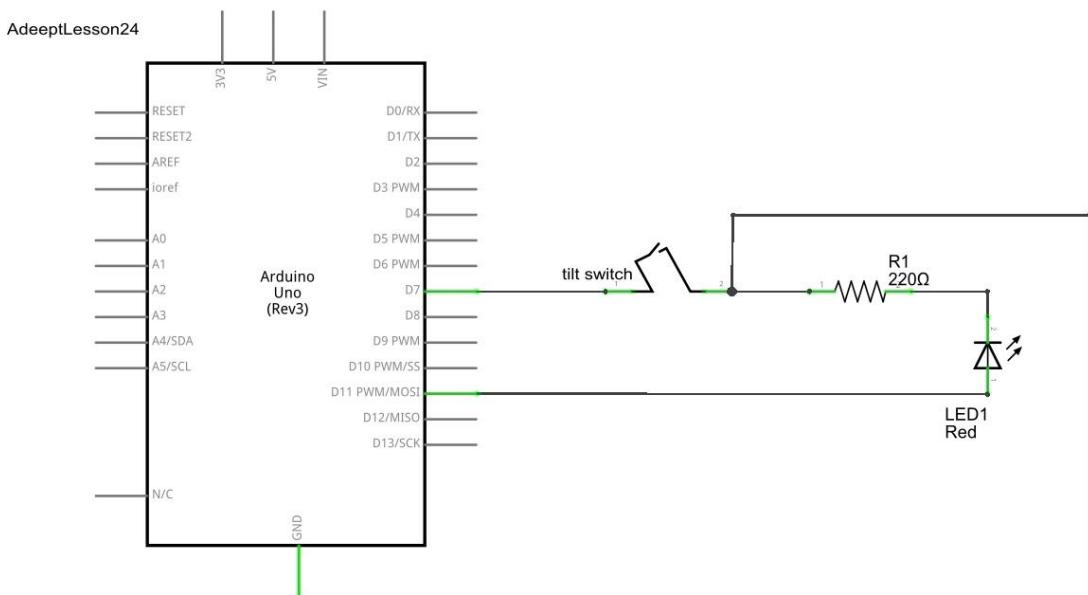
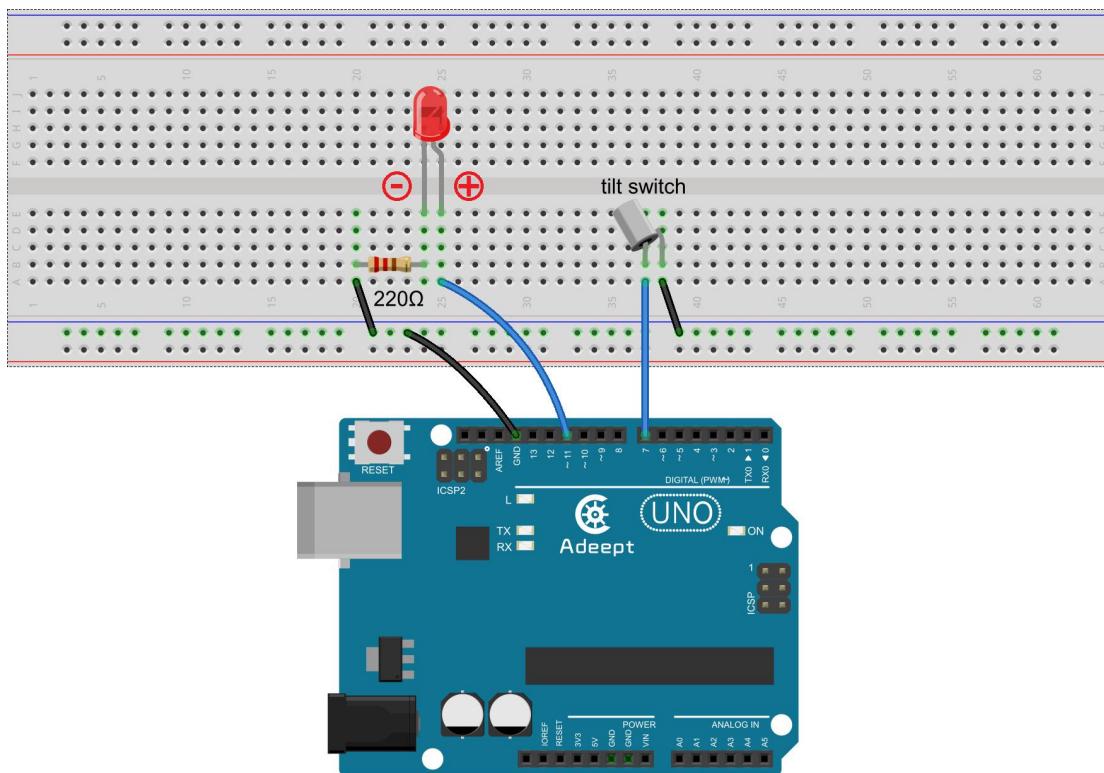
(2)Working principle of tilt switch

The tilt switch is also called a ball switch, which controls the connection or disconnection of the circuit by the principle that the ball rolls on the guide pin. The tilt switch is also called the ball switch. When the switch is tilted in the appropriate direction, the contacts will be connected, tilting the switch the opposite direction causes the metallic ball to move away from that set of contacts, thus breaking that circuit.



3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:



4. Controlling LED with tilt switch

We provide two different methods to control LED with tilt switch. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to

master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

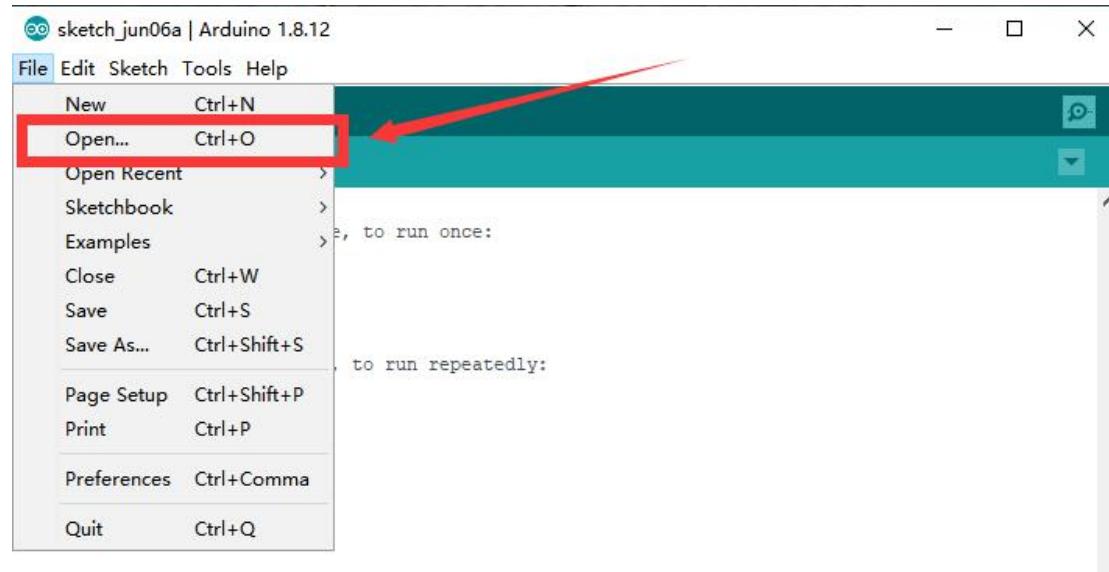
1. Using C language to program to control LED with tilt switch on Arduino UNO

(1) Compile and run the code program of this course

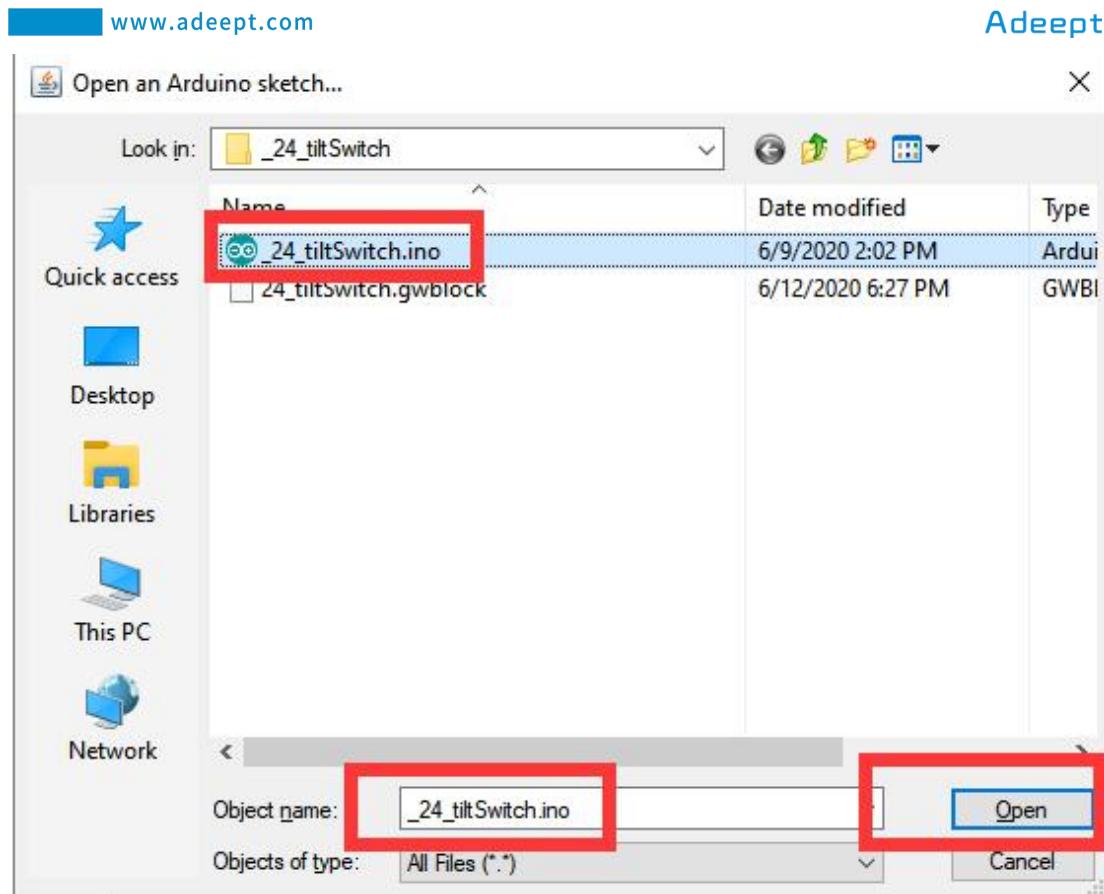
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\24_tiltSwitch directory. Select 24_tiltSwitch.ino. This file is the code program we need in this course. Then click Open.



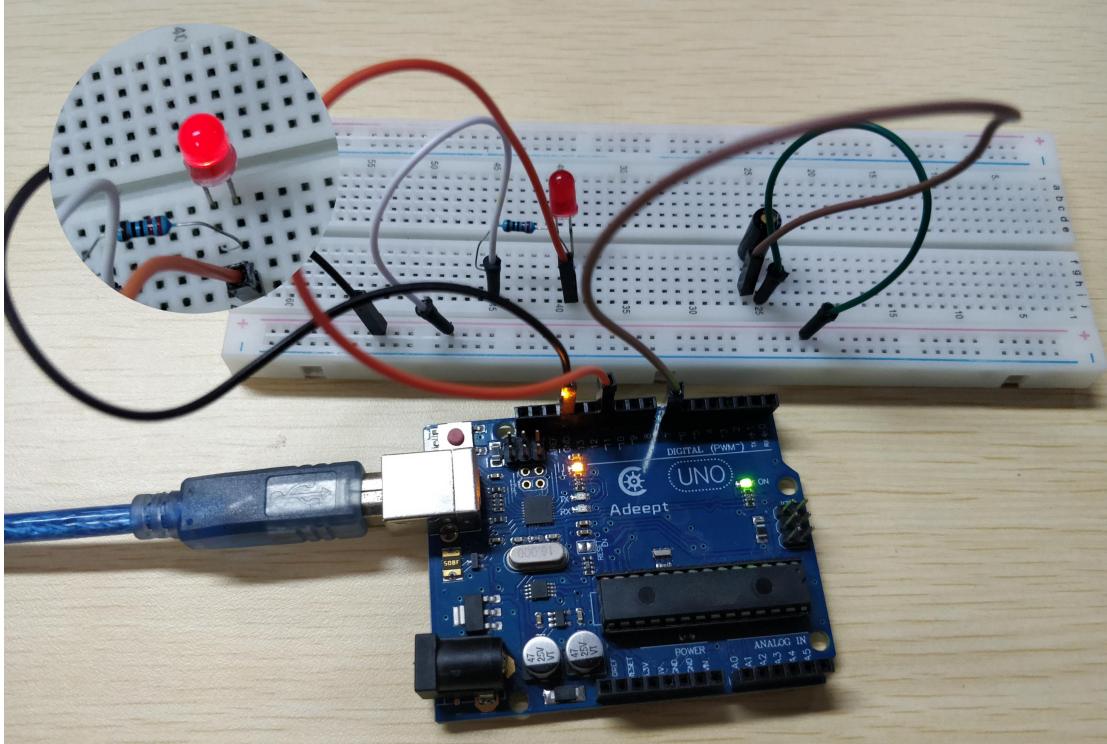
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After successfully running the program, you can tilt or tap the tilt switch with your hand, and the LED will be on. The physical connection diagram of the experiment is as follows:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to control LED with tilt switch on Arduino UNO. We will introduce how our core code can be achieved:

1.In the setup() function, set ledpin to OUTPUT mode with pinMode(ledpin, OUTPUT); pinMode(tiltSwitchpin, INPUT_PULLUP) sets tiltSwitchpin to INPUT_PULLUP mode.

```
void setup()
{
    pinMode(ledpin,OUTPUT);      //Define small lights interface for the output interface
    pinMode(tiltSwitchpin,INPUT_PULLUP); //define the tilt switch interface for input interface
}
```

2.In the loop() function, determine if the tilt switch is turned off by if(val==LOW). If the tilt switch is turned off, then we turn off the LED with digitalWrite(ledpin, LOW); if the tilt switch is closed, then we use digitalWrite(ledpin, HIGH) to turn on the LED.

```
void loop()
{
    val=digitalRead(tiltSwitchpin); //Read the number seven level value is assigned to val
    if(val==LOW)                  //Detect tilt switch is disconnected, the tilt switch when small lights go out
    { digitalWrite(ledpin,LOW); }   //Output low, LED OFF
    else                          //Detection of tilt switch is conduction, tilt the little lights up when the switch conduction
    { digitalWrite(ledpin,HIGH); } //output high, LED ON
}
```

2.Using graphical code blocks to program to control LED with tilt switch on Arduino UNO

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1.Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:



```

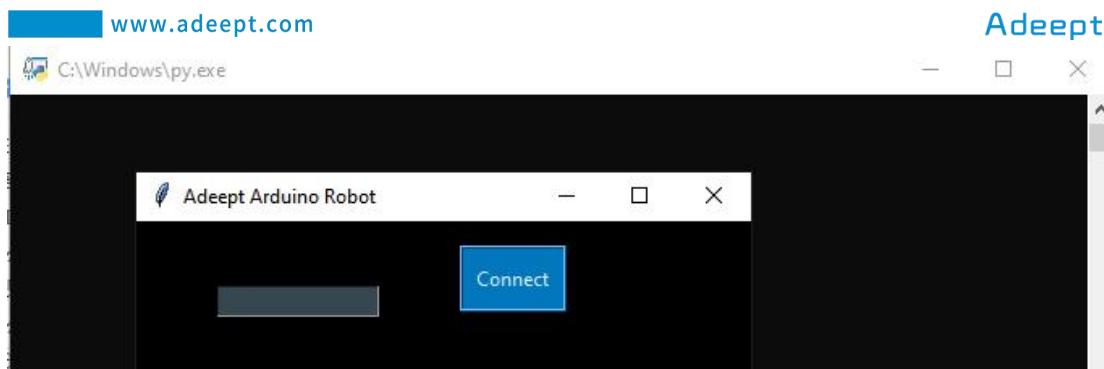
/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<

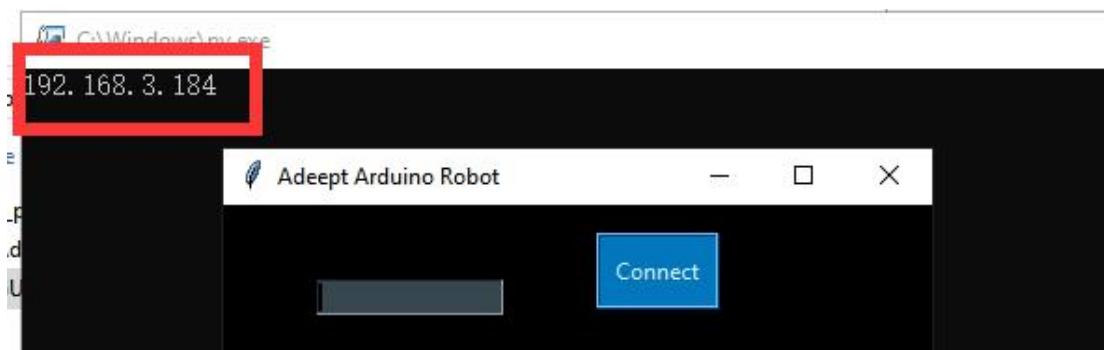
Done uploading.
Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum i
<
1
Arduino Uno on COM4

```

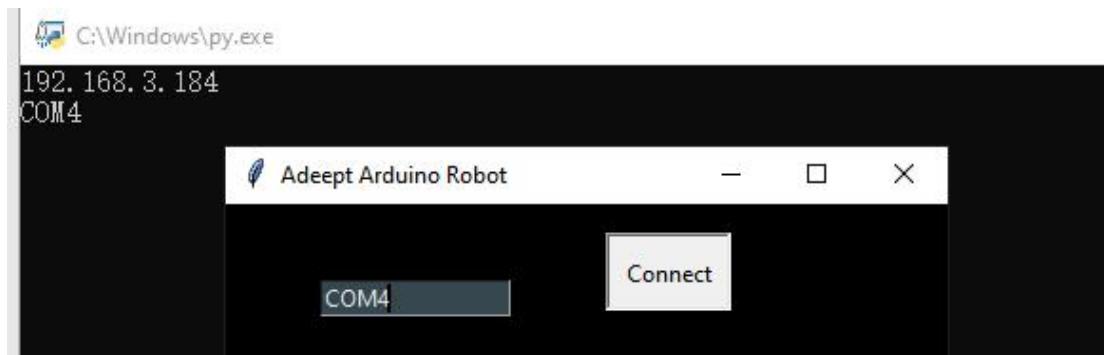
2.Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again.Then open the websocket folder. Double-click to open the GUI info v1.0.py file.It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



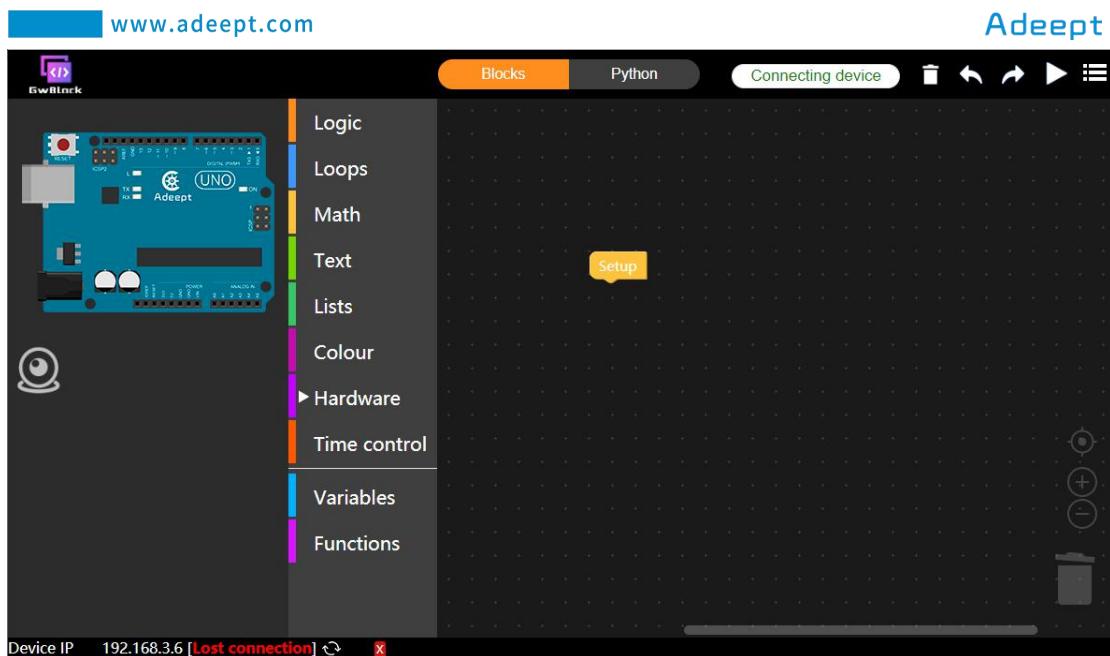
4. Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



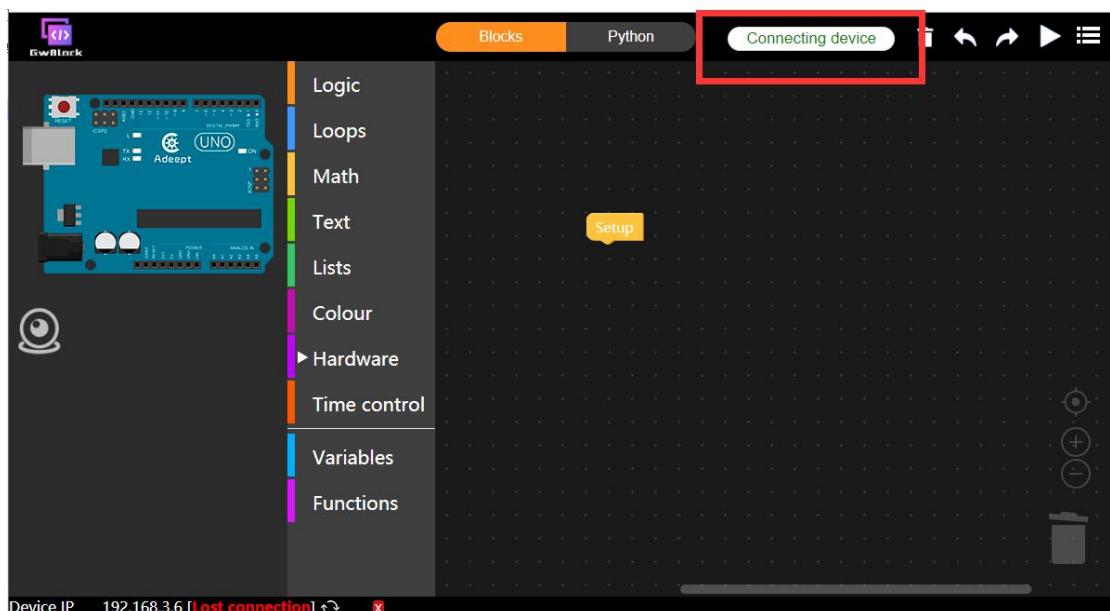
5. Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

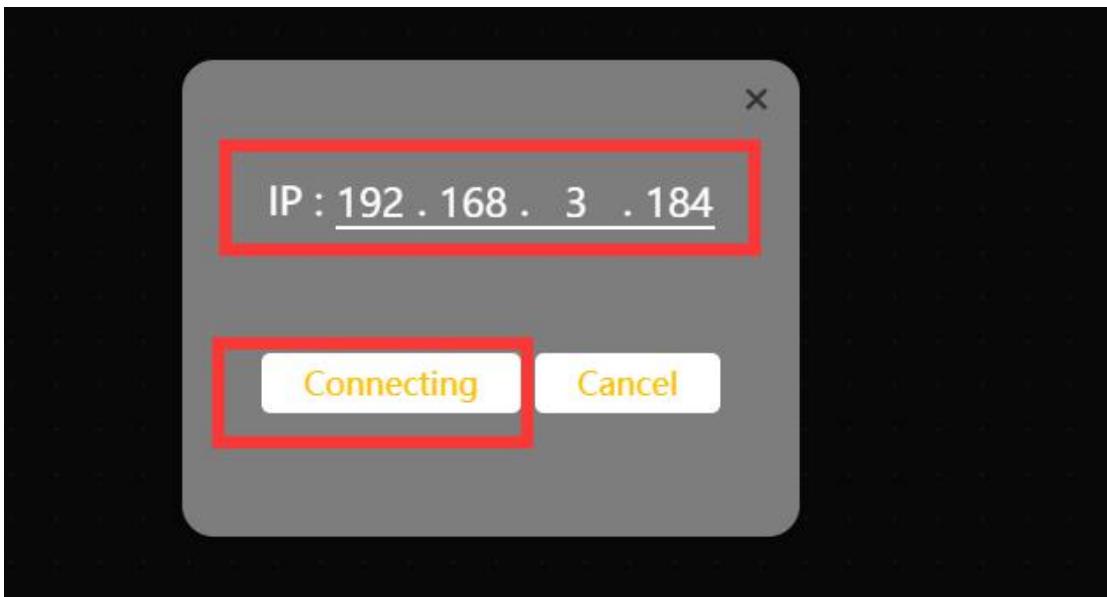
After successfully entering the website, the interface is as follows:



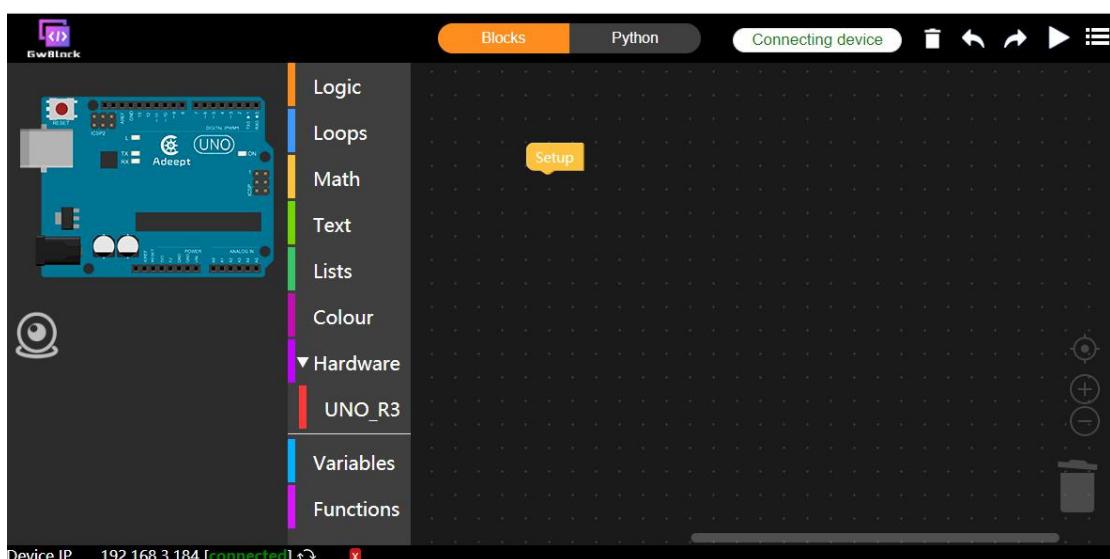
6. Click the "Connecting device" button in the upper right corner. It will show as below:



7. In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184. Then click the Connecting . It will show as below:



8. After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:



(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

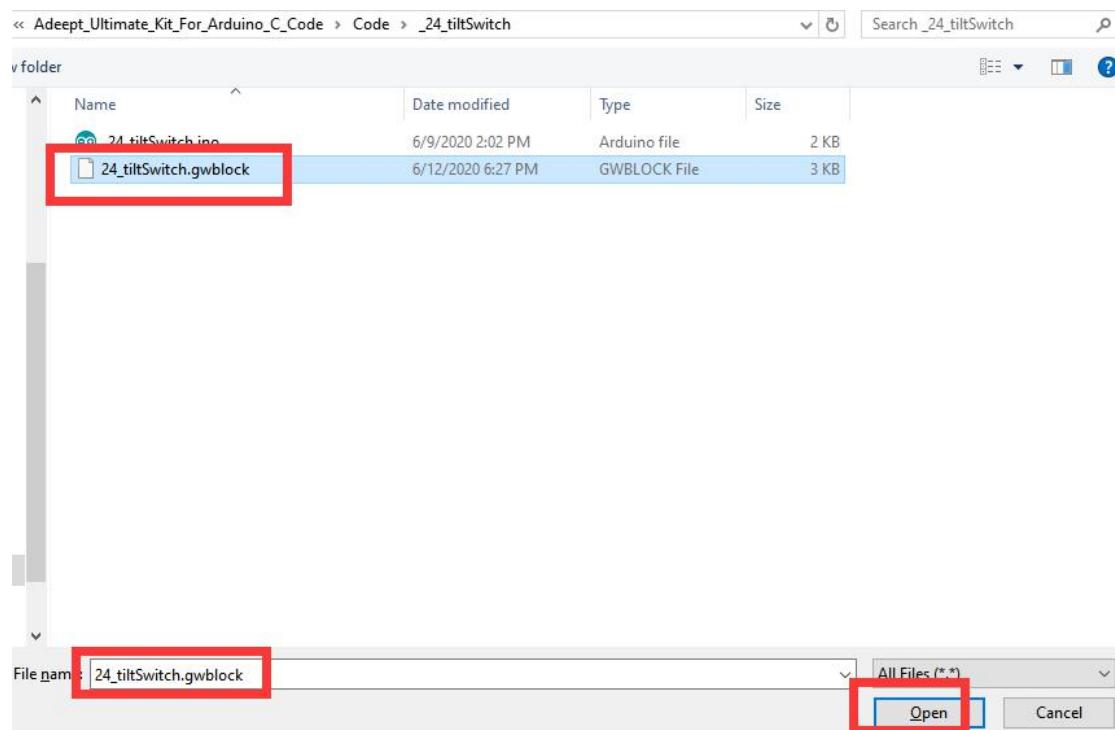
Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_24_tiltSwitch

It will show as below:

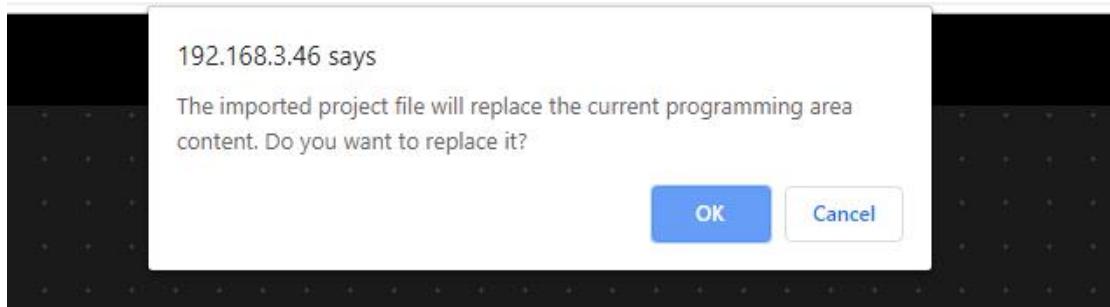
Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

5. Select the "24_tiltSwitch.gwblock" file. This file is the graphical code program for

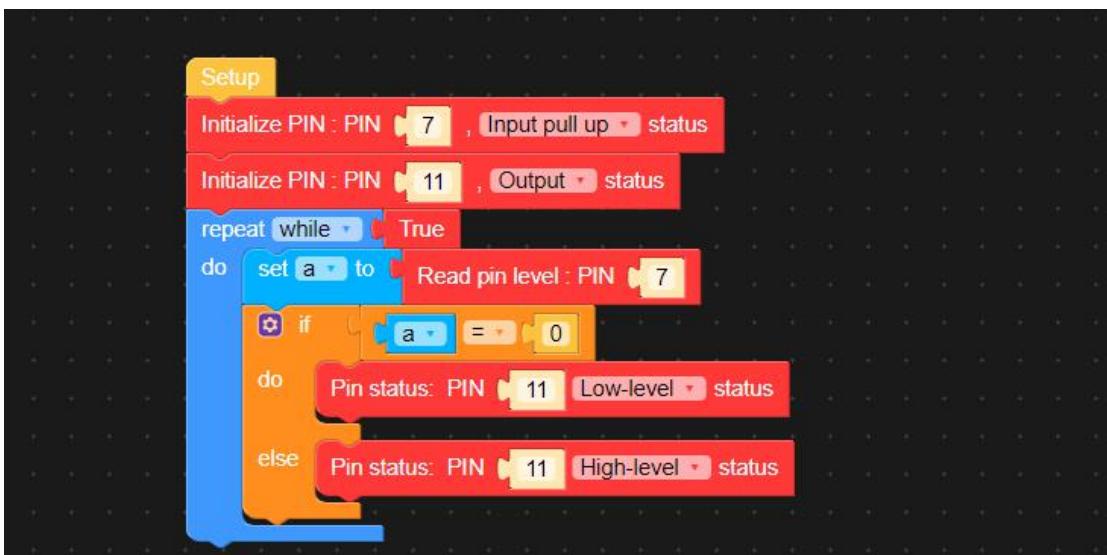
our lesson. Click "Open" in the lower right corner. It will show as below:



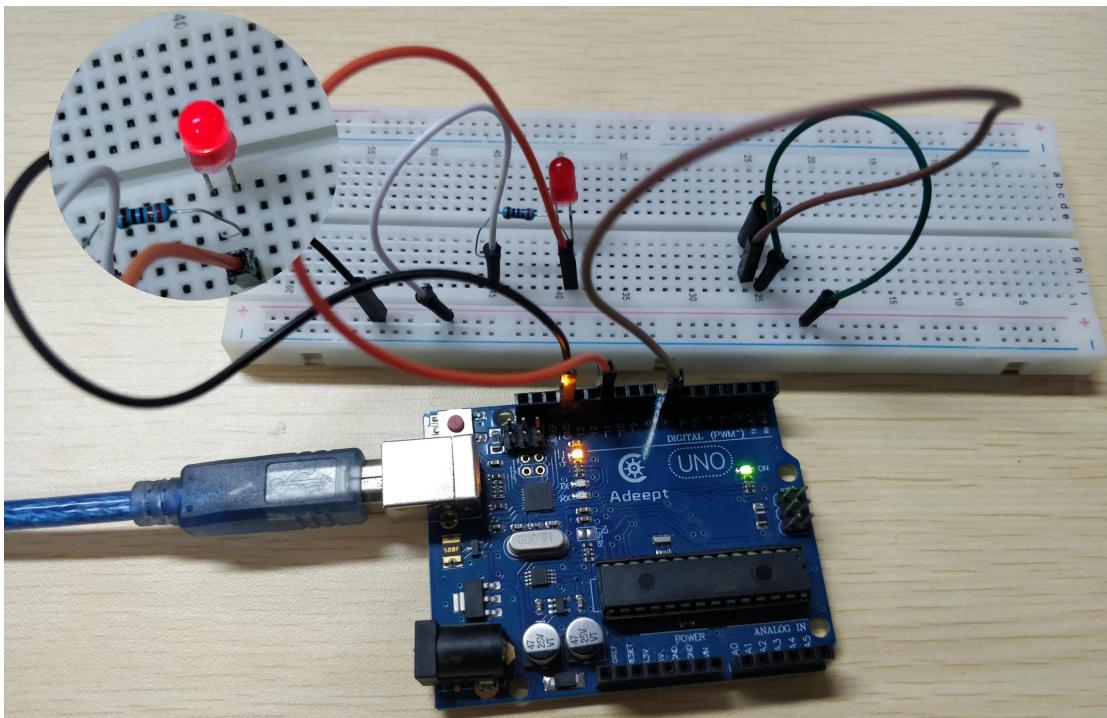
6. Click OK. It will show as below:



7. It will show as below after successfully opening:



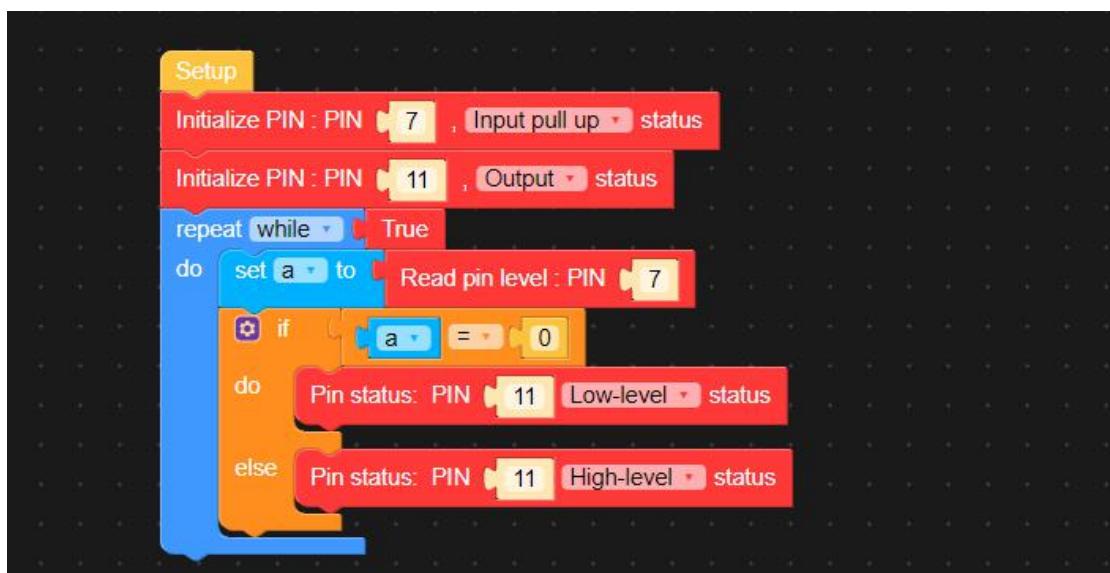
8. Click the button  on the upper right corner. After successfully running the program, you can tilt or tap the tilt switch with your hand, and the LED will be on. The physical connection diagram of the experiment is as follows:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to control LED with tilt switch on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

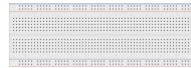
In the GwBlock graphical editor, all code programs are executed from **Setup**. Read the state value of the tilt switch with the instruction block **set a to [Read pin level : PIN 7]**. The instruction block **if [a = 0]** will judge the state value of a. When the state value equal to 0, it means that the tilt switch is in the off state, and then turn off the LED with the instruction block **Pin status: PIN 11 Low-level status**; when the state value of a is not equal to 0, it means that the tilt switch is closed, and the LED is turned on with the command block **Pin status: PIN 11 High-level status**.



Lesson 25 The Application of the 8*8 Dot-matrix Module

In this lesson, we will learn the application of the 8*8 Dot-matrix Module.

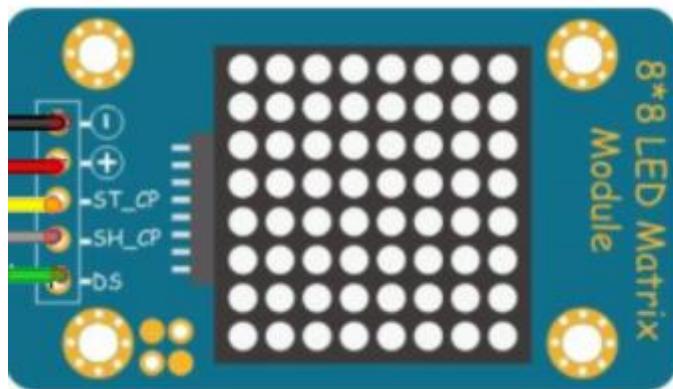
1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
LCD1602	1	
Adeept IIC Module	1	
Male to Female Jumper Wires	Several	
Adeept Dot-matrix Display Module	1	

2. The introduction of the 8*8 Dot-matrix Module

(1)The 8*8 Dot-matrix Module

Our course uses an 8x8 dot matrix display module, which is composed of 64 light-emitting diodes, and each light-emitting diode is placed at the intersection of row and column lines. Using the progressive scan method, when the scanning speed is accelerated, for the persistence of vision, we will see the 8-line display.

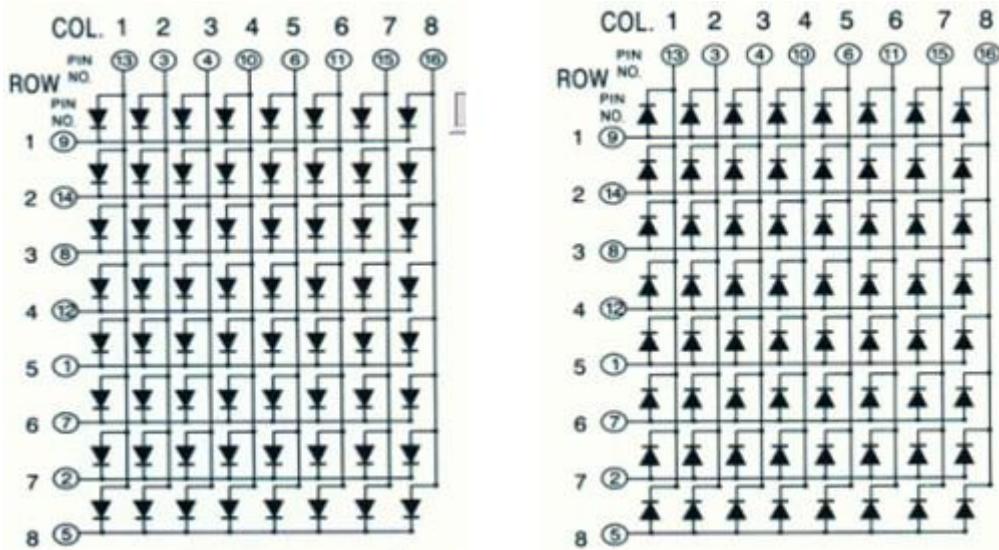


(2)Working principle of the 8*8 Dot-matrix Module

8x8 dot matrix display module can be divided into two types: common anode and common cathode according to different internal structures.

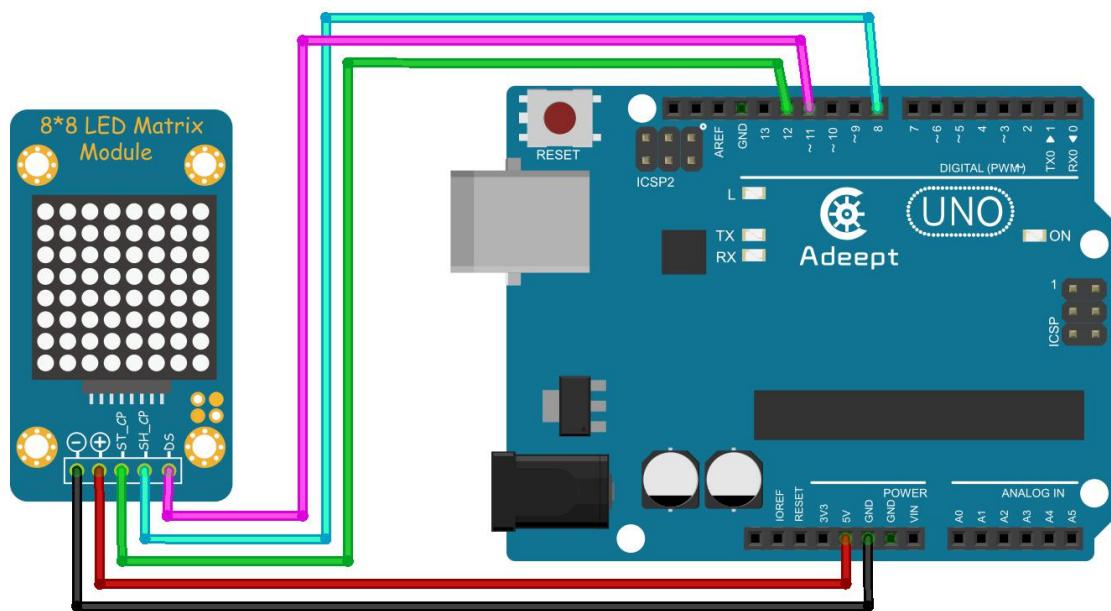
Common cathode: when a corresponding row is set to 1 level and a column is set to 0 level, the corresponding diode is on; if the first point is to be lit, pin 1 (bit selection signal) is connected to high Connect pin a (segment selection signal) to low level, then the first point will be on.

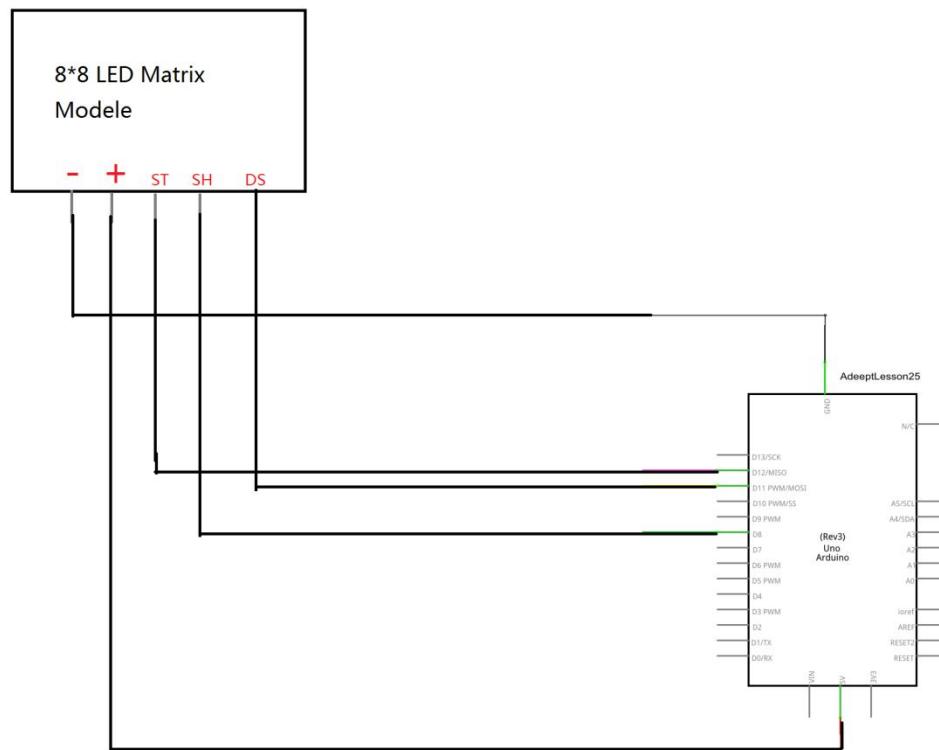
Common anode: when a corresponding row is set to 0 level and a column is set to 1 level, the corresponding diode will be on; if the first point is to be lit, pin 1 is connected to low level and pin a High level, then the first point is on, the common anode circuit is used in this experiment, so the bit selection signal is low when the code is written, the segment selection signal is high, and the segment selection bit The character selection is displayed in an array, from top to bottom for display from status to high. The internal structure is as follows: the left is a common cathode, the right is a common anode.



3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:





4. The Application of the 8*8 Dot-matrix Module

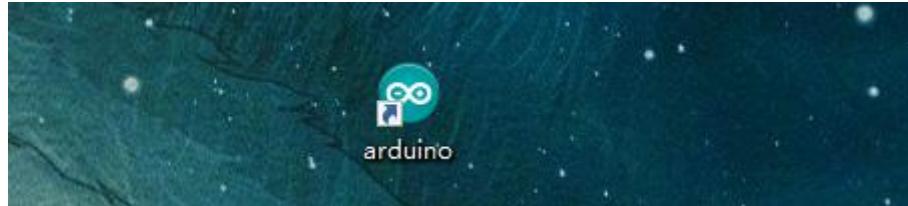
We provide two different methods to display character on 8*8 dot-matrix module. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++.

We will introduce these methods respectively.

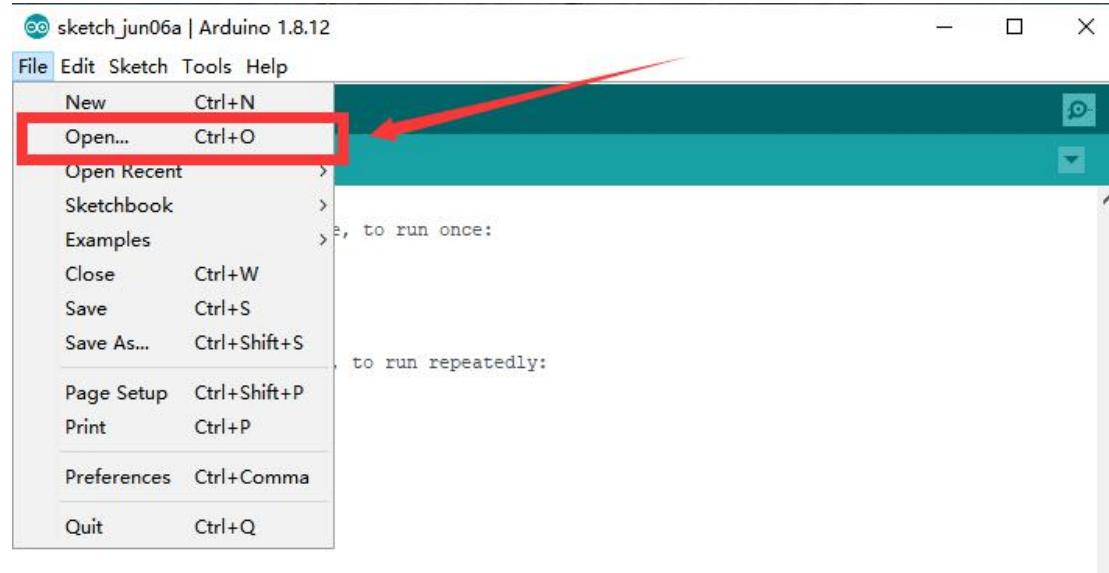
1. Using C language to program to display character on 8*8 dot-matrix module on Arduino UNO

(1) Compile and run the code program of this course

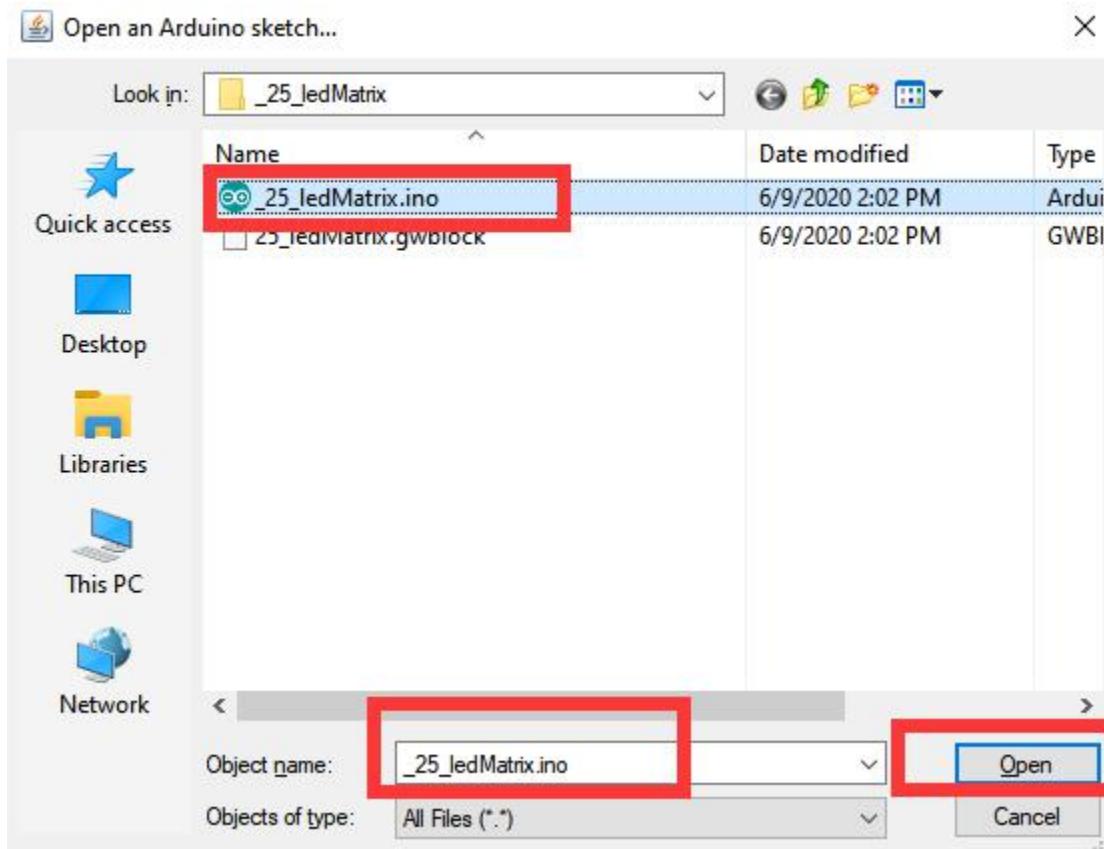
1. Open the Arduino IDE software, as shown below:



2.Click Open in the File drop-down menu:



3.Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\25_ledMatrix directory. Select_25_ledMatrixk.ino. This file is the code program we need in this course. Then click Open.



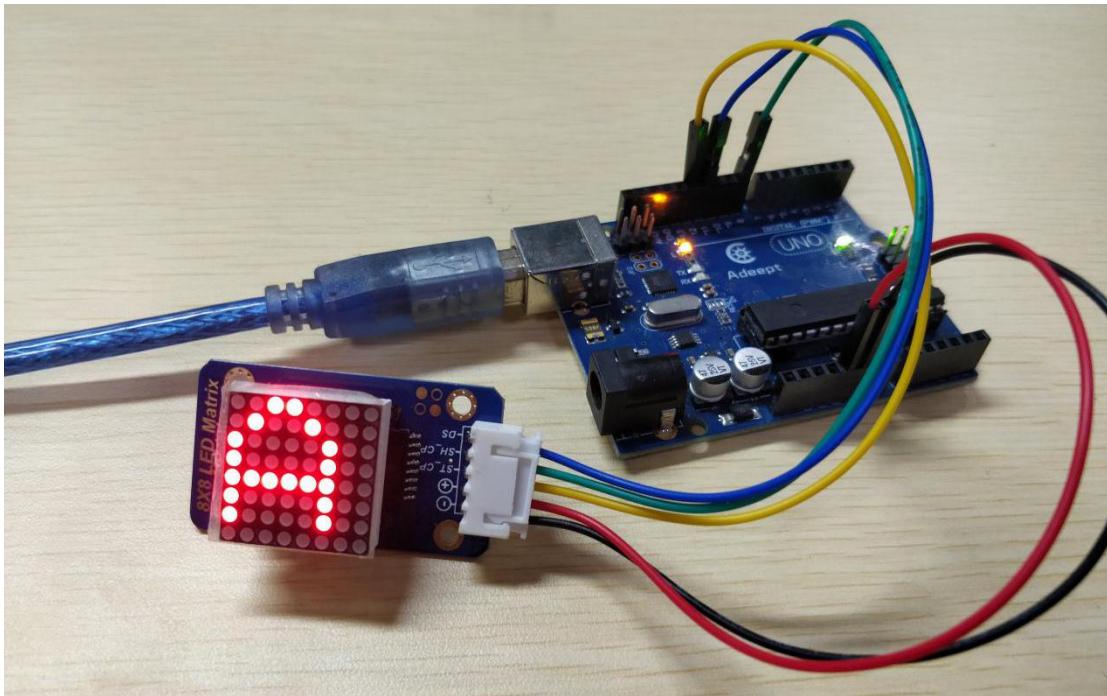
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                                         Arduino Uno on COM4
```

5. After successfully running the program, we observe that the dot-matrix module will display A, d, e, e, p, t in sequence. The physical connection diagram of the experiment is as follows:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to display character on 8*8 dot-matrix module on Arduino UNO. We will introduce how our core code can be achieved:

1. Define the text you need to display in the array data[]: Adeept

```
int data[] = { 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00, //Null
               0x00,0x3E,0x48,0x88,0x88,0x48,0x3E,0x00, //A
               0x00,0x00,0x0C,0x12,0x12,0xFE,0x00,0x00, //d
               0x00,0x00,0x7C,0x92,0x92,0x92,0x70,0x00, //e
               0x00,0x00,0x7C,0x92,0x92,0x92,0x70,0x00, //e
               0x00,0x00,0xFE,0x90,0x90,0x60,0x00,0x00, //p
               0x00,0x00,0x10,0x10,0x7E,0x12,0x10,0x00, //t
               0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 //Null
             };
```

2. In the loop () function, it is used to loop traversal through the indicator lights on the 8x8 dot matrix display module. If any indicator light is received, it is lit by digitalWrite (latchPin, HIGH) and digitalWrite (latchPin, LOW).

```

void loop()
{
    for(int n = 0; n < 56; n++) //Send column scanning cycle data
    {
        for(int t=0;t<100;t++) //Control data scrolling speed
        {
            for(int num=n; num < 8+n; num++)//8 columns of data sent to a dot matrix
            {
                shiftOut(dataPin,clockPin,MSBFIRST,data[num]); //Send column data to a dot matrix
                shiftOut(dataPin,clockPin,MSBFIRST,tab[num-n]);//Send line data to a dot matrix
                //The rising edge of the data shift
                digitalWrite(latchPin,HIGH); //Output control latch HIGH
                digitalWrite(latchPin,LOW); //Output control latch LOW
            }
        }
    }
}

```

2.Using graphical code blocks to program to display character on 8*8 dot-matrix module on Arduino UNO

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1.Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

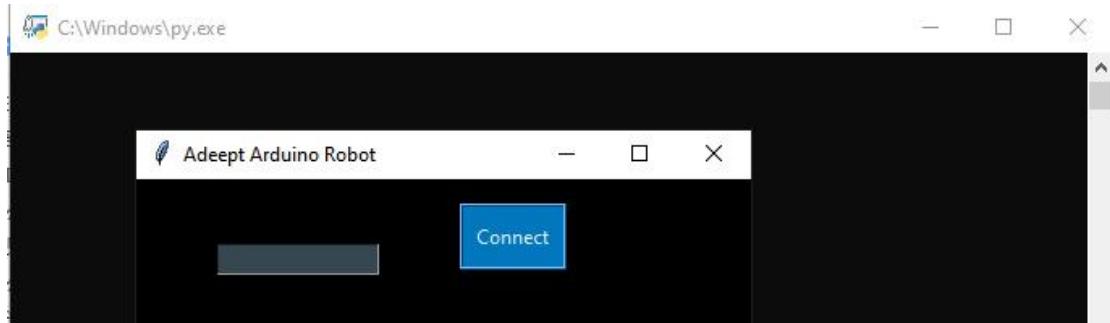
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

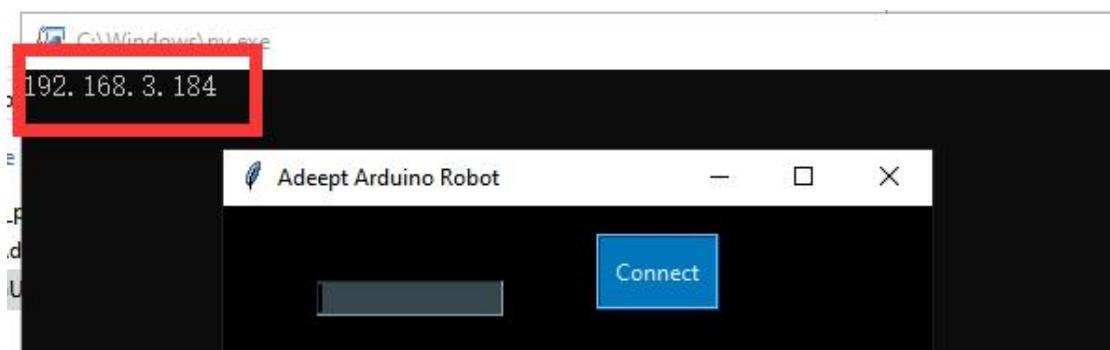
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

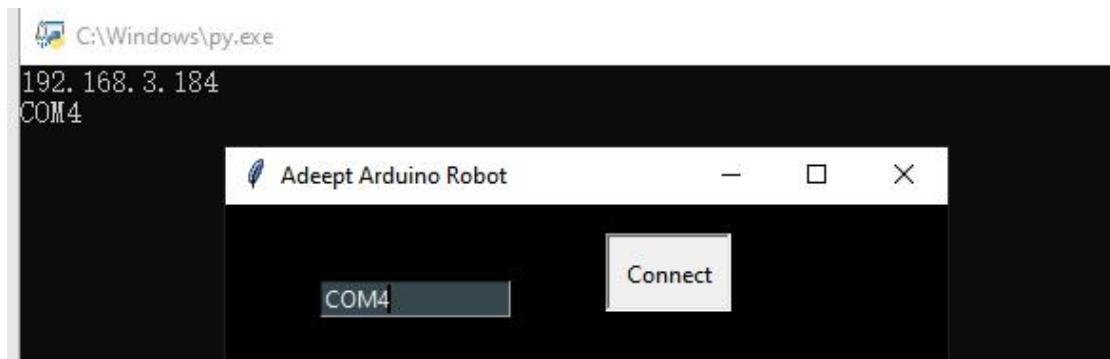
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



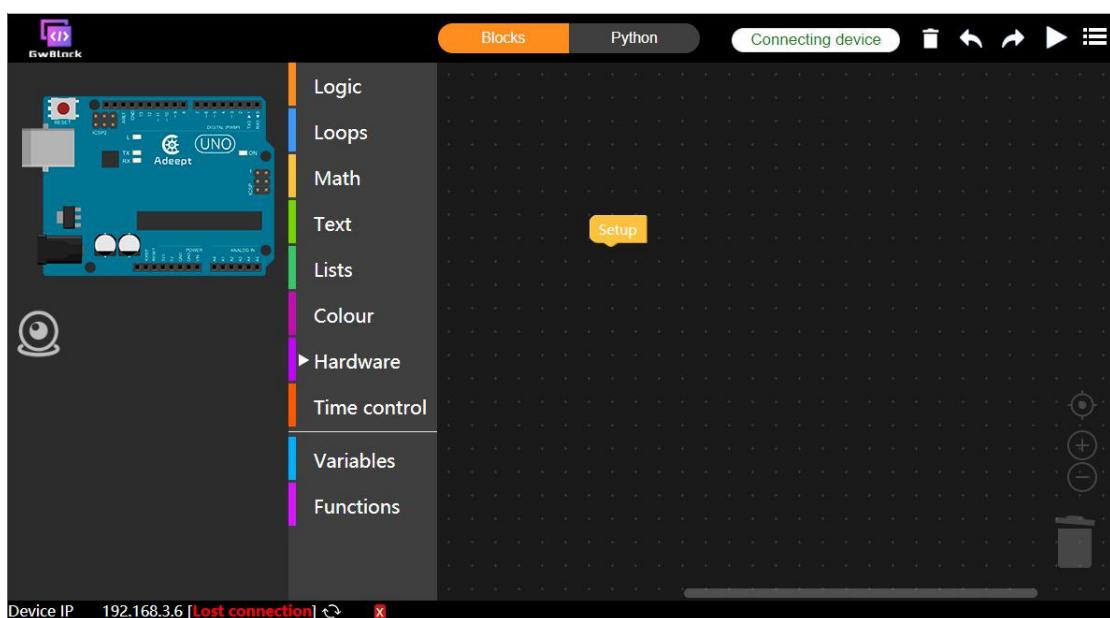
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



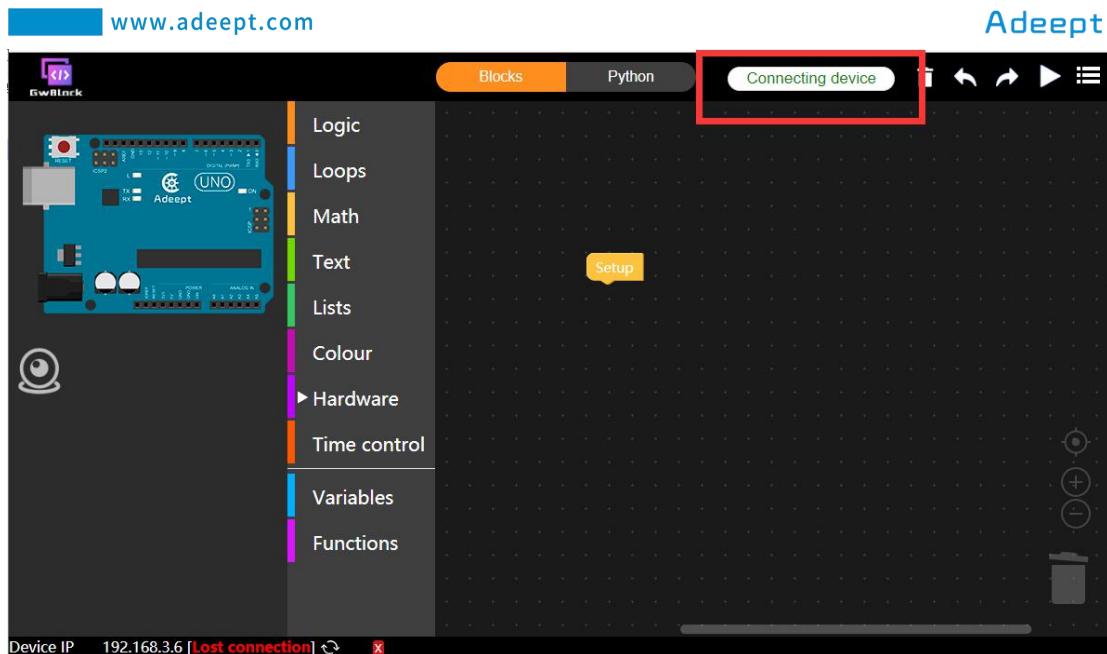
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

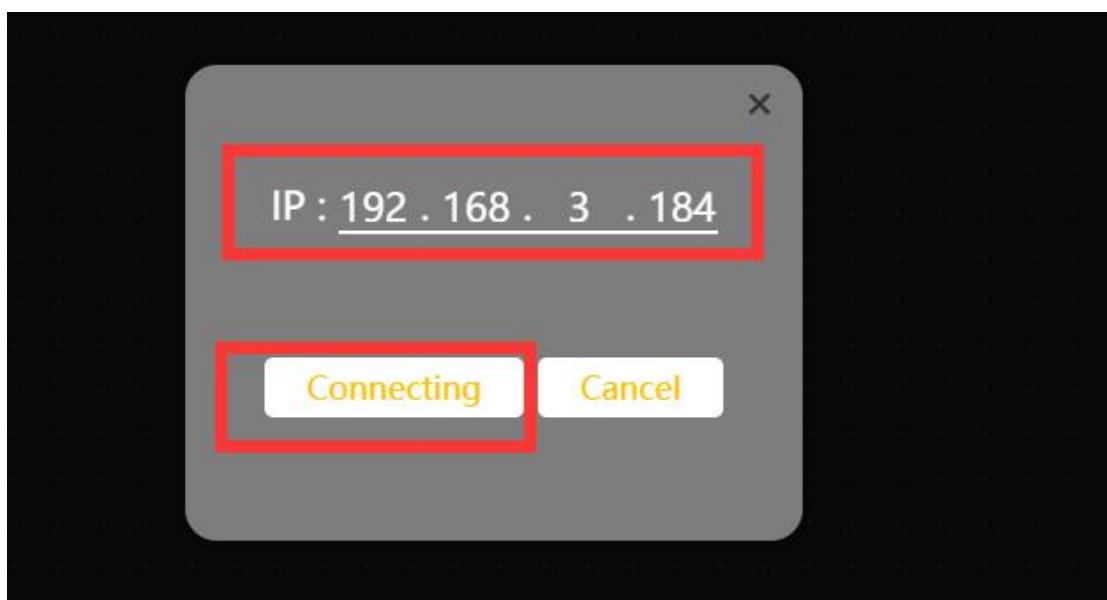
After successfully entering the website, the interface is as follows:



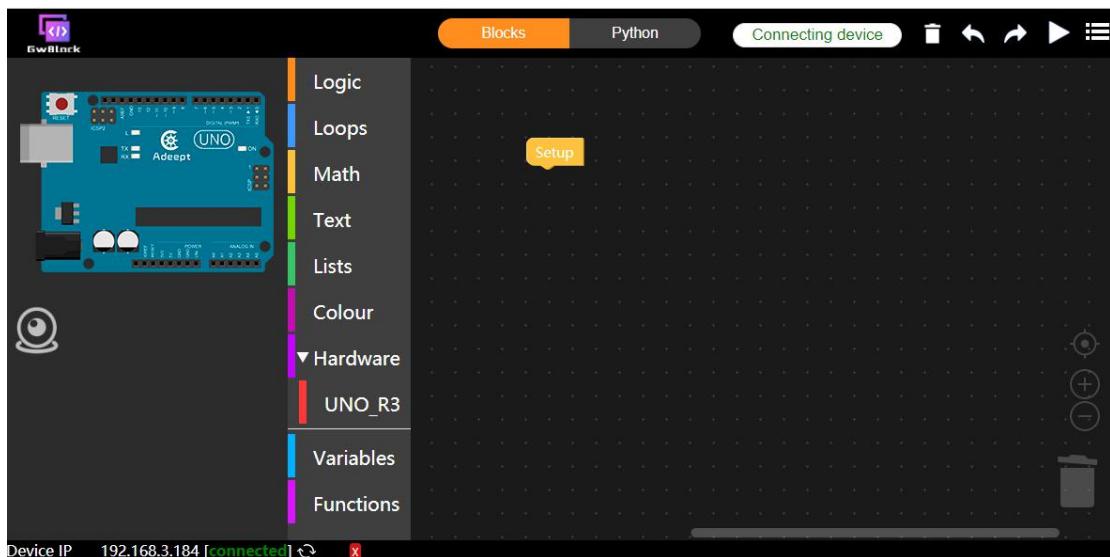
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:



(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

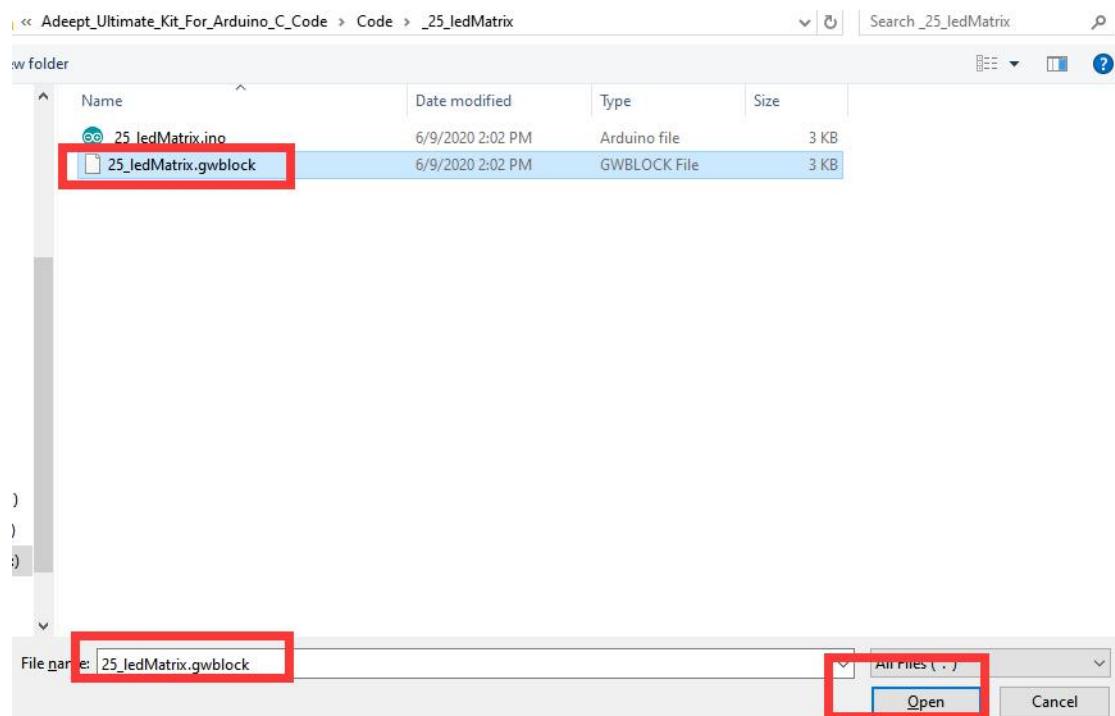
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_25_ledMatrixk

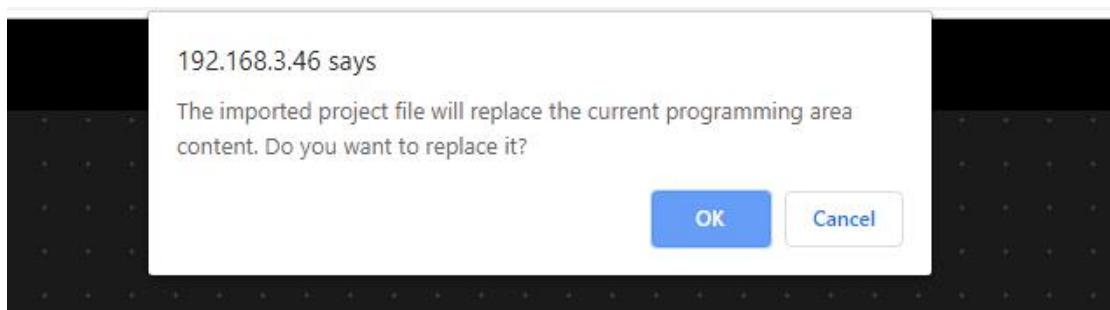
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

5. Select the "25_ledMatrixk.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



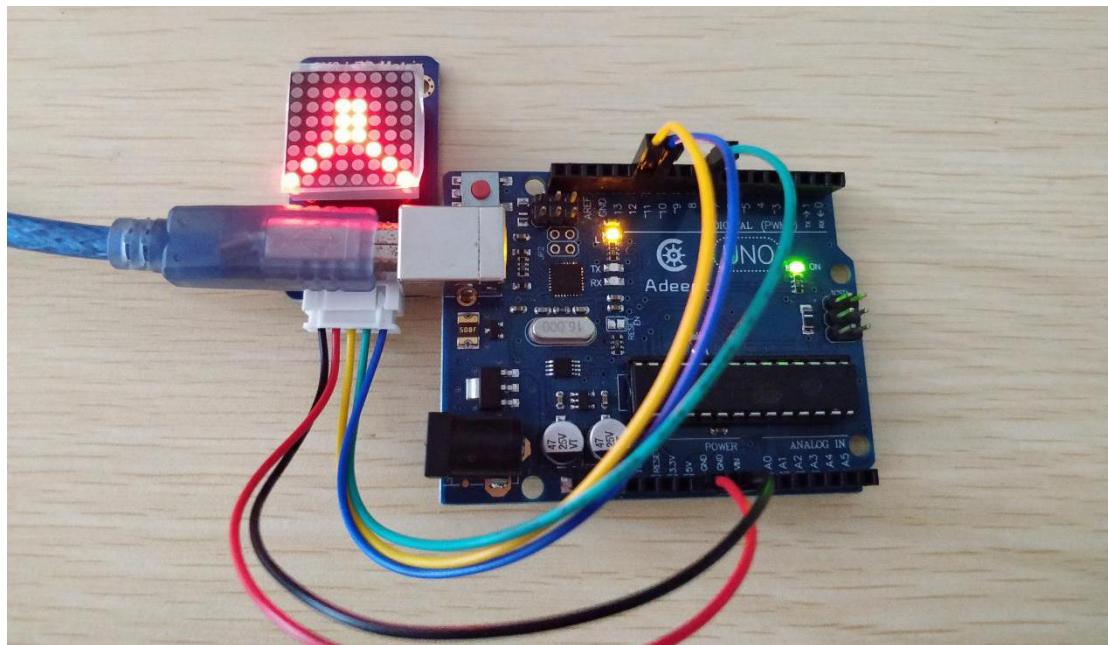
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



8. Click the button  on the upper right corner. After successfully running the program, you can observe that the corresponding pattern will be displayed on the 8*8 dot-matrix module. The physical connection diagram of the experiment is as follows:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to display character on 8*8 dot-matrix module on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

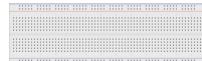
In the GwBlock graphical editor, all code programs are executed from . You can use the command block  to control the display pattern of the 8*8 dot-matrix module. Initialize the data ports 12, 8, and 11 connected to the Arduino UNO with the command block . In the command block , you can display the picture you want by ticking the led.



Lesson 26 Controlling the Stepper Motor

In this lesson, we will learn how to control the Stepper Motor.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
ULN2003-based Stepper Motor Driver	1	
Stepper Motor	1	

2. The introduction of the Stepper Motor

(1) Stepper Motor

A stepper motor is a motor that converts electrical pulse signals into corresponding angular or linear displacements. Each time a pulse signal is input, the rotor rotates by an angle or a step forward. The output angular displacement or linear displacement is proportional to the number of pulses input, and the rotation speed is proportional to the pulse frequency. Therefore, stepper motors are also called pulse motors. There are two types of steppers, unipolar and bipolar. It is important to know which type to use. In this lesson, we will use a unipolar stepper type stepper motor.



(2) Working principle of the Stepper Motor

Stepper motor is also known as pulse motor, based on the most basic electromagnet principle, it is a kind of free rotation electromagnet, its operation principle is to rely on the change of air gap permeability to generate electromagnetic torque. The stepper's axis is controlled by a series of solenoid coils that are charged positively and negatively in a particular order to move it precisely forward or backward at a small "step size." The advantage of a stepper motor is that it can be precisely positioned, turn "one step" forward or backward, and turn continuously. The biggest difference between stepping motor and other control motors is that it receives digital control signals (electrical impulse signals) and converts them into corresponding angular displacement or linear displacements. It is an executive element to complete the transformation of digital mode. And it can be open loop position control, input a pulse signal to get a specified position increment, such a so-called incremental position control system compared with the traditional dc control system, its cost is significantly reduced, almost no system adjustment. The angular displacement of the stepper motor is strictly proportional to the number of input pulses and synchronizes with the pulses in time. Therefore, as long as the number of pulses, frequency and phase sequence of motor windings are controlled, the desired Angle, speed and direction can be obtained.

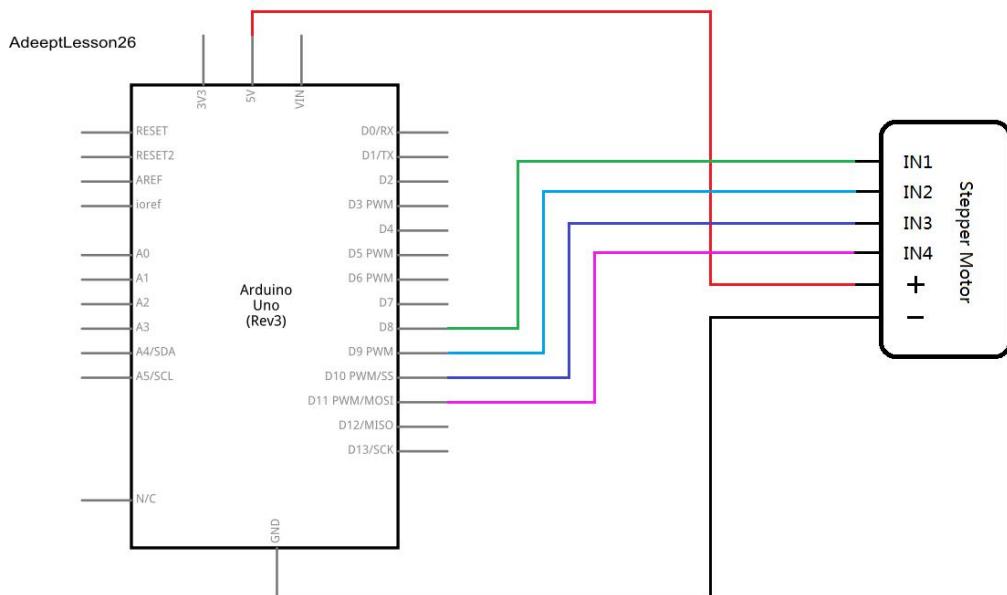
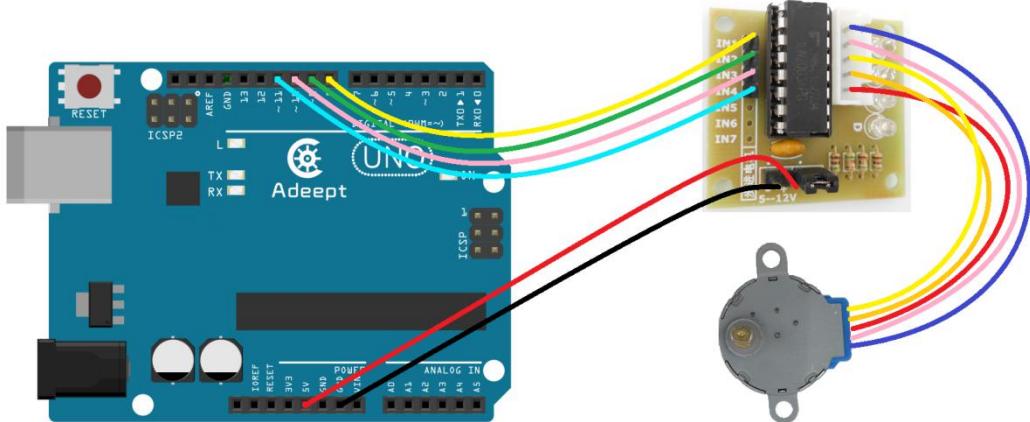
(3) Stepper Motor Driver Module (Based on ULN2003A)

Due to low current, Arduino UNO development board's digital port is unable to drive the stepping motor directly. Therefore, a driving circuit must be used to control the stepper motor. The driver module based on ULN2003 is used in this lesson. There are four LEDS on the module. The white socket in the middle is used to connect the stepper motor. IN1, IN2, IN3, IN4 are used to connect with Arduino UNO development board's digital port.



3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:



4. How to control the Stepper Motor

We provide two different methods to control the stepper motor. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

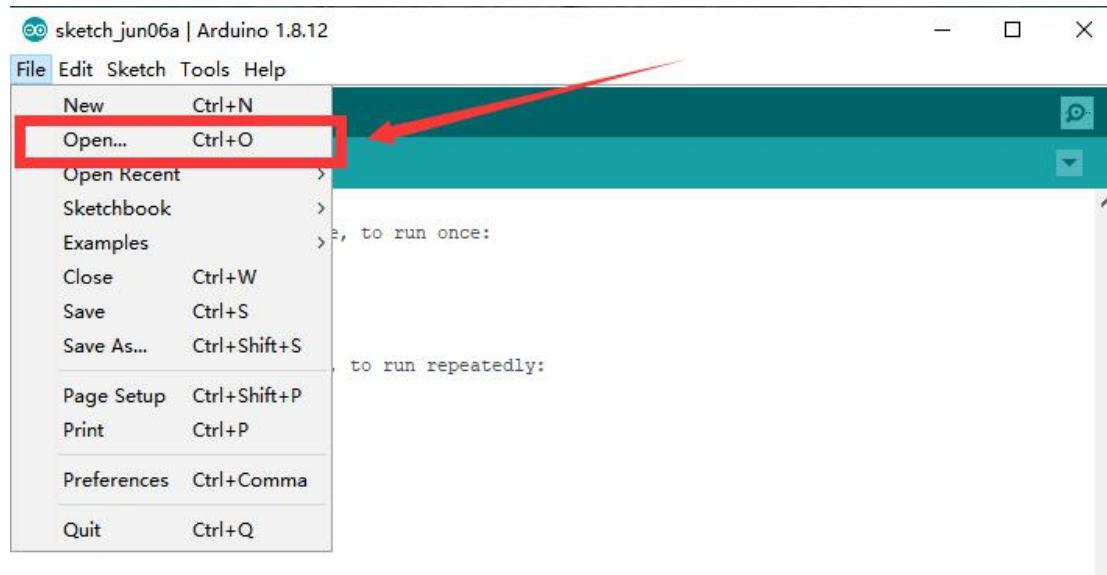
1.Using C language to program to control the Stepper Motor on Arduino UNO

(1)Compile and run the code program of this course

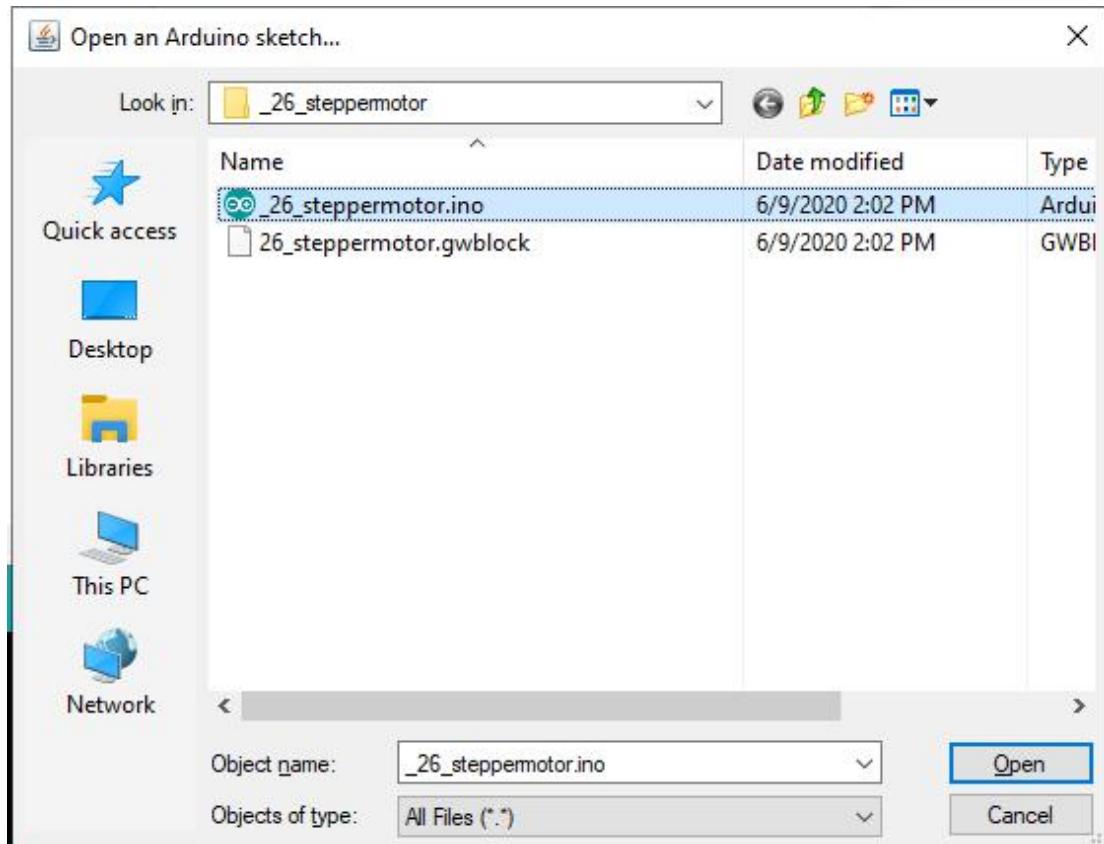
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\26_steppermotor directory. Select _26_steppermotor.ino. This file is the code program we need in this course. Then click Open.



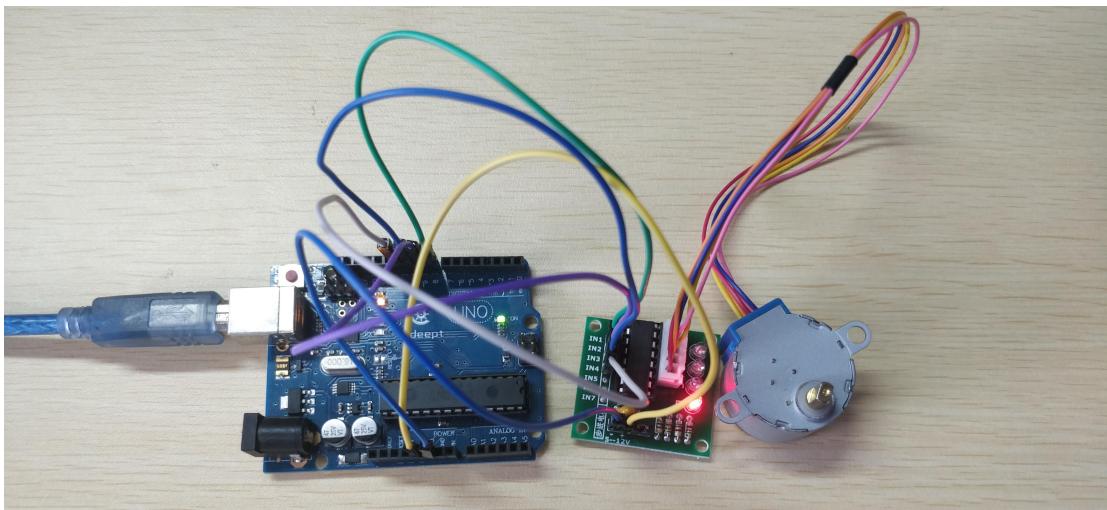
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After successfully running the program, you will observe that the stepper motor will rotate. The physical connection diagram of the experiment is as follows:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to control the stepper motor on Arduino UNO. We will introduce how our core code can be achieved:

In the setup() function, set Pin0, Pin1, Pin2, and Pin3 to OUTPUT mode with the pinMode() function. Then the Speed (15) is used to control the stepper motor to rotate at a stepping speed of 15. And the stepper motor can be set to rotate 1024 degrees by Step (1024). 360 degrees is a circle.

```
void setup()
{
    pinMode(Pin0, OUTPUT); //Set digital 8 port mode, the OUTPUT for the output
    pinMode(Pin1, OUTPUT); //Set digital 9 port mode, the OUTPUT for the output
    pinMode(Pin2, OUTPUT); //Set digital 10 port mode, the OUTPUT for the output
    pinMode(Pin3, OUTPUT); //Set digital 11 port mode, the OUTPUT for the output
    Speed(15); //Stepper motor speed = 15 fast (note:speed from 1 to 15)
    Step(1024); //Stepper motor forward 512 steps ---- 360 angle
}
```

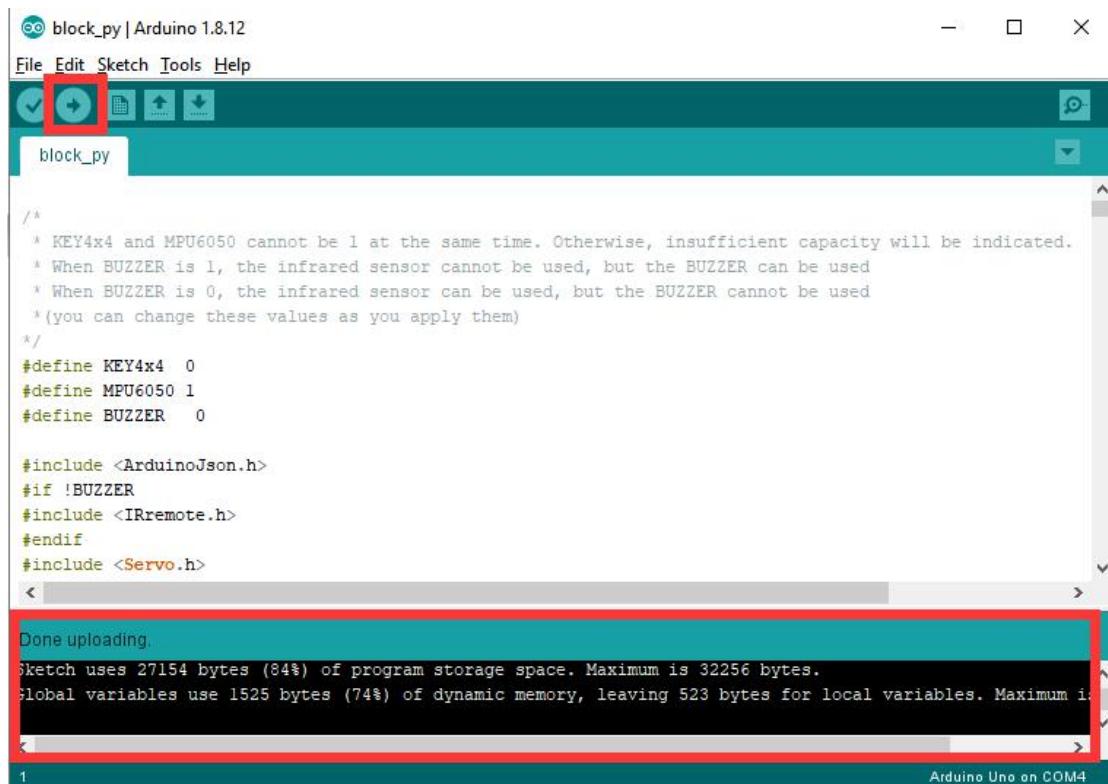
2.Using graphical code blocks to program to control the stepper motor on Arduino UNO

(1)Connecting to GwBlock graphical editor

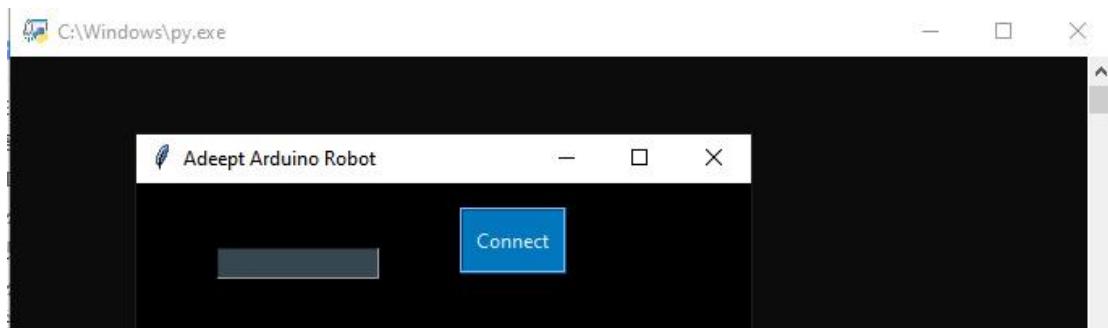
In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user:

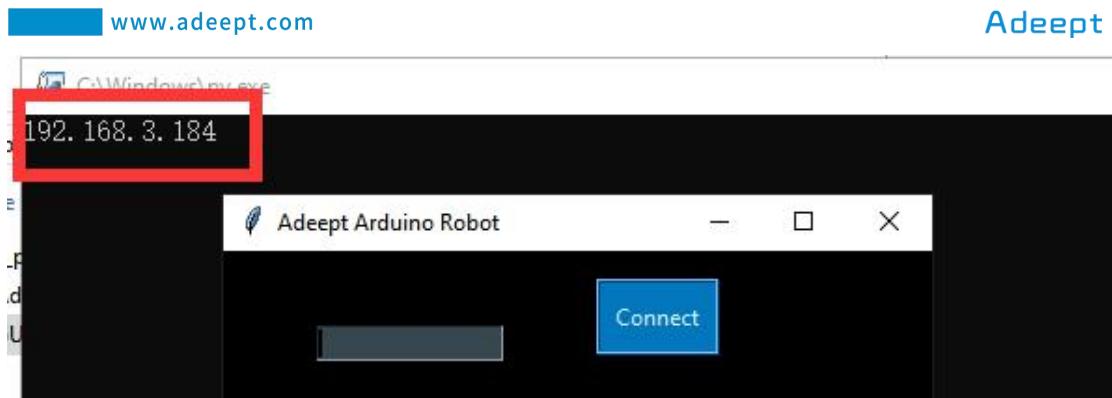
Adept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:



2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3.Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



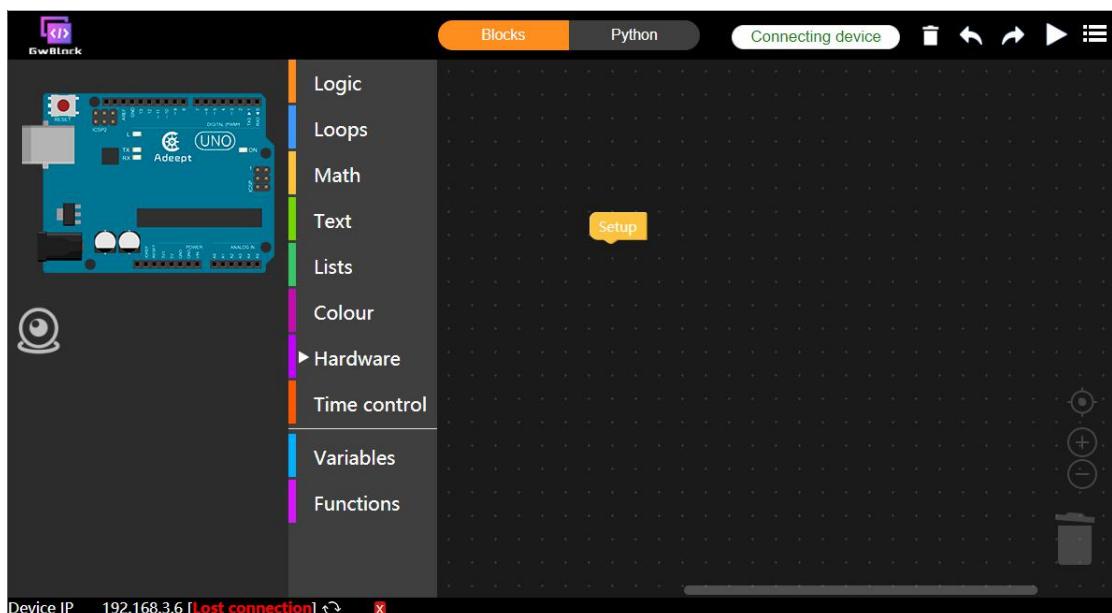
4. Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



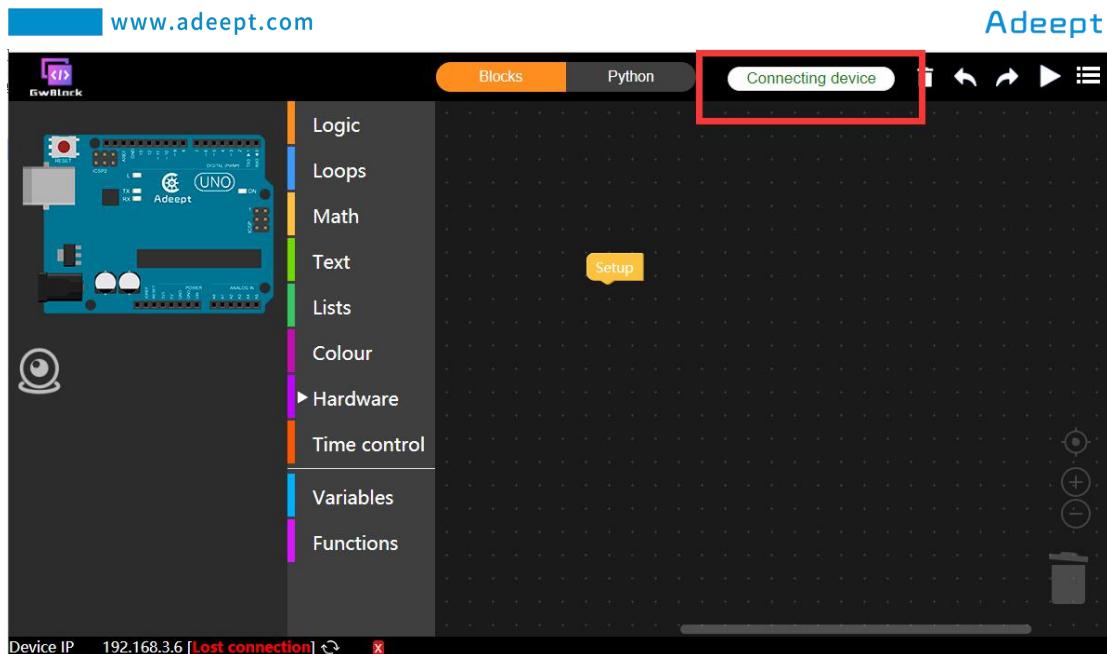
5. Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

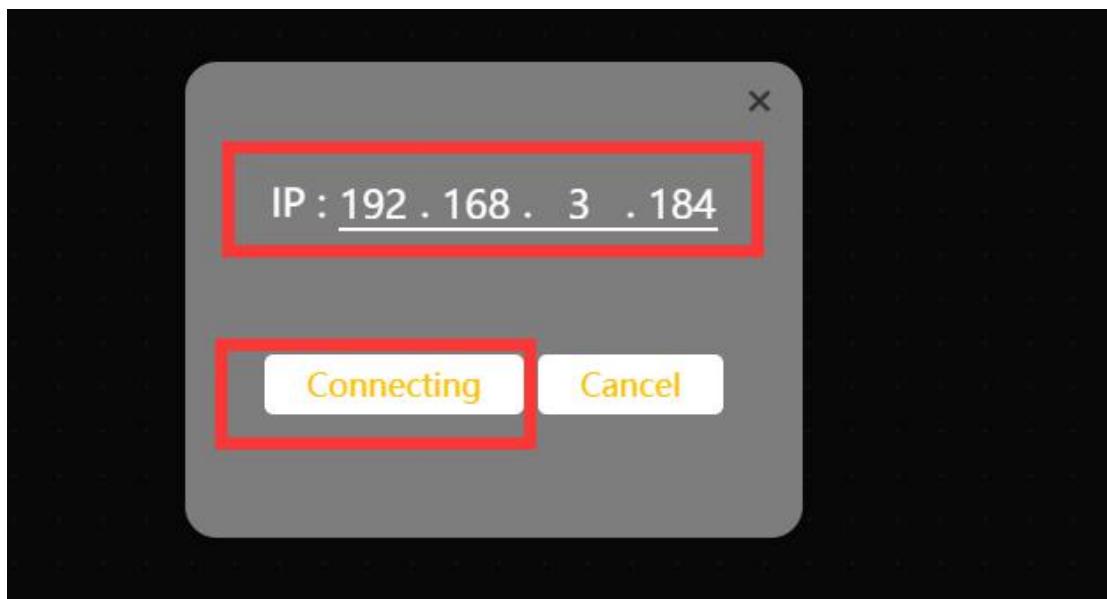
After successfully entering the website, the interface is as follows:



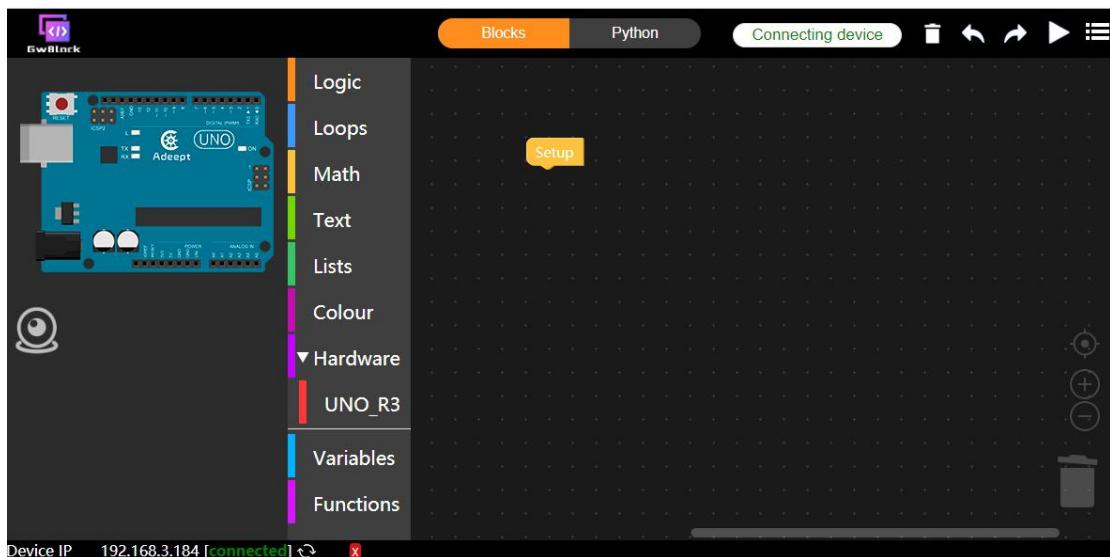
6. Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

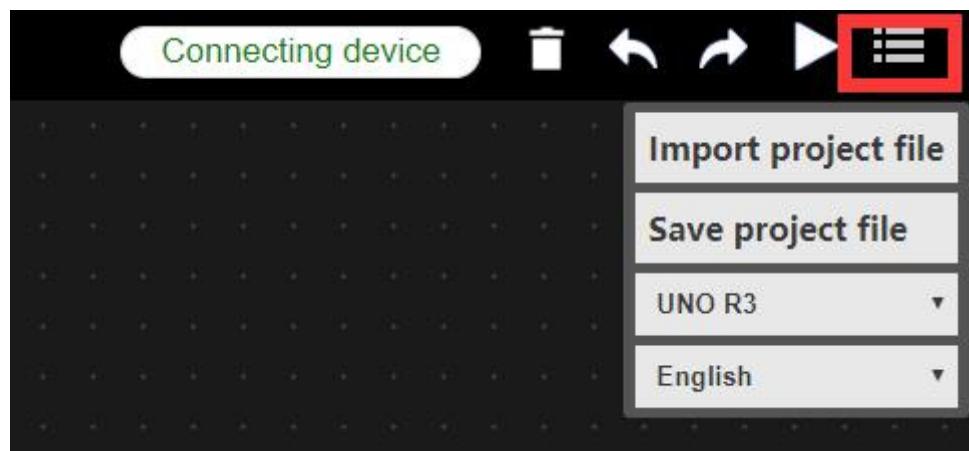


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3.The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

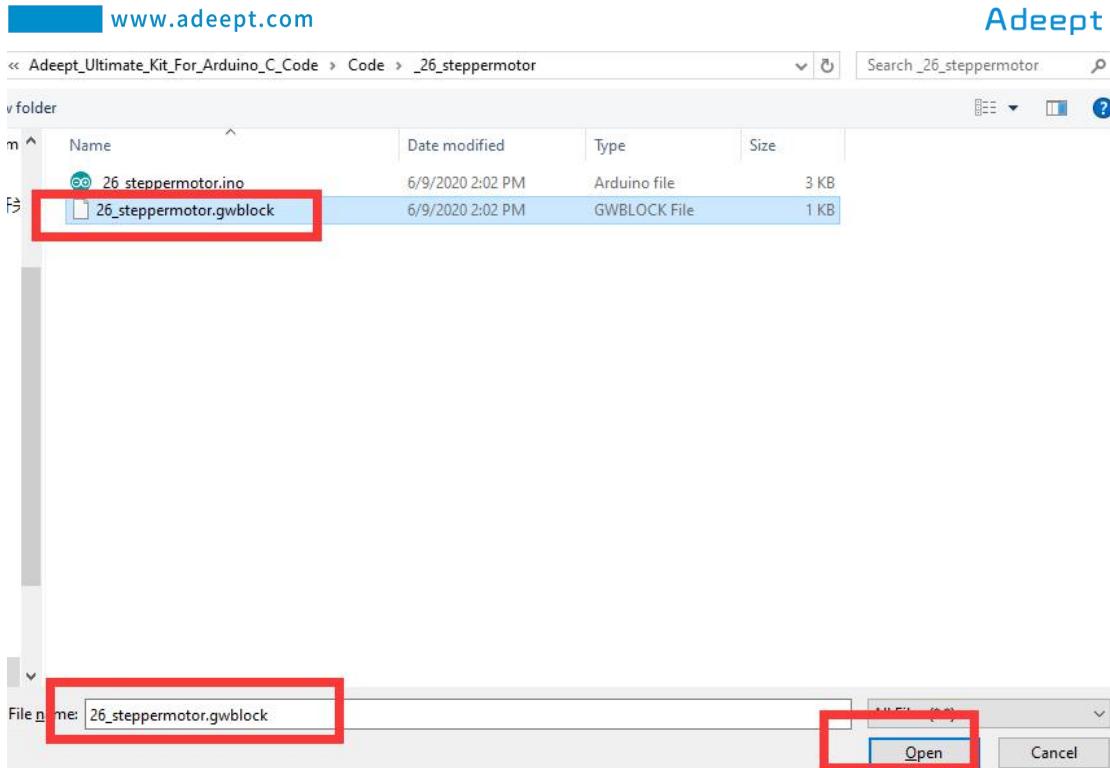
4.Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_26_steppermotor

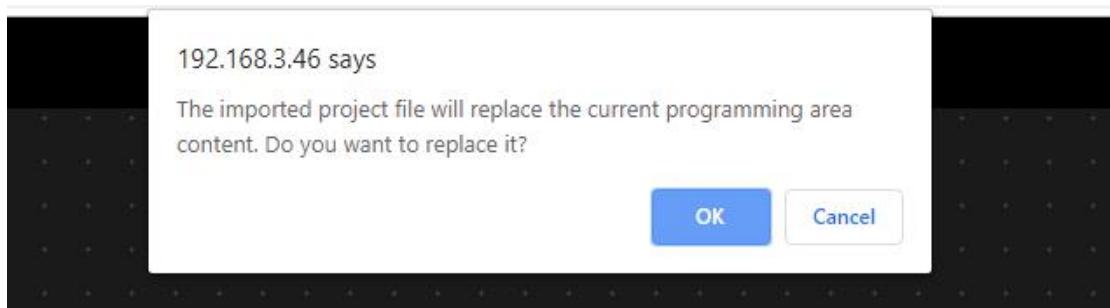
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

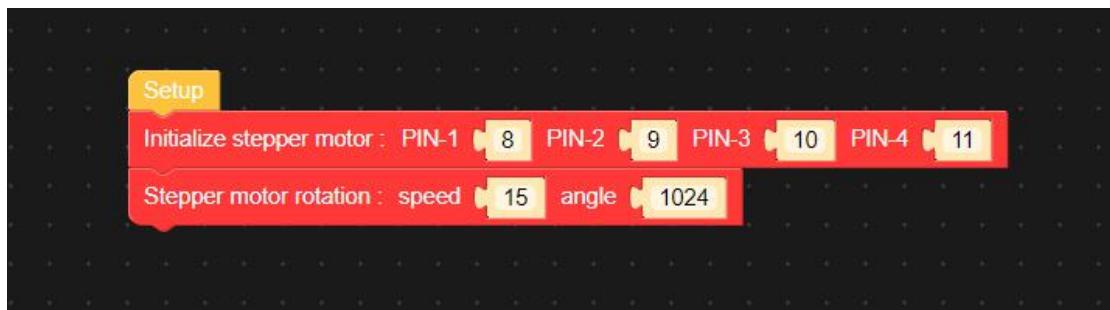
5. Select the "26_steppermotor.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



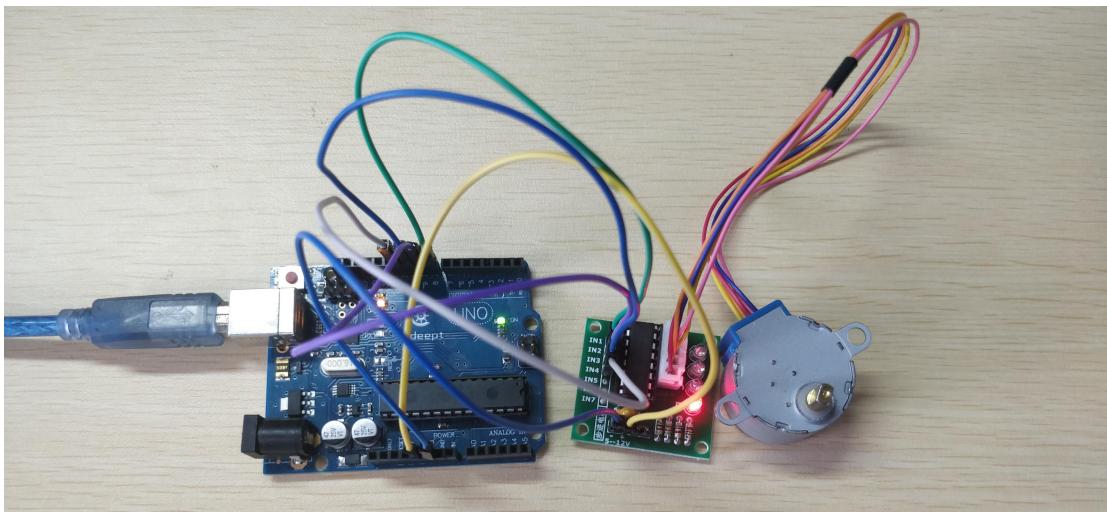
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



8. Click the button  on the upper right corner.After running the program successfully, you will observe that the stepper motor will rotate. The physical connection diagram of the experiment is as follows:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to control the stepper motor on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

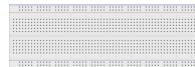
In the GwBlock graphical editor, all code programs are executed from **Setup**. The instruction block **Stepper motor rotation : speed [15] angle [1024]** can directly control the stepper motor to rotate 1024 degrees at a step speed of 15. 360 degrees is a circle.



Lesson 27 Using Photoresistor to Measuring Light intensity

In this lesson, we will learn how to use Photoresistor to measure light intensity.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
LCD1602	1	
Adeept IIC Module	1	
Male to Female Jumper Wires	Several	
Resistor(1KΩ)	1	
Photoresistor	1	

2. The introduction of the Photoresistance

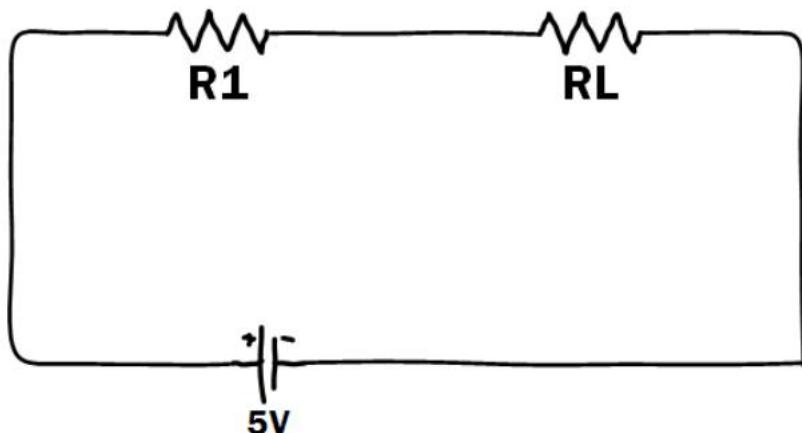
(1)Photoresistance

Photovaristor (photovaristor), also known as photoresistor, is a kind of resistor made by using the photoelectric effect of semiconductor to change the resistance value with the intensity of incident light;If the incident light is strong, the resistance decreases, and if the incident light is weak, the resistance increases. Photoresistors are

generally used for light measurement, light control, and photoelectric conversion (converting changes in light into changes in electricity).

Photoresistors can be widely used in various light control circuits, such as the control and adjustment of lights, and can also be used in light control switches.

In dark, dark conditions, the resistance of the photoresistor is very high. The stronger the light, the smaller the resistance value. By measuring the voltage change value on both sides of the photoresistor, the change of the photoresistor value can be known and the light intensity value can be obtained. In the connection diagram, we can find a voltage divider connected in series with the photoresistor.



In the above figure, RL is the photoresistor, R1 is the voltage-dividing resistor in series, $V_{out} = \frac{R_L}{R_1 + R_L} \cdot V_{in}$. In the dark, the resistance of RL will be very large, so V_{out} will be very Large, close to 5V. Once the light is irradiated, the value of RL will decrease rapidly, so V_{out} will decrease accordingly. It can be seen from the above formula that R1 should not be too small, preferably around 1k~10k, otherwise the ratio will not change significantly.

(2) Experimental principle

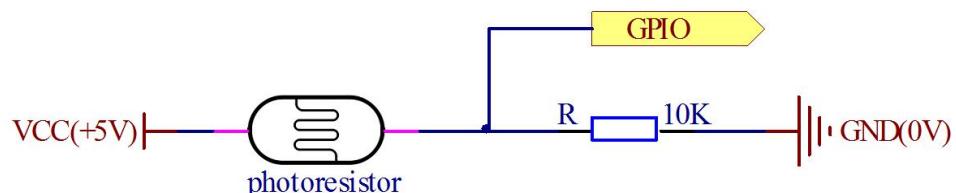
In this experiment, we first conduct a relatively simple experiment of using a photoresistor. Since the photoresistor is a component that can change the resistance value according to the light intensity, naturally the analog port is also required to read the analog value. In this experiment, the PWM interface experiment can be used to

replace the potentiometer with a photoresistor to realize the LED light when the light intensity is different. The brightness will also change accordingly.

A photoresistor is a light-controlled variable resistor. The resistance of a photoresistor decreases with the increasing incident light intensity; in other words, it exhibits photoconductivity. A photoresistor can be applied in light-sensitive detector circuits.

A photoresistor is made of a high resistance semiconductor. In the dark, a photoresistor can have a resistance as high as a few megohms ($M\Omega$), while in the light, a photoresistor can have a resistance as low as a few hundred ohms. If incident light on a photoresistor exceeds a certain frequency, photons absorbed by the semiconductor give bound electrons enough energy to jump into the conduction band. The resulting free electrons (and their hole partners) conduct electricity, thereby lowering resistance. The resistance range and sensitivity of a photoresistor can substantially differ among dissimilar devices. Moreover, unique photoresistors may react substantially differently to photons within certain wavelength bands.

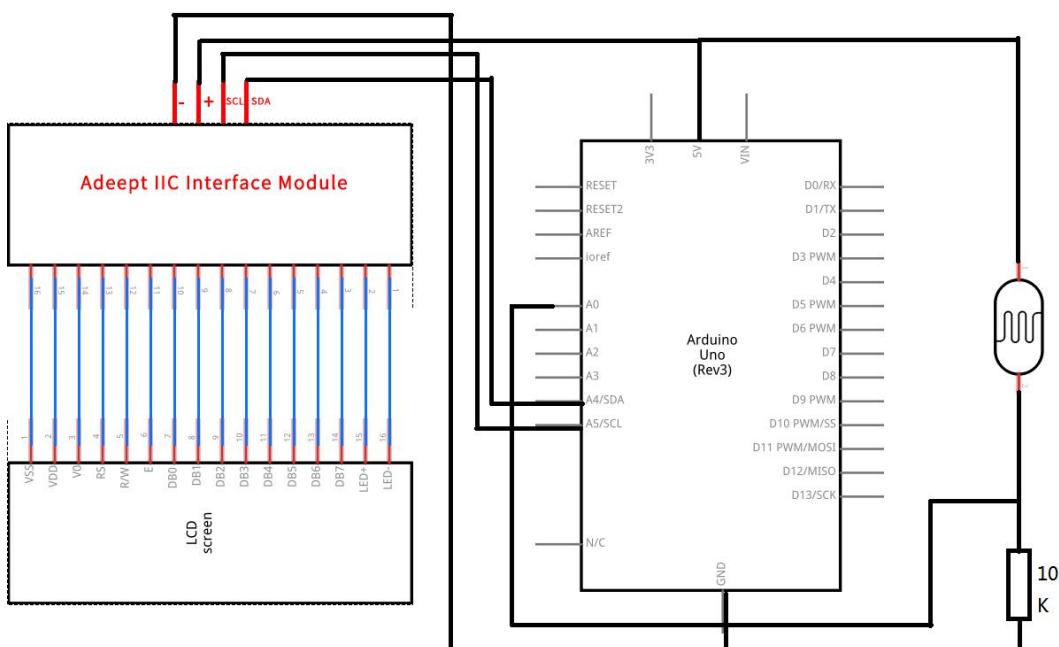
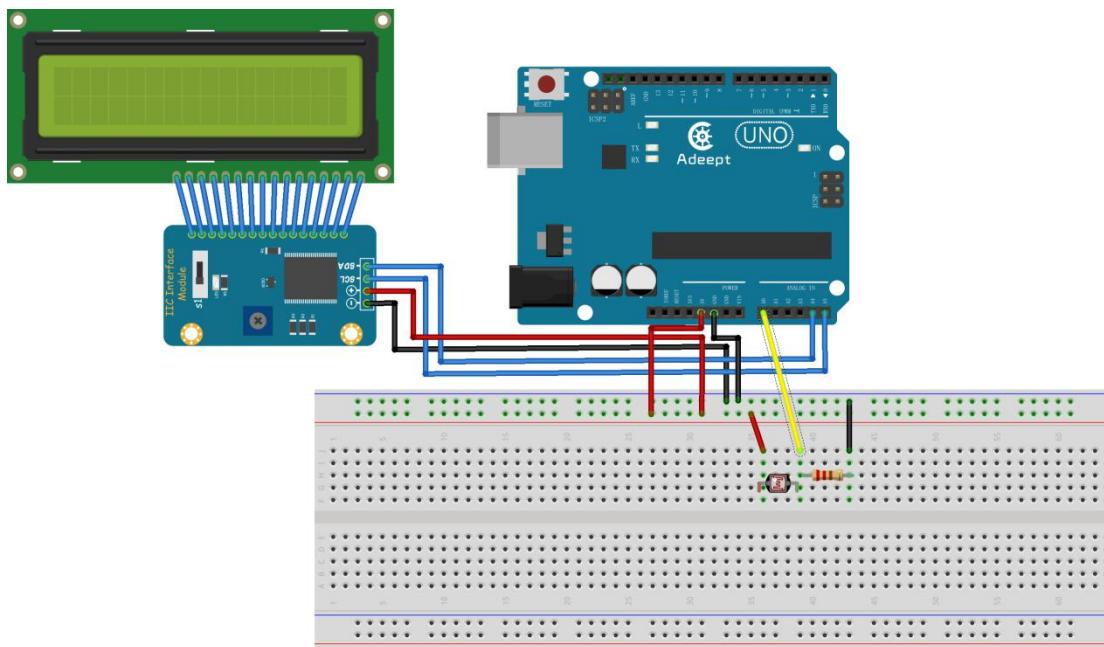
The schematic diagram of this experiment is shown below:



With the increase of the light intensity, the resistance of photoresistor will be decreased. The voltage of GPIO port in the above figure will become high.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. The resistance is $1K\Omega$. As shown in the following figure:



4.Using photoresistor to measure light intensity

We provide two different methods to obtain the light intensity. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not

mastered the advanced programming languages C and C++. We will introduce these methods respectively.

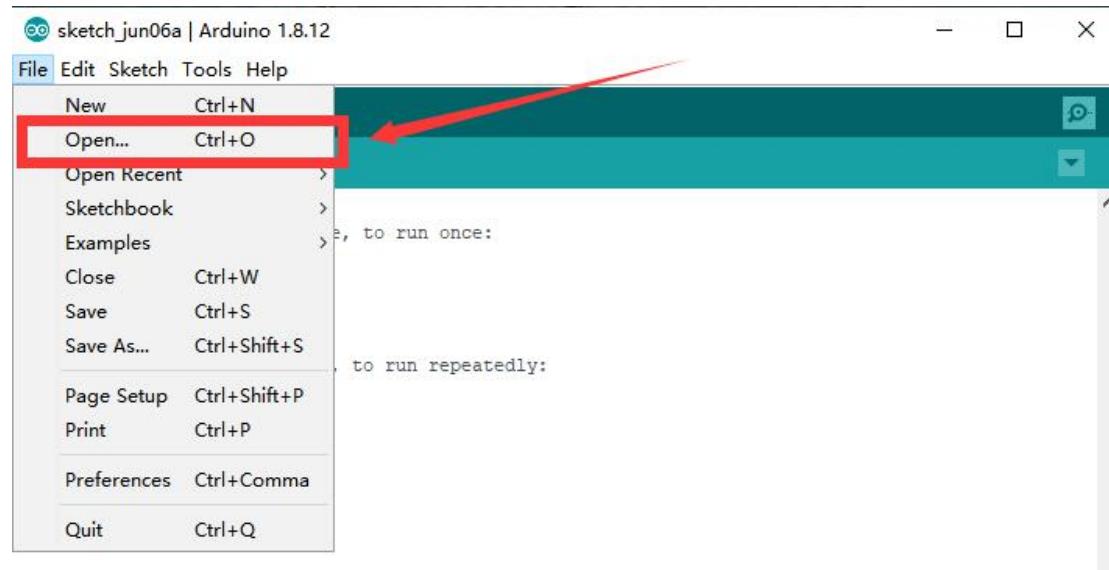
1.Using C language to program to use Photoresistor to measure light intensity on Arduino UNO

(1)Compile and run the code program of this course

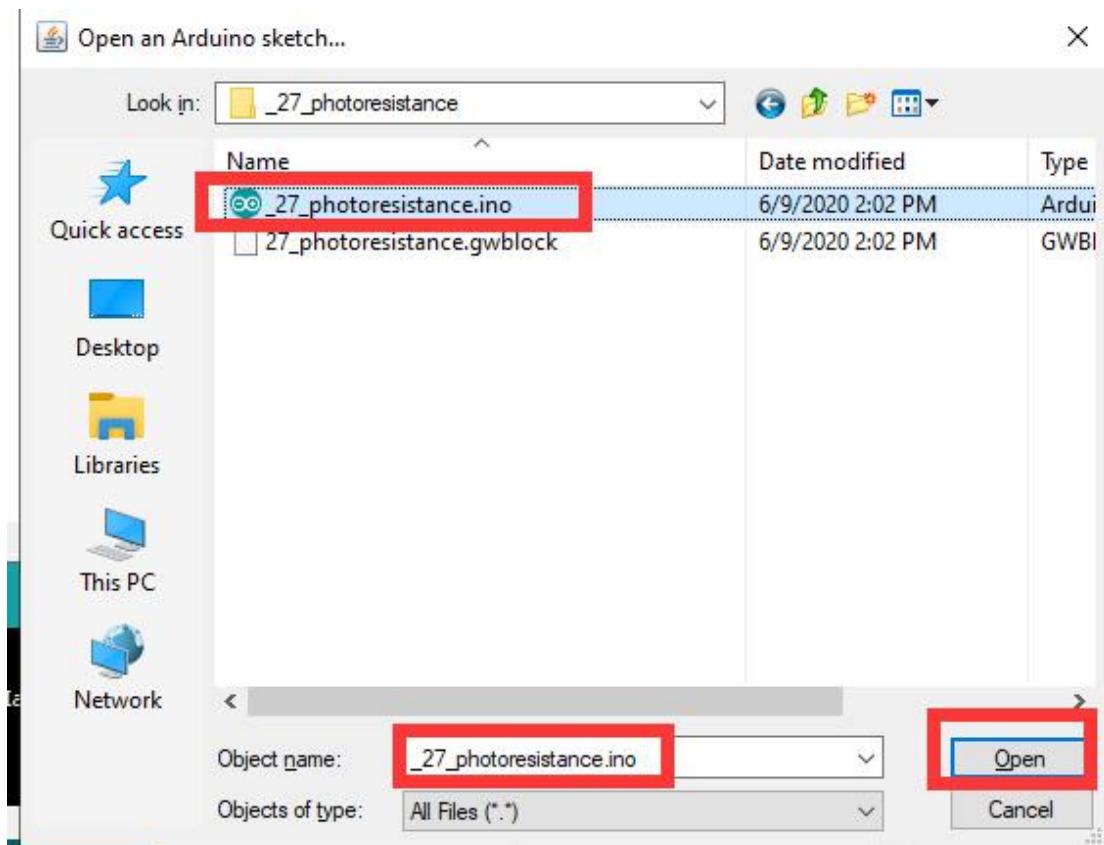
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\27_photoresistanc directory. Select _27_photoresistance.ino. This file is the code program we need in this course. Then click Open.



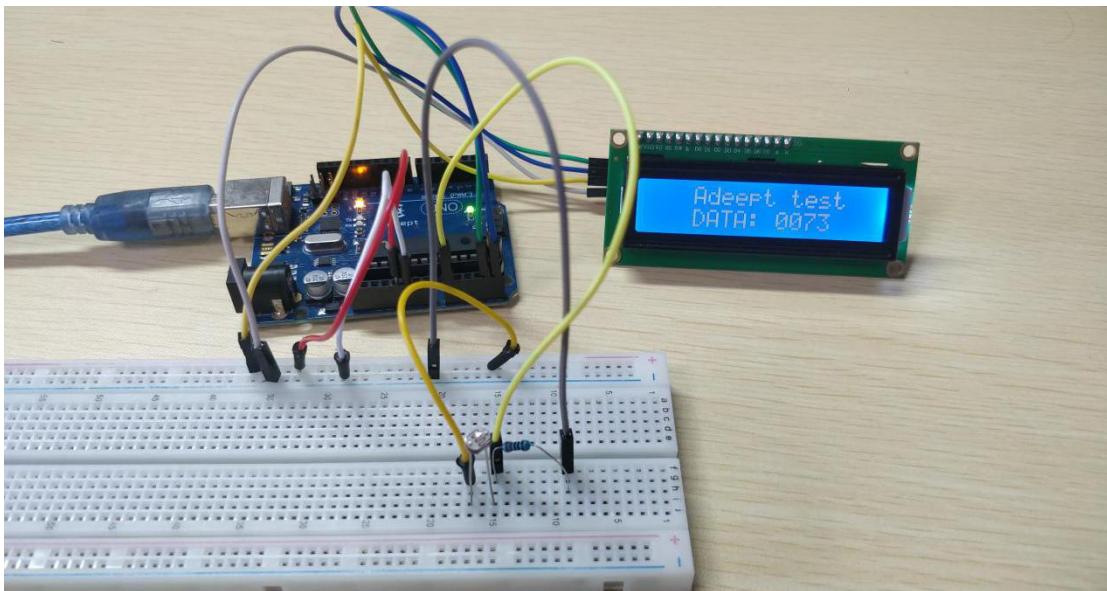
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                                         Arduino Uno on COM4
```

5. After running the program successfully, you will observe that the detected light intensity displays on the LCD1602 screen. The physical connection diagram of the experiment is as follows:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to use Photoresistor to measure light intensity on Arduino UNO. We will introduce how our core code can be achieved:

1.In the setup() function, displayed "Adeept test" on the LCD1602 screen with the lcd.print(array1[positionCounter1]) function in the for loop statement, .

```
void setup()
{
    lcd.init(); //initialize the lcd
    lcd.backlight(); //turn on the backlight
    lcd.clear(); //Clears the LCD screen and positions the cursor in the upper-left corner
    lcd.setCursor(0,0); // set the cursor to column 15, line 0
    for (int positionCounter1 = 0; positionCounter1 < 16; positionCounter1++)
    {
        lcd.print(array1[positionCounter1]); // Print a message to the LCD.
        delay(250); //wait for 250 microseconds
    }
}
```

2.In the loop() function, display the data DATA of the photoresistor on the LCD1602 screen by calculating with the lcd.print(array2[positionCounter3]) function in the for loop statement.

```

void loop()
{
    array2[9]=analogRead(photoresistorPin)/1000+0x30;    //Take one thousand - bit data
    array2[10]=analogRead(photoresistorPin)/100%10+0x30;//Take one hundred - bit data
    array2[11]=analogRead(photoresistorPin)/10%10+0x30; //Take ten-bit data
    array2[12]=analogRead(photoresistorPin)%10+0x30;     //Take a bit of data
    lcd.setCursor(0,1);                                // set the cursor to column 15, line 1
    for (int positionCounter3 = 0; positionCounter3 < 16; positionCounter3++)
    {
        lcd.print(array2[positionCounter3]);           // Print a message to the LCD.
        delay(tim);                                 //wait for 250 microseconds
    }
}

```

2.Using graphical code blocks to program to use Photoresistor to measure light intensity on Arduino UNO

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

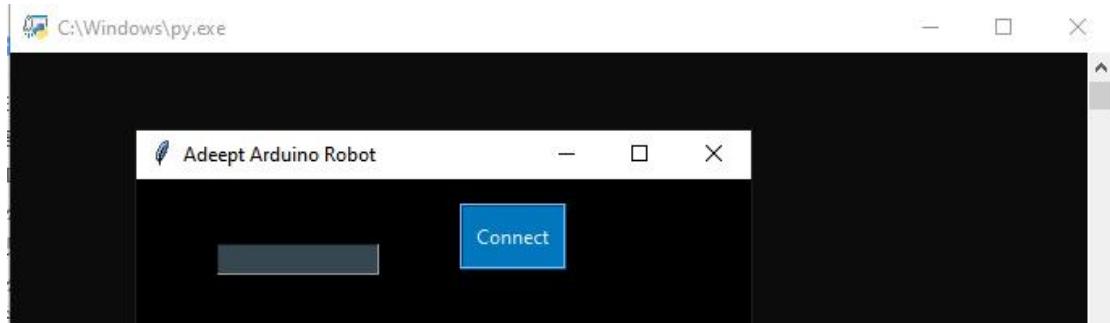
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

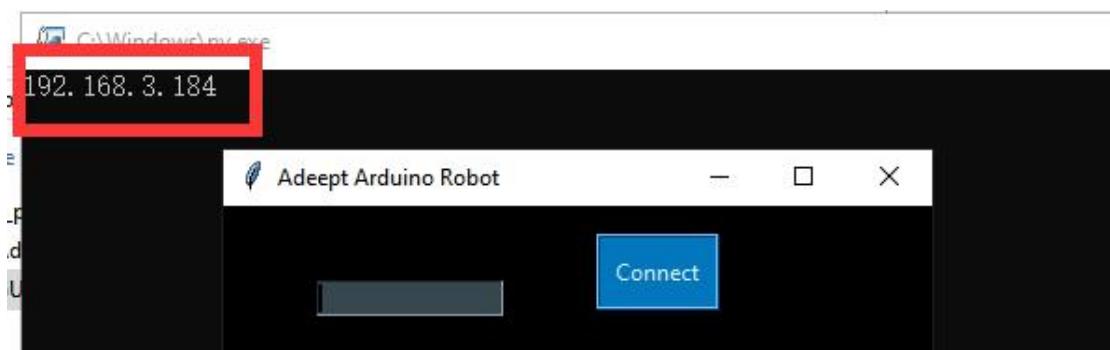
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



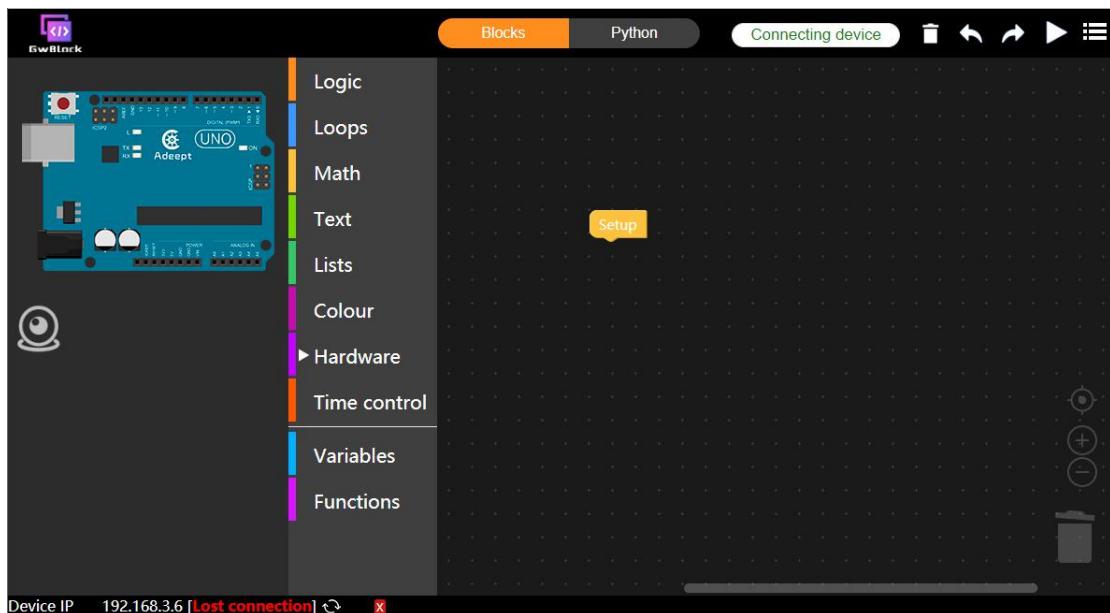
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



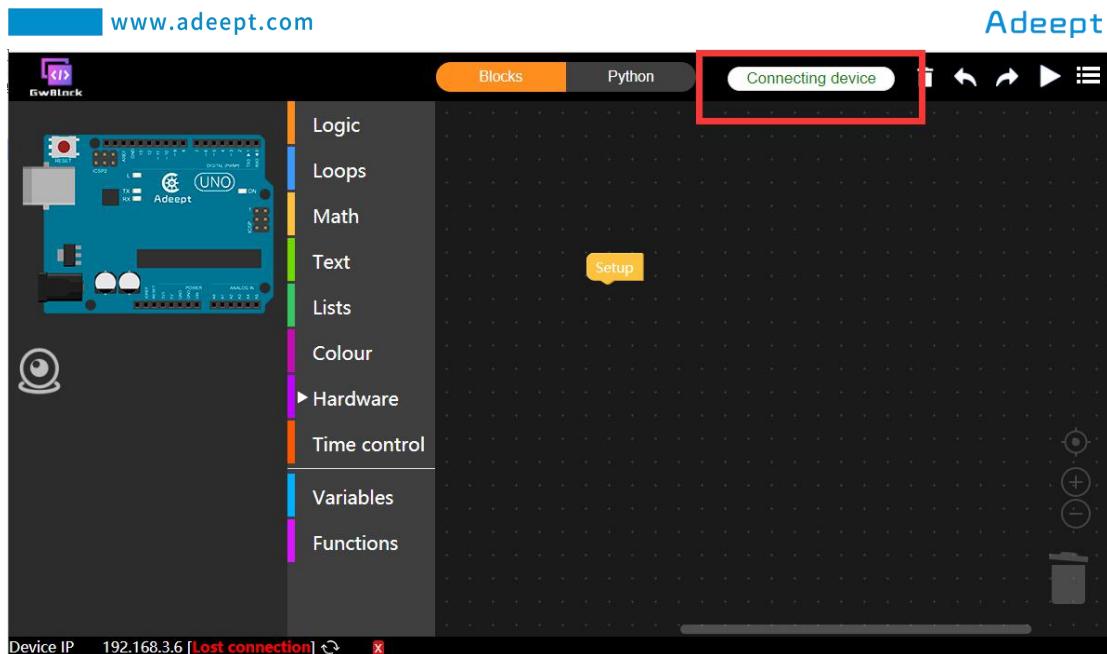
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

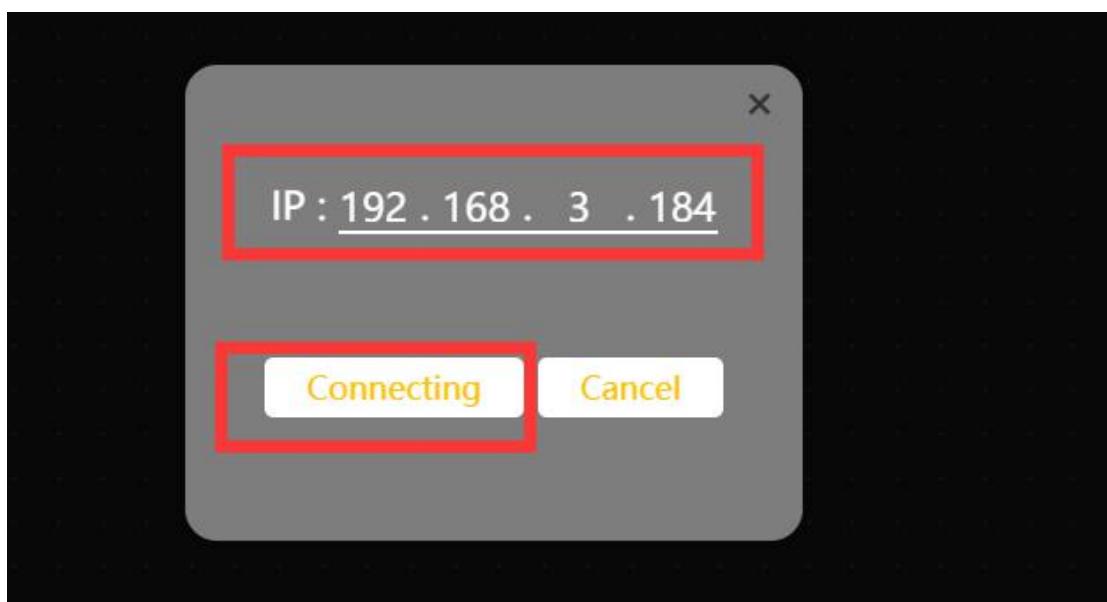
After successfully entering the website, the interface is as follows:



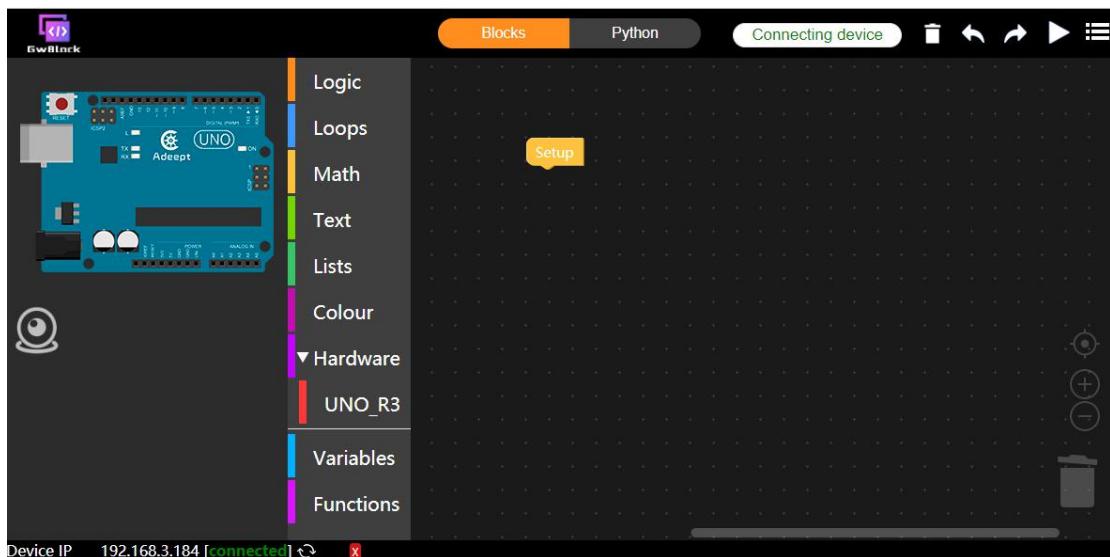
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

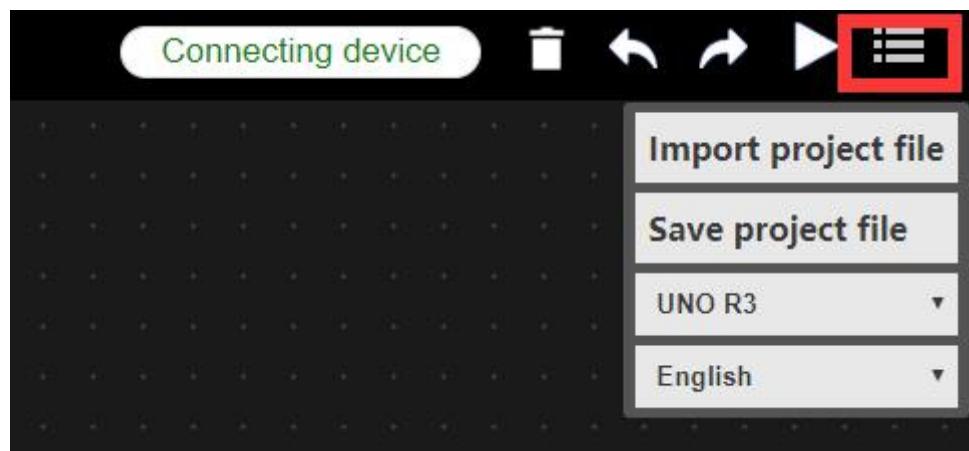


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

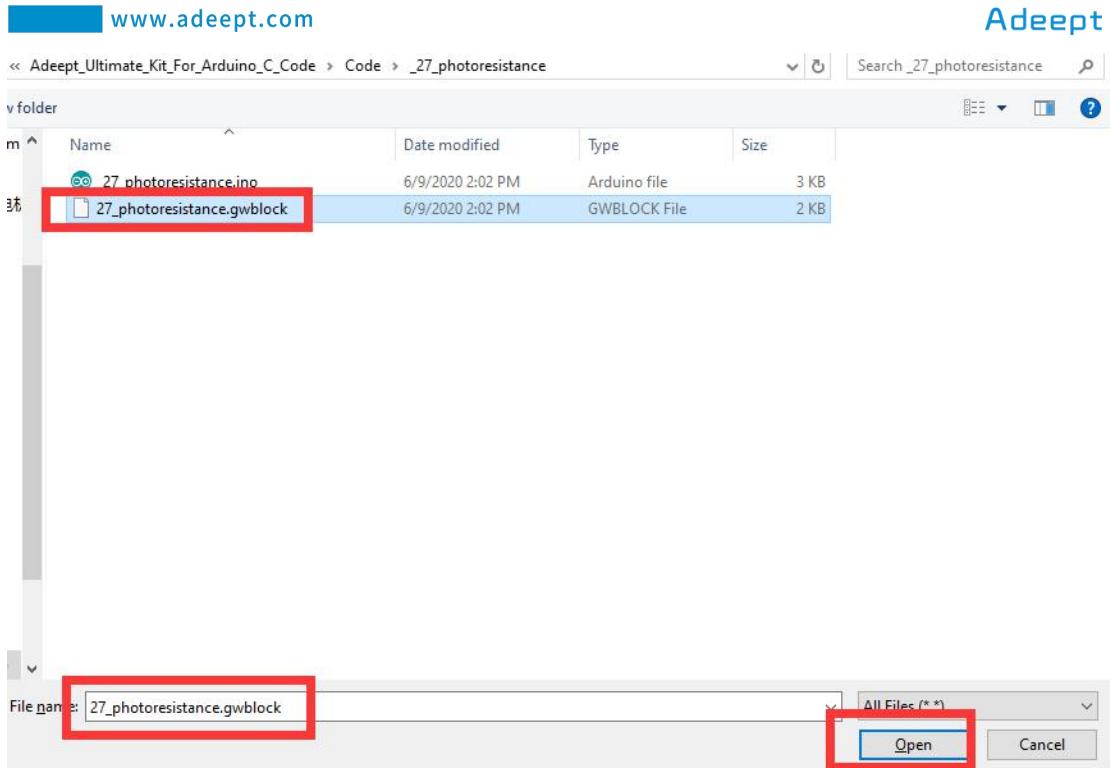
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_27_photoresistance

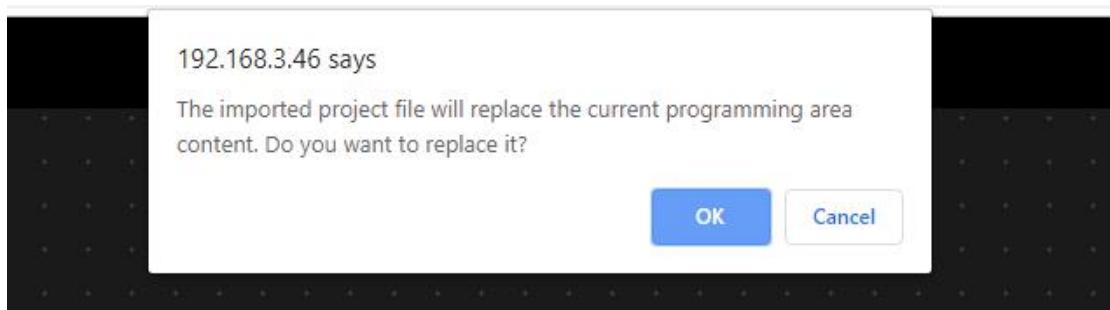
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

5. Select the "27_photoresistance.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



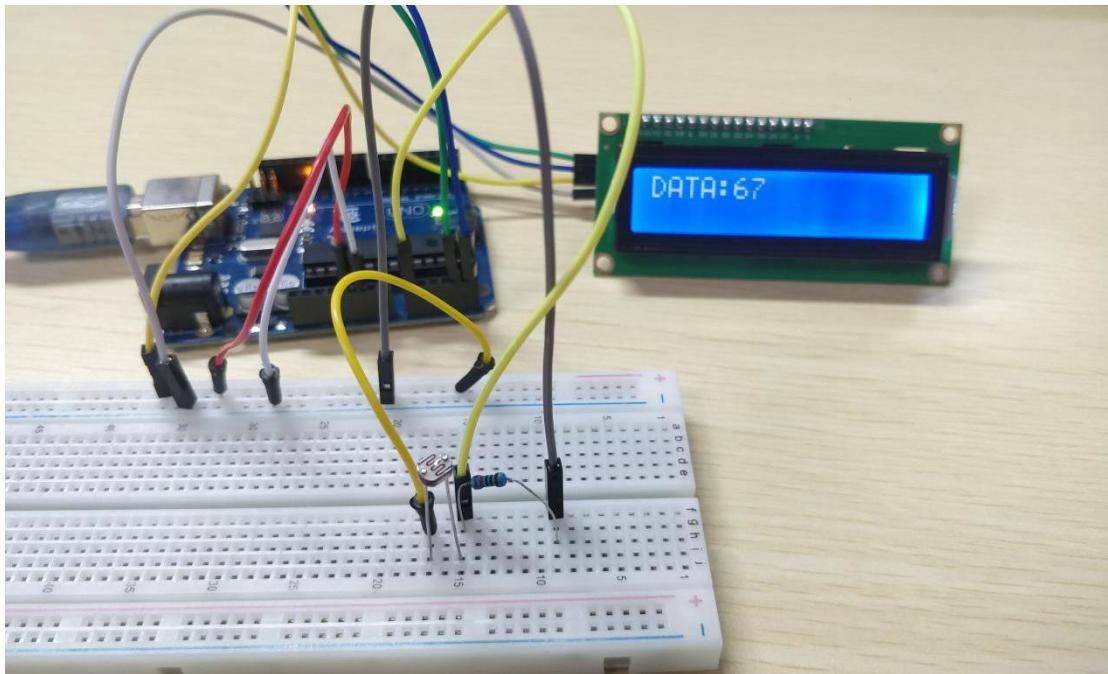
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



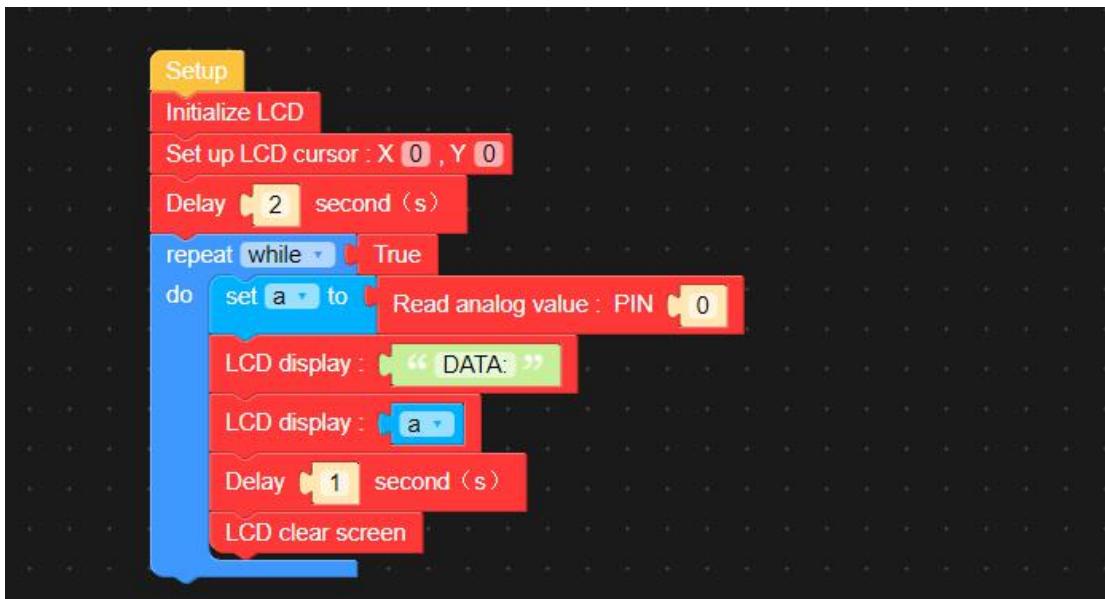
8. Click the button ▶ on the upper right corner. After successfully running the program, you will observe that the light intensity detected by the photoresistor will be displayed on the LCD1602 screen. The physical connection diagram of the experiment is as follows:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to use Photoresistor to measure light intensity on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

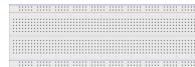
In the GwBlock graphical editor, all code programs are executed from **Setup**. Read the **photoresistor** data with the **instruction** block **set a to [Read analog value : PIN 0]**. Display the photoresistor data on the LCD1602 screen with **LCD display : [a]**.



Lesson 28 Making Light Tracking System

In this lesson, we will learn how to use photoresistance and servo motor to make light tracking system.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
LCD1602	1	
Adeept IIC Module	1	
Male to Female Jumper Wires	Several	
Resistor(1KΩ)	1	
Photoresistor	1	
AD002 Servo	1	

2. Photoresistance and servo motor

(1)photoresistance

Please refer to Lesson 27, for we have introduced the Photoresistance in detail in Lesson 27.

(2)servo motor

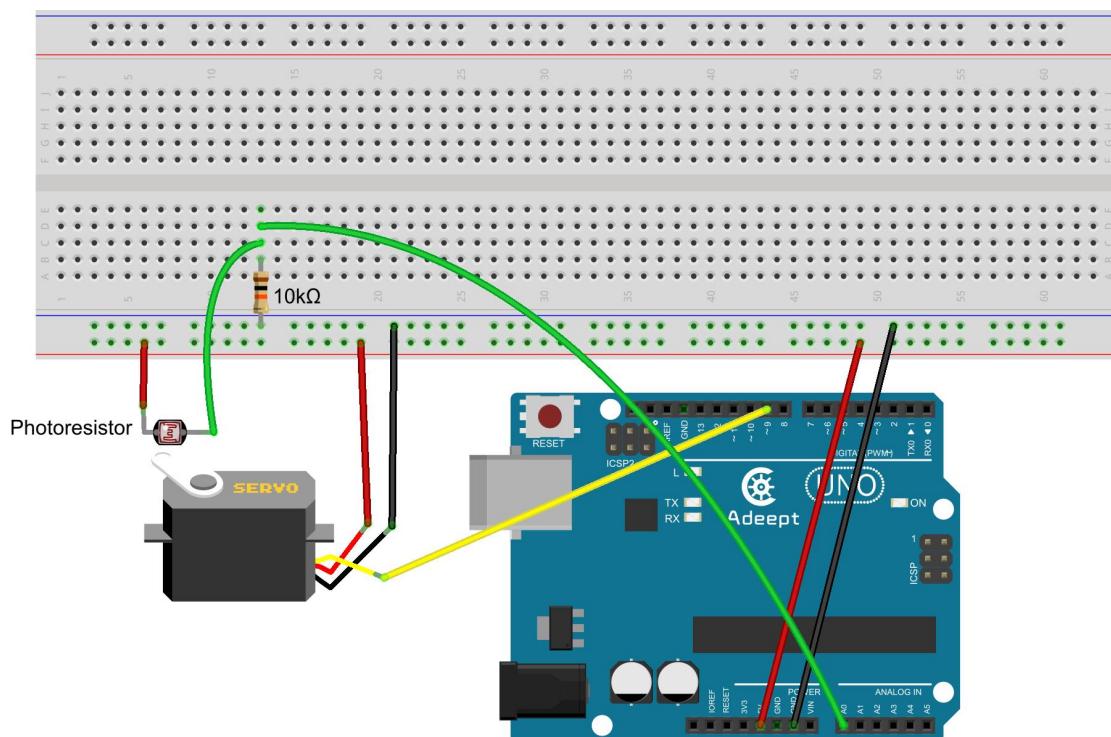
Please refer to Lesson 15, for we have introduced the servo motor in detail in Lesson 15.

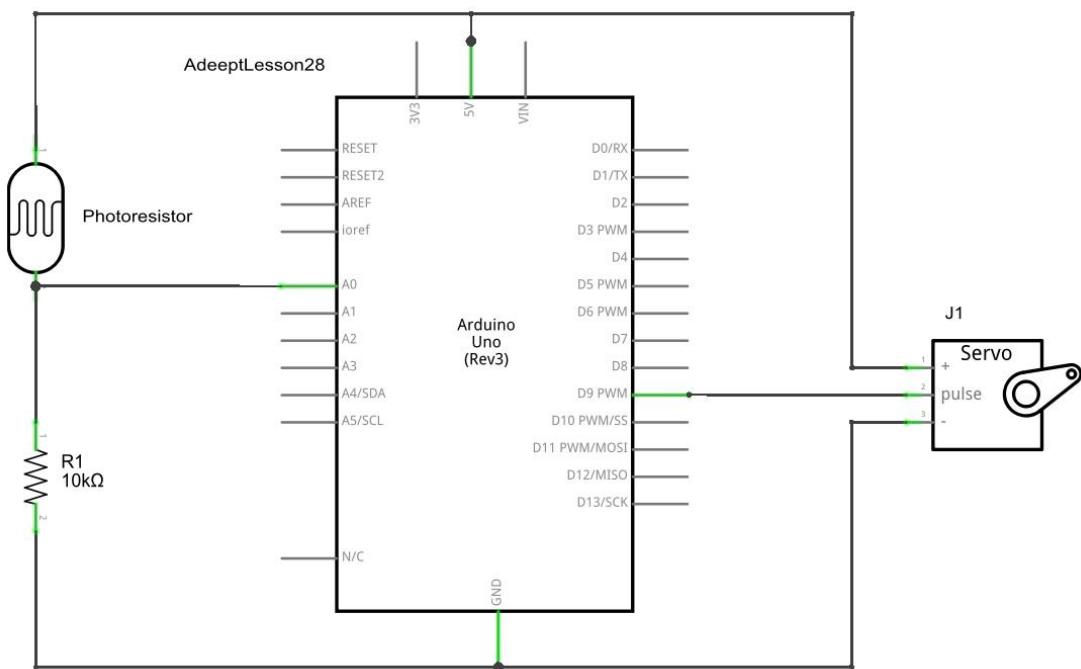
(3)Experimental principle

In this experiment, we need to fasten the photoresistor with the horn of servo. First, we control the servo with a photoresistor to rotate from 0° to 180° to record the intensity of illumination, and then the photoresistor will stop at the brightest position.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:





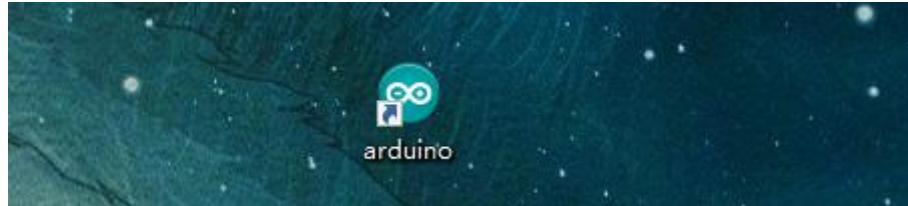
4. How to make light tracking system.

We provide two different methods to use photoresistance and servo motor to make light tracking system. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

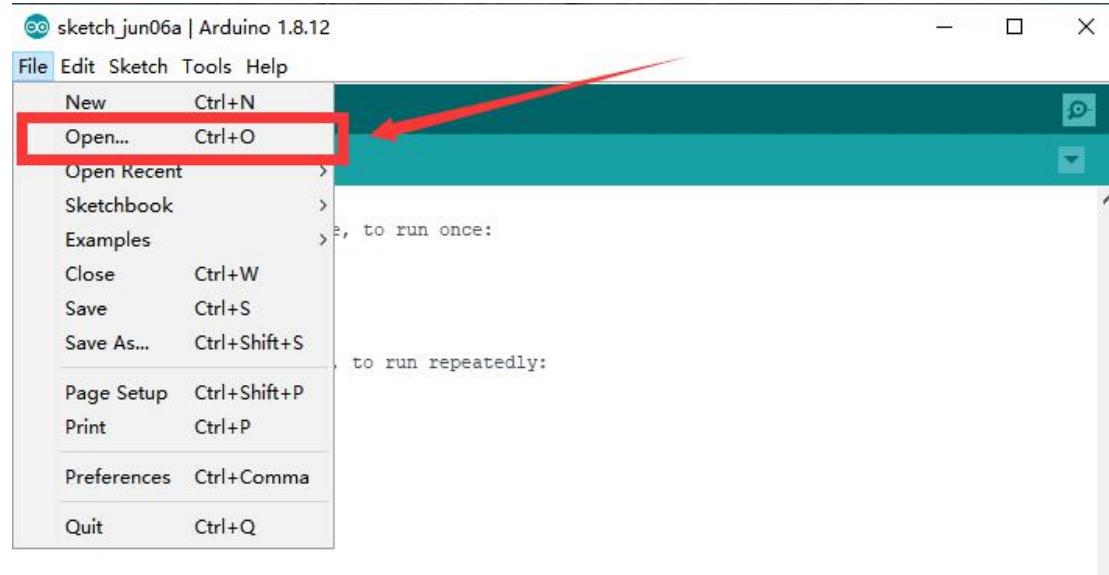
1. Using C language to program to make light tracking system on Arduino UNO

(1) Compile and run the code program of this course

1. Open the Arduino IDE software, as shown below:

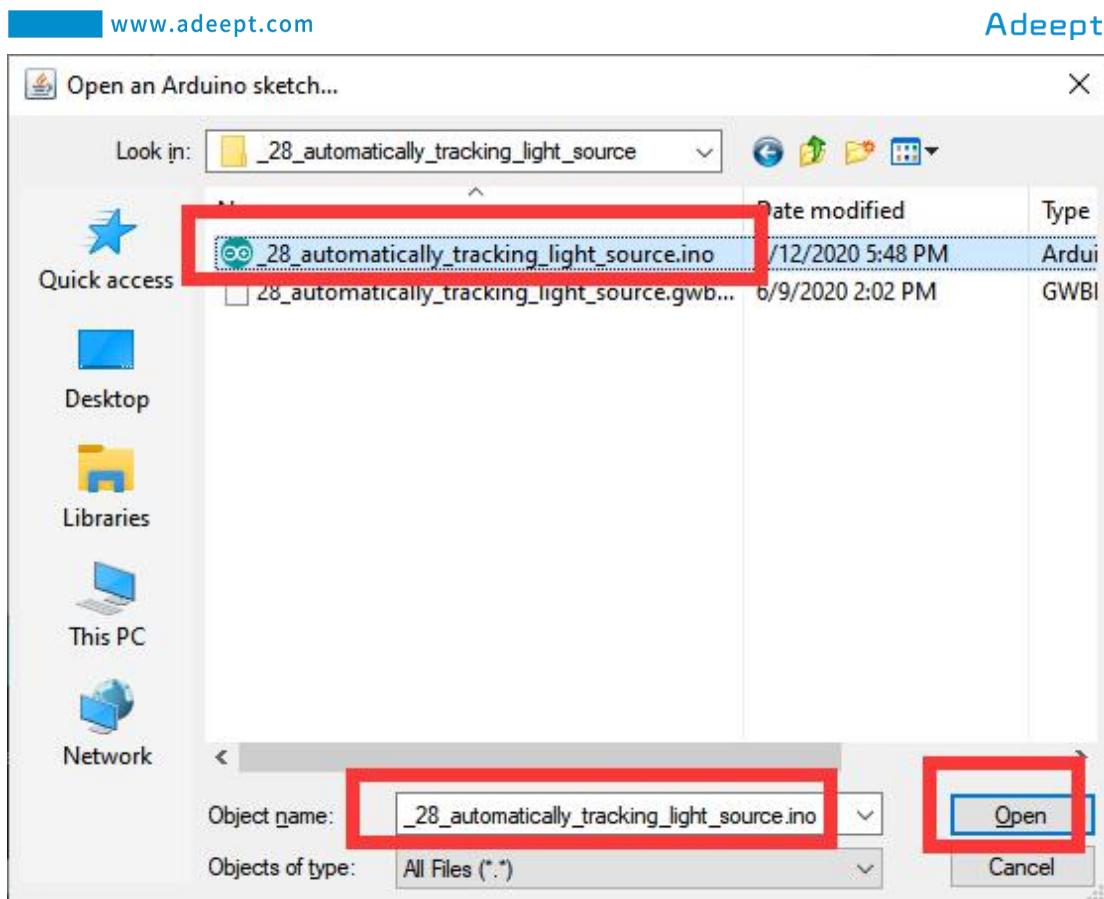


2.Click Open in the File drop-down menu:



3.Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\28 Automatically Tracking Light Source directory.

Select 28_Automatic Tracking Light Source.ino. This file is the code program we need in this course. Then click Open.



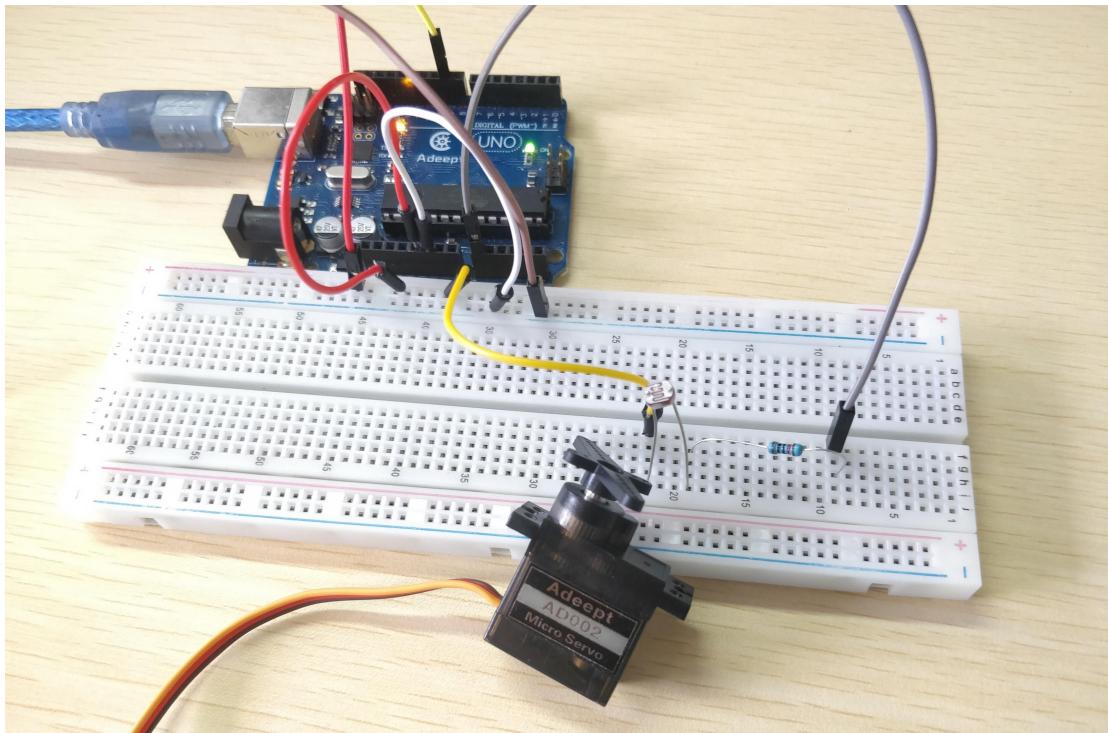
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After successfully running the program, you will observe that the servo motor will rotate. If you change the light intensity, for example, you can cover the photoresistor with a black cloth to see if the servo motor will rotate. The physical connection diagram of the experiment is as follows:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to use photoresistance and servo motor to make light tracking system on Arduino UNO. We will introduce how our core code can be achieved:

1.In the setup() function, bind the digital port 9 on the Arduino UNO by the myservo.attach(9) function.

```
void setup()
{
    myservo.attach(9); //attaches the servo on pin 9 to servo object
}
```

2.In the loop() function, read the data of the photoresistor by analogRead(photocellPin), and then control the rotation of the servo motor by the myservo.write(outputValue) function.

```
void loop()
{
    outputValue = analogRead(photocellPin);

    outputValue %=180;

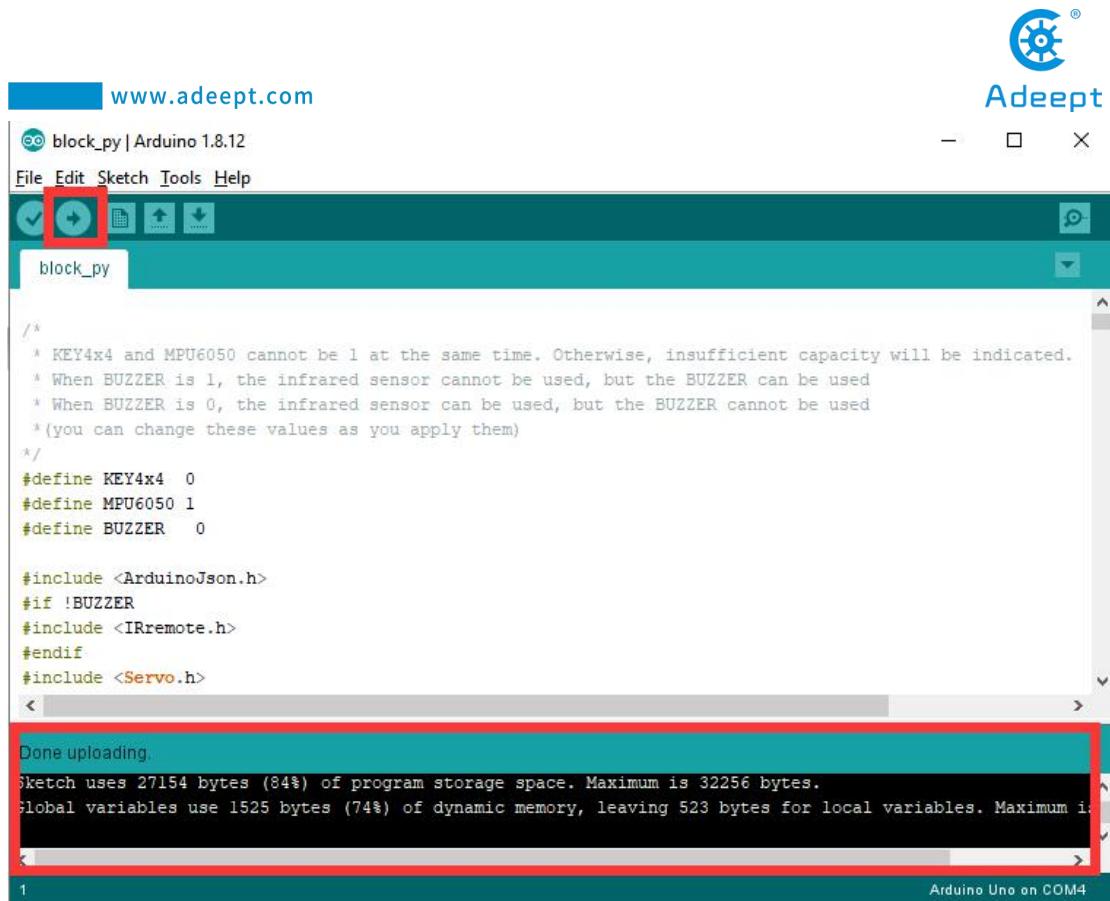
    myservo.write(outputValue);
    delay(1000);
}
```

2. Using graphical code blocks to program to make light tracking system on Arduino UNO

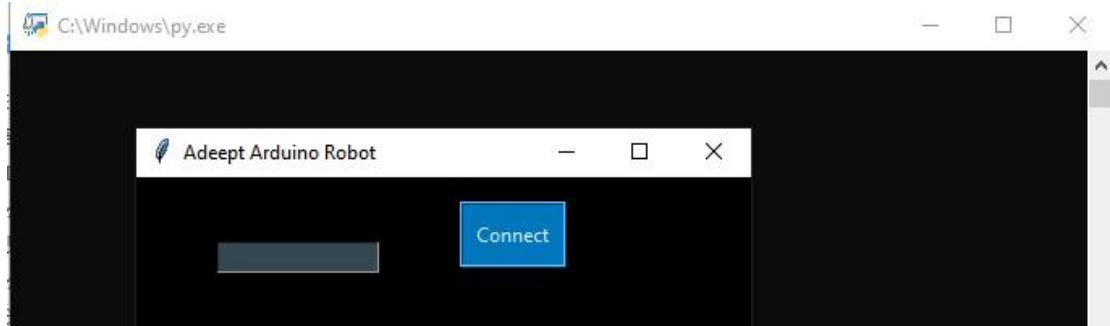
(1) Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

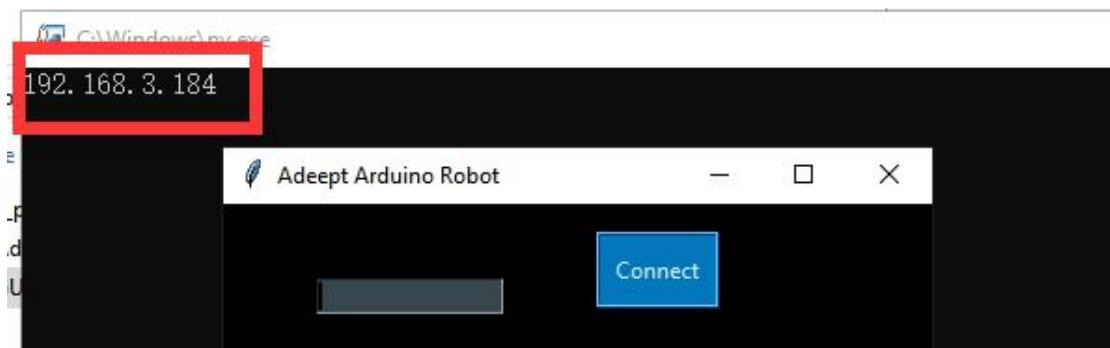
1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:



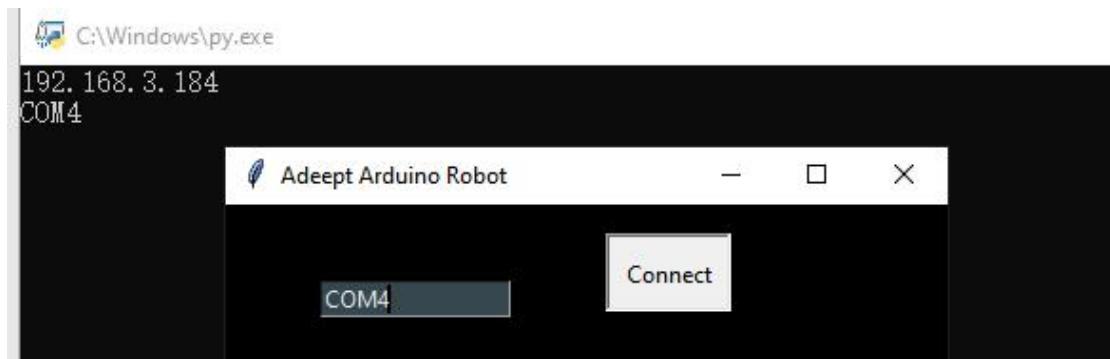
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3.Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



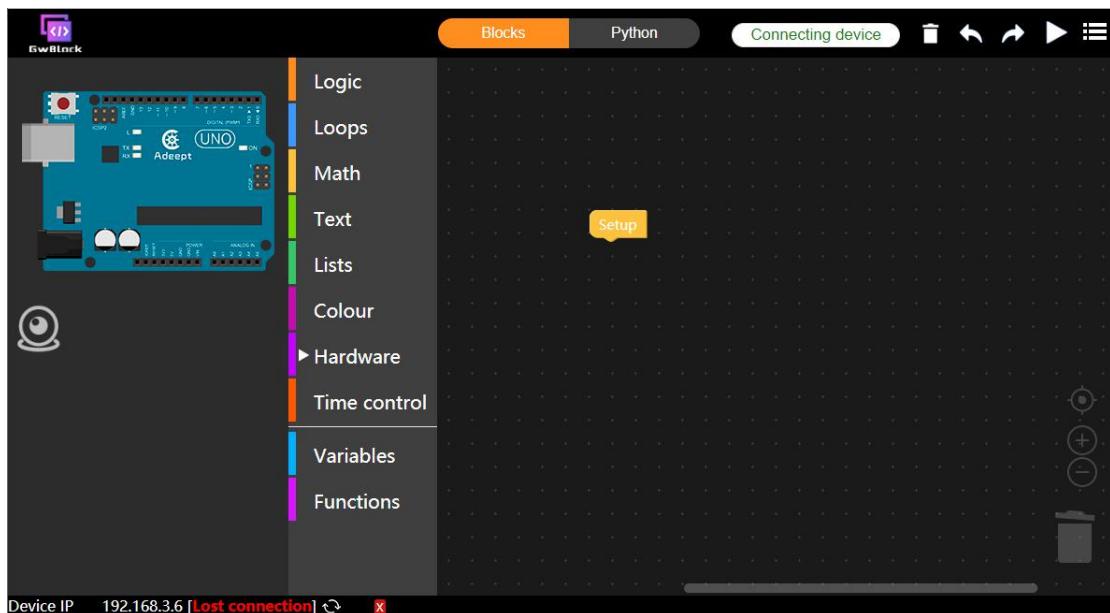
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



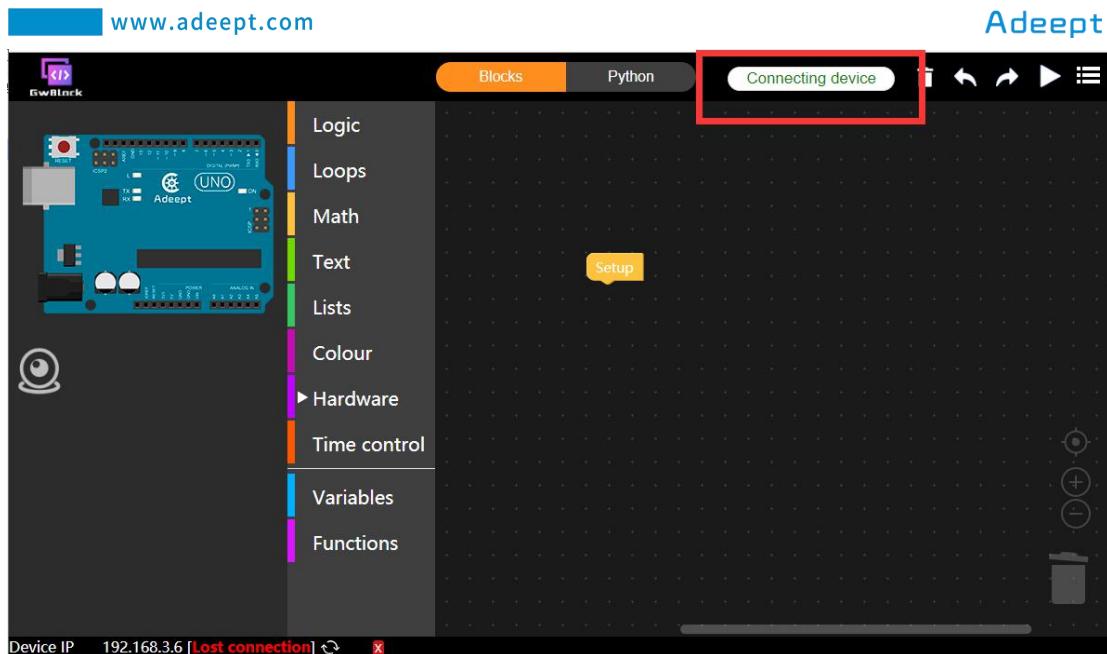
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

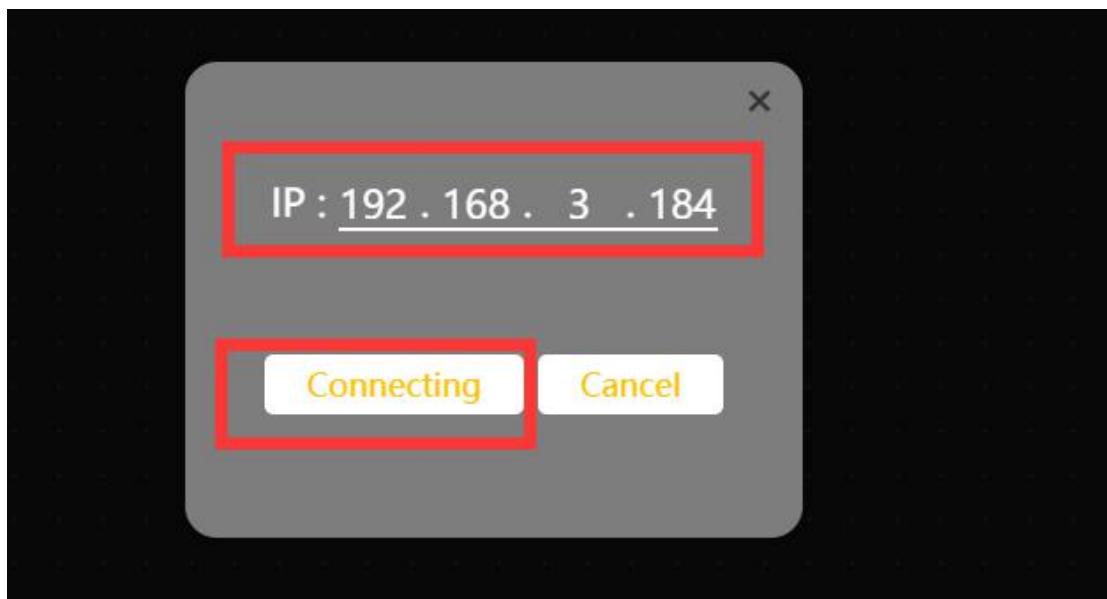
After successfully entering the website, the interface is as follows:



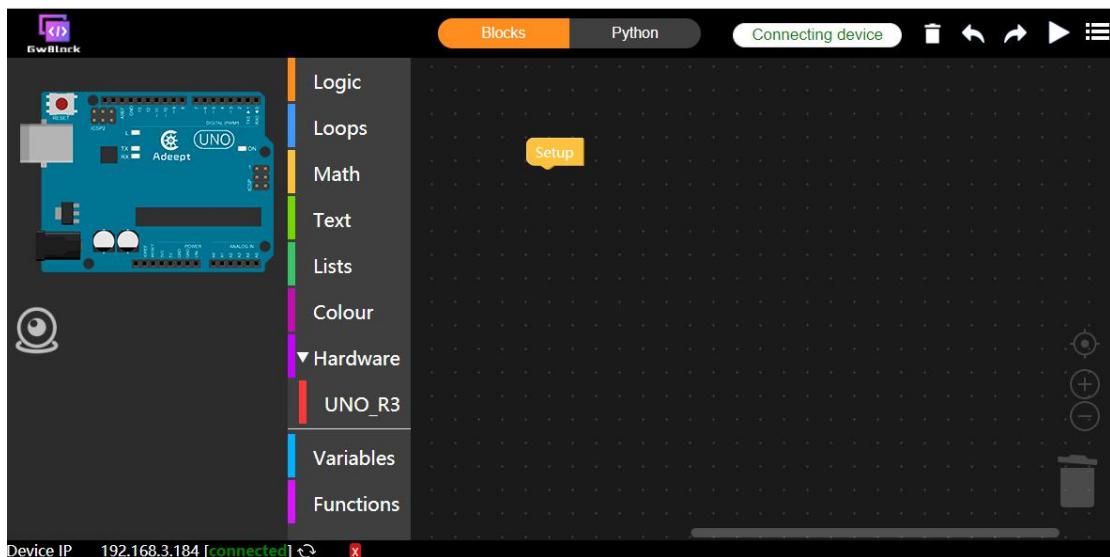
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:



(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

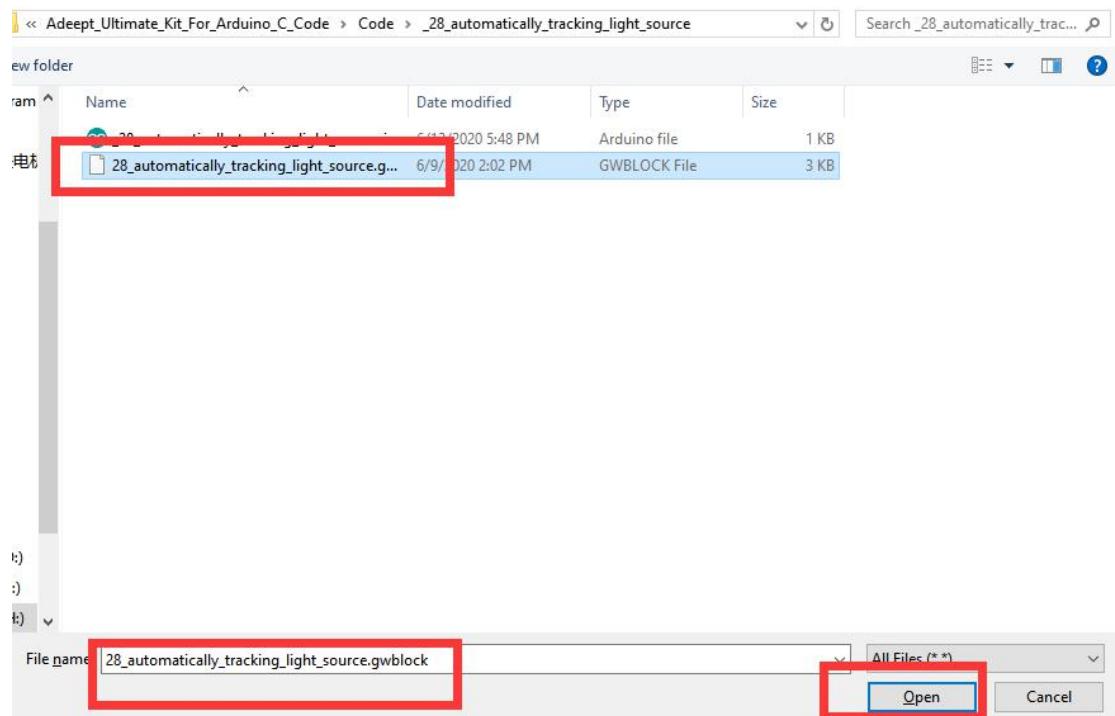
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_28 Automatically_tracking_light_source

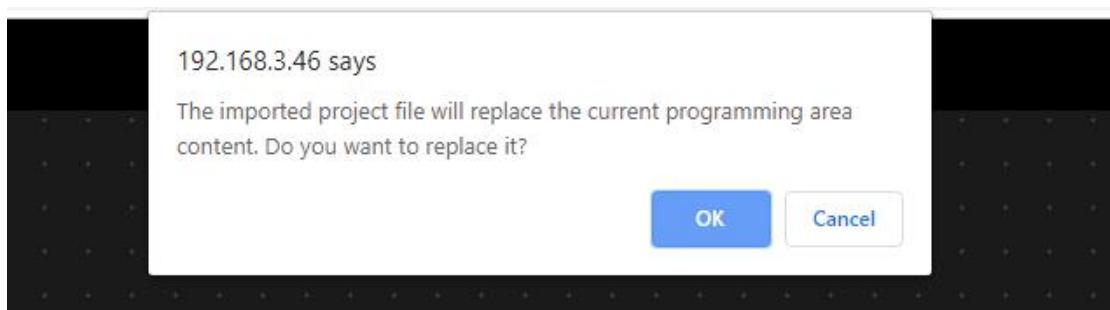
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

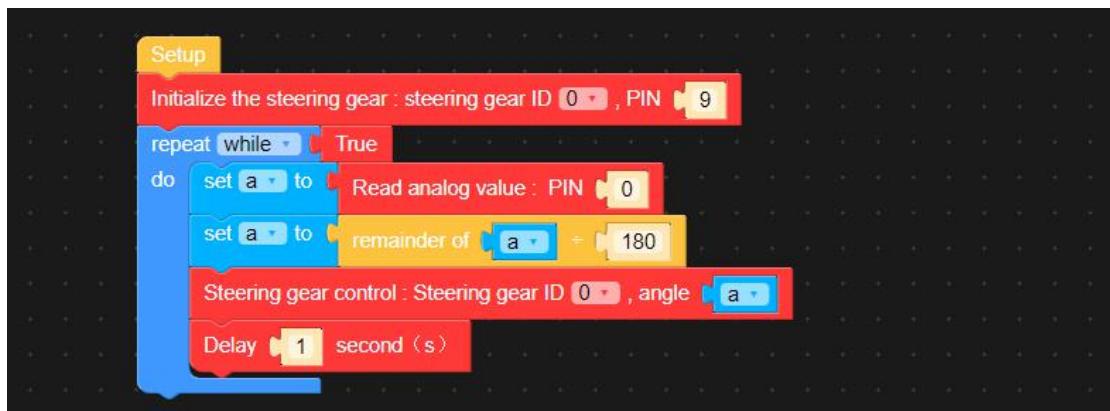
5. Select the "28_Automatic Tracking Light Source.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



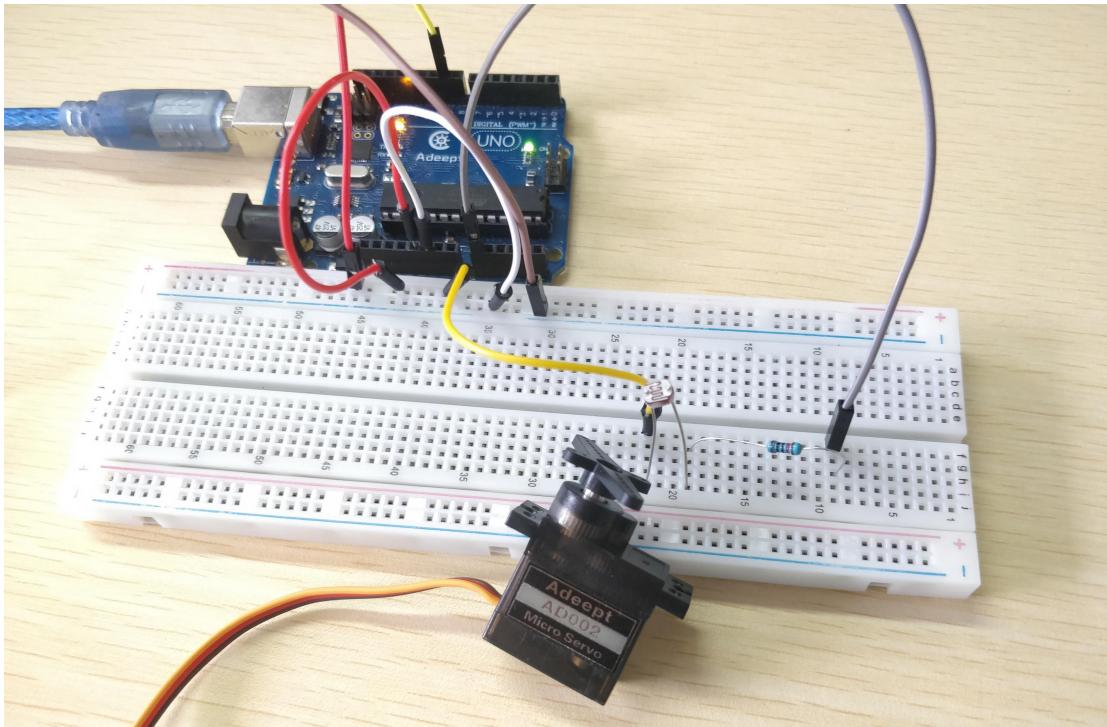
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



8. Click the button  on the upper right corner.After successfully running the program, you will find that the servo motor will rotate. The physical connection diagram of the experiment is as follows:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to use photoresistance and servo motor to make light tracking system on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

In the GwBlock graphical editor, all code programs are executed from **Setup**. Initialize the 9th pin with the command **Initialize the steering gear : steering gear ID 0 , PIN 9**. Read the data of the photoresistor with **set a to Read analog value : PIN 0**. Control the rotation of the servo motor with the command block **Steering gear control : Steering gear ID 0 , angle a**.



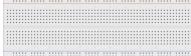
The Scratch script consists of the following blocks:

- Setup:**
 - Initialize the steering gear : steering gear ID [0], PIN [9]
- repeat while [True]**
 - do** [set [a] to [Read analog value : PIN [0]]]
 - set [a] to [remainder of [a] ÷ [180]]**
 - Steering gear control : Steering gear ID [0], angle [a]**
 - Delay [1] second (s)**

Lesson 29 Making Frequency Meter

In this lesson, we will learn how to use potentiometer and NE555 timer to make frequency meter.

1. Components used in this course

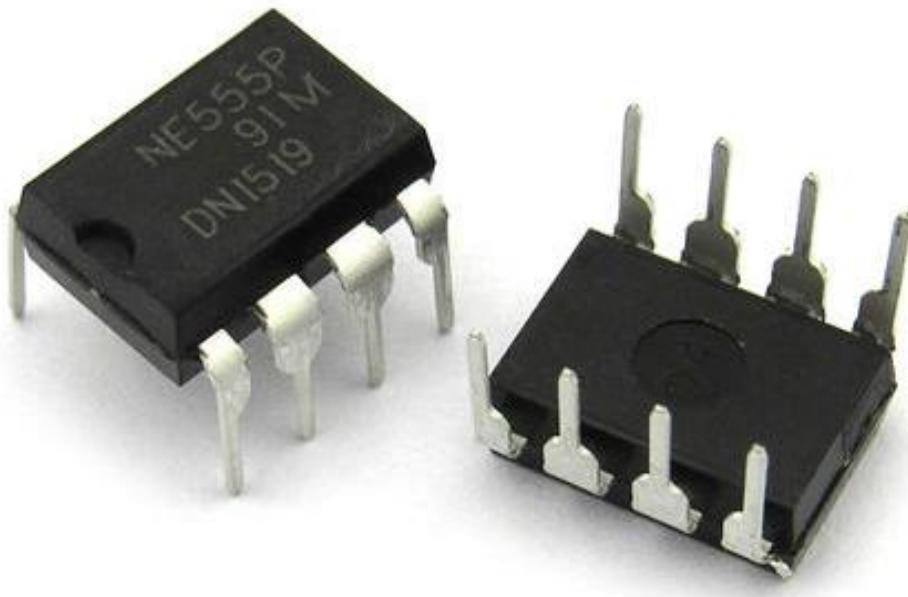
Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Male to Female Jumper Wires	Several	
Potentiometer	1	
NE555 Timer	1	
Resistor(10KΩ)	1	
Capacitor(104)	1	

2.NE555 Timer

(1)NE555 Timer

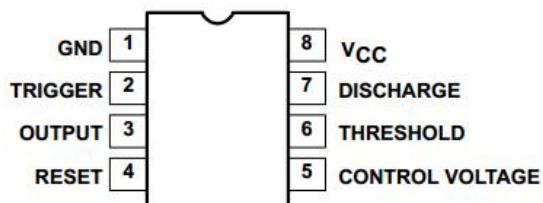
The 555 integrated circuit is originally used as a timer, and that is why it is called 555 timer or 555 time-based circuit. It is widely used in various electronic products because of its reliability, convenience and low price. There are dozens of components in the 555 integrated circuit, such as divider, comparator, basic R-S trigger, discharge

tube, and buffer. This is a complex circuit and a hybrid composed of analog and digital circuits.



(1)Working principle

As shown in the above figure, the 555 integrated circuit is dual in-line with 8 pins package(DIP).



Pin 6 is the THRESHOLD for the input of upper comparator;

Pin 2 is TRIGGER for the input of lower comparator;

Pin 3 is the OUTPUT having two states of 0 and 1 decided by the input electrical level;

Pin 7, the DISCHARGE which has two states of suspension and ground connection also decided by input, is the output of the internal discharge tube;

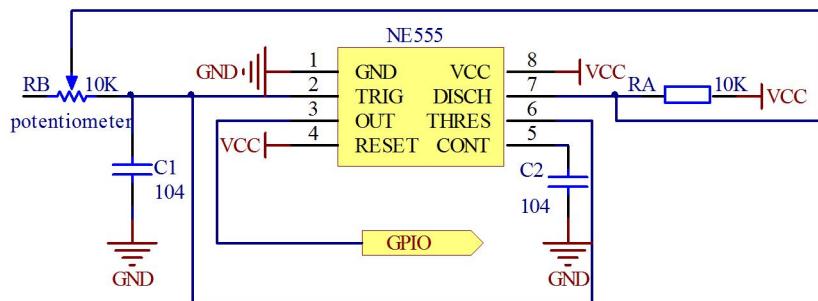
Pin 4 is the RESET that outputs low level when supplied low voltage level;

Pin 5 is the CONTROL VOLTAGE that can change the upper and lower level trigger value;

Pin 8 (Vcc) is the power supply;

Pin 1(GND) is the ground.

The schematic diagram used in the experiment is as shown below:

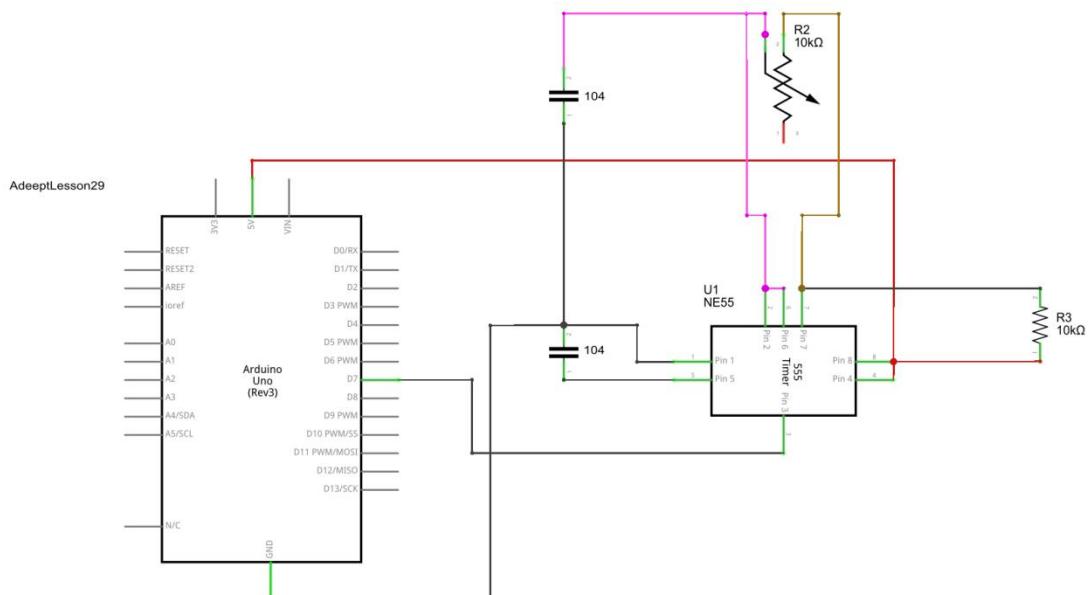
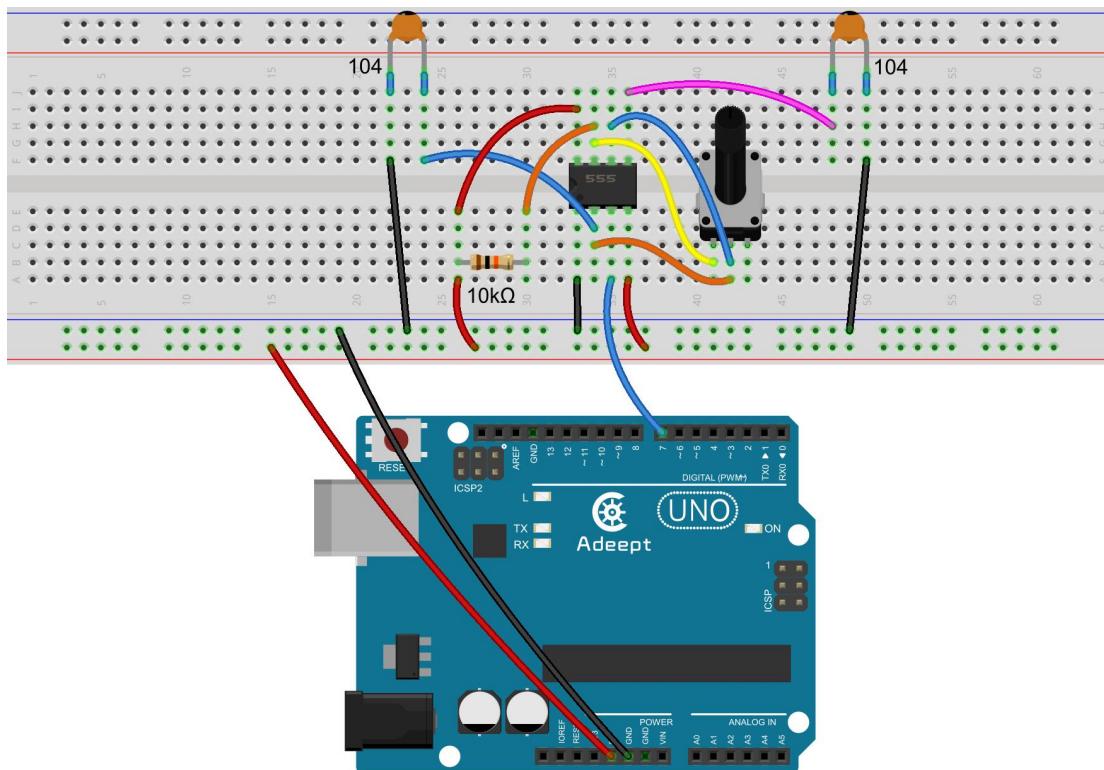


The circuit can generate a square wave signal that the frequency is adjustable.
The frequency can be calculated by the formula:

$$\text{Frequency} = \frac{1.44}{(R_A + 2R_B) * C}$$

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:



4. Making Frequency Meter

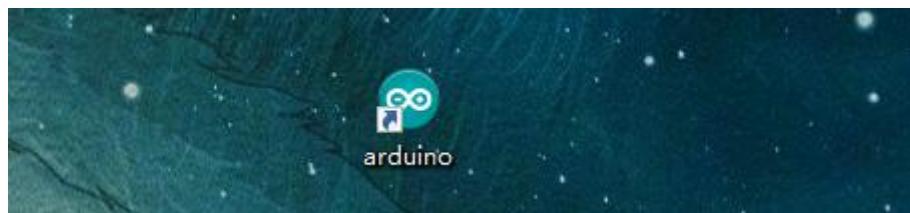
We provide two different methods to use Potentiometer and NE555 Timer to make Frequency Meter. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program

on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

1. Using C language to program to make Frequency Meter on Arduino UNO

(1) Compile and run the code program of this course

1. Open the Arduino IDE software, as shown below:

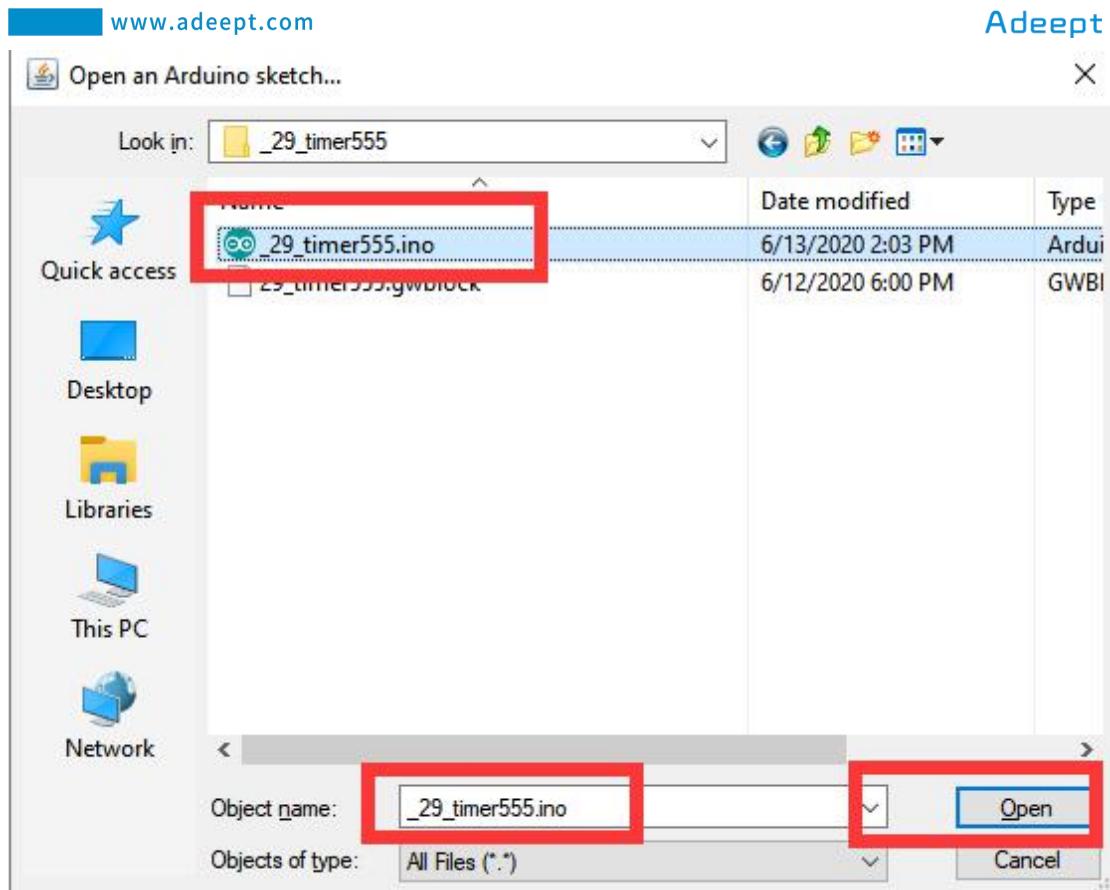


2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\29_timer555 directory.

Select 29_timer555.ino. This file is the code program we need in this course. Then click Open.



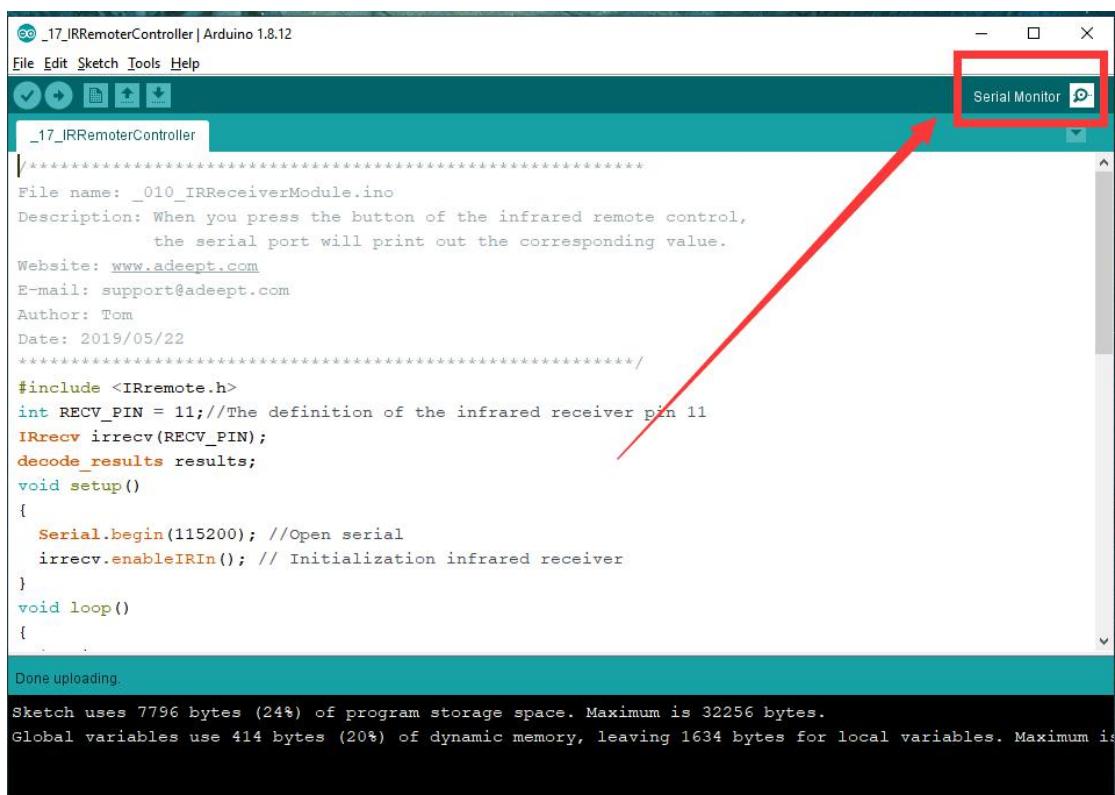
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After successfully running the program, we need to open the serial monitor on the Arduino software. How to open the serial monitor? You need to click the "Serial Monitor" button  in the upper right corner, as shown below:



The screenshot shows the Arduino IDE interface with the sketch `_17_IIRemoteController` open. The serial monitor button in the top right corner is highlighted with a red box and an arrow pointing to it. The code in the editor is for an infrared remote controller, and the serial monitor window shows the uploaded sketch details.

```

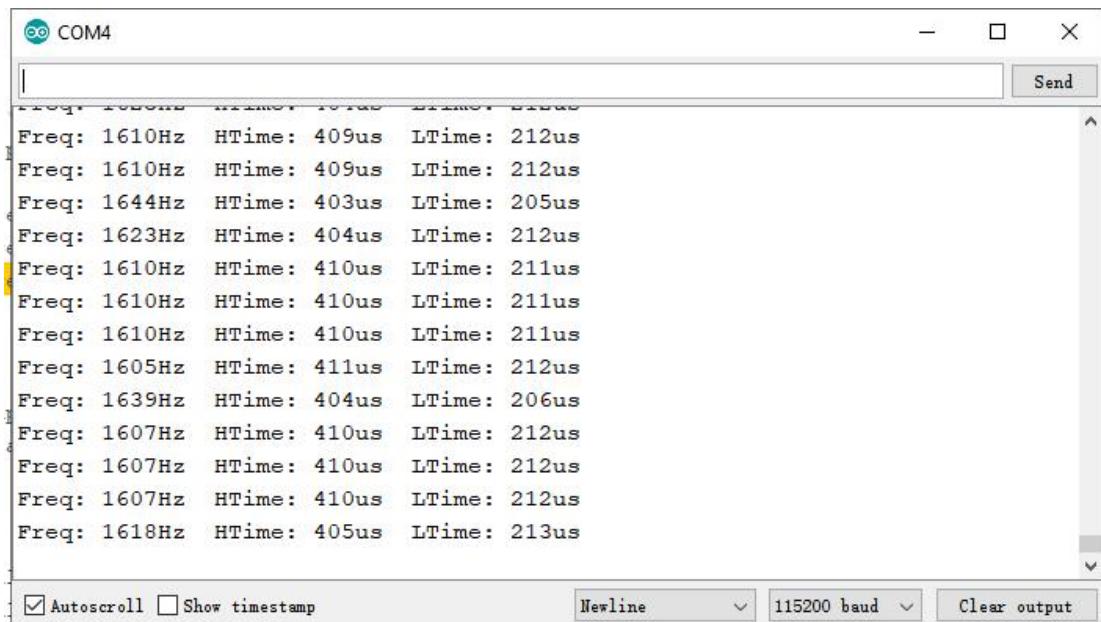
_17_IIRemoteController | Arduino 1.8.12
File Edit Sketch Tools Help
Serial Monitor
_17_IIRemoteController
File name: _010_IRReceiverModule.ino
Description: When you press the button of the infrared remote control,
the serial port will print out the corresponding value.
Website: www.adeept.com
E-mail: support@adeept.com
Author: Tom
Date: 2019/05/22
*****
#include <IRremote.h>
int RECV_PIN = 11;//The definition of the infrared receiver pin 11
IRrecv irrecv(RECV_PIN);
decode_results results;
void setup()
{
    Serial.begin(115200); //Open serial
    irrecv.enableIRIn(); // Initialization infrared receiver
}
void loop()
{
}

Done uploading.

Sketch uses 7796 bytes (24%) of program storage space. Maximum is 32256 bytes.
Global variables use 414 bytes (20%) of dynamic memory, leaving 1634 bytes for local variables. Maximum is

```

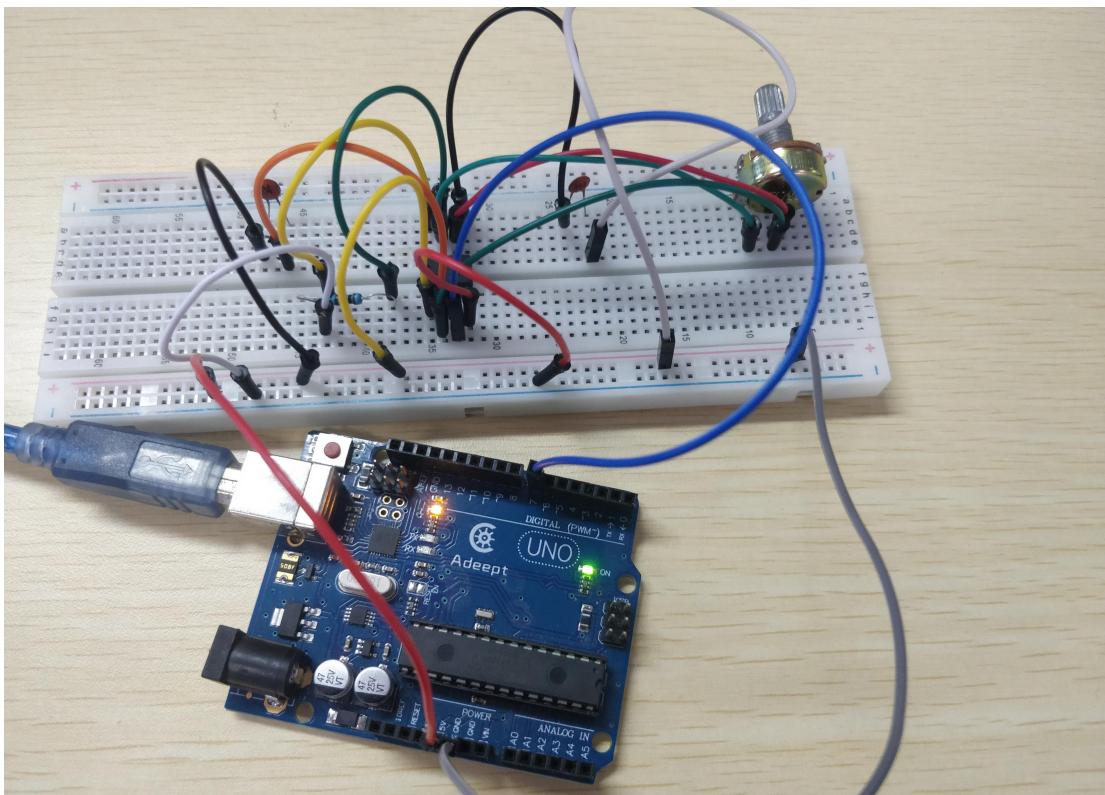
6. After clicking  , the serial monitor window will pop up.Rotating the potentiometer, the serial monitor will print out the corresponding value, as shown below:



The screenshot shows the Serial Monitor window titled "COM4". It displays a series of infrared signal parameters (Freq, HTime, LTime) being printed. The parameters are listed as Freq: [value]Hz, HTime: [value]us, LTime: [value]us. The window includes standard serial monitor controls at the bottom: Autoscroll, Show timestamp, Newline, baud rate (set to 115200), and Clear output.

Freq	HTime	LTime
1610Hz	409us	212us
1610Hz	409us	212us
1644Hz	403us	205us
1623Hz	404us	212us
1610Hz	410us	211us
1610Hz	410us	211us
1610Hz	410us	211us
1605Hz	411us	212us
1639Hz	404us	206us
1607Hz	410us	212us
1607Hz	410us	212us
1607Hz	410us	212us
1618Hz	405us	213us

7. The physical connection diagram of the experiment is as follows:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to use Potentiometer and NE555 Timer to make Frequency Meter on Arduino UNO. We will introduce how our core code can be achieved:

1.In the setup() function, set the pin pin to INPUT mode with the pinMode() function.

```
void setup()
{
    pinMode(pin, INPUT);      //set the pin as an input
    Serial.begin(115200);    // start serial port at 115200 bps:
}
```

2.In the loop() function, after calculation, print the values of Freq, HTime, and LTime on the serial port of the Arduino software.

```

void loop()
{
    durationhigh = pulseIn(pin, HIGH); //Reads a pulse on pin
    durationlow = pulseIn(pin, LOW); //Reads a pulse on pin
    duration = 1000000/(durationhigh + durationlow);
    Serial.print("Freq: "); //print Freq:
    Serial.print(duration); //print the length of the pulse on the serial monitor
    Serial.print("Hz HTime: "); //print Hz HTime:
    Serial.print(durationhigh); //print the length of the pulse on the serial monitor
    Serial.print("us LTime: "); //print us LTime:
    Serial.print(durationlow); //print the length of the pulse on the serial monitor
    Serial.print("us ");
    Serial.println(); //print a blank on serial monitor
    delay(500); //wait for 500 microseconds
}

```

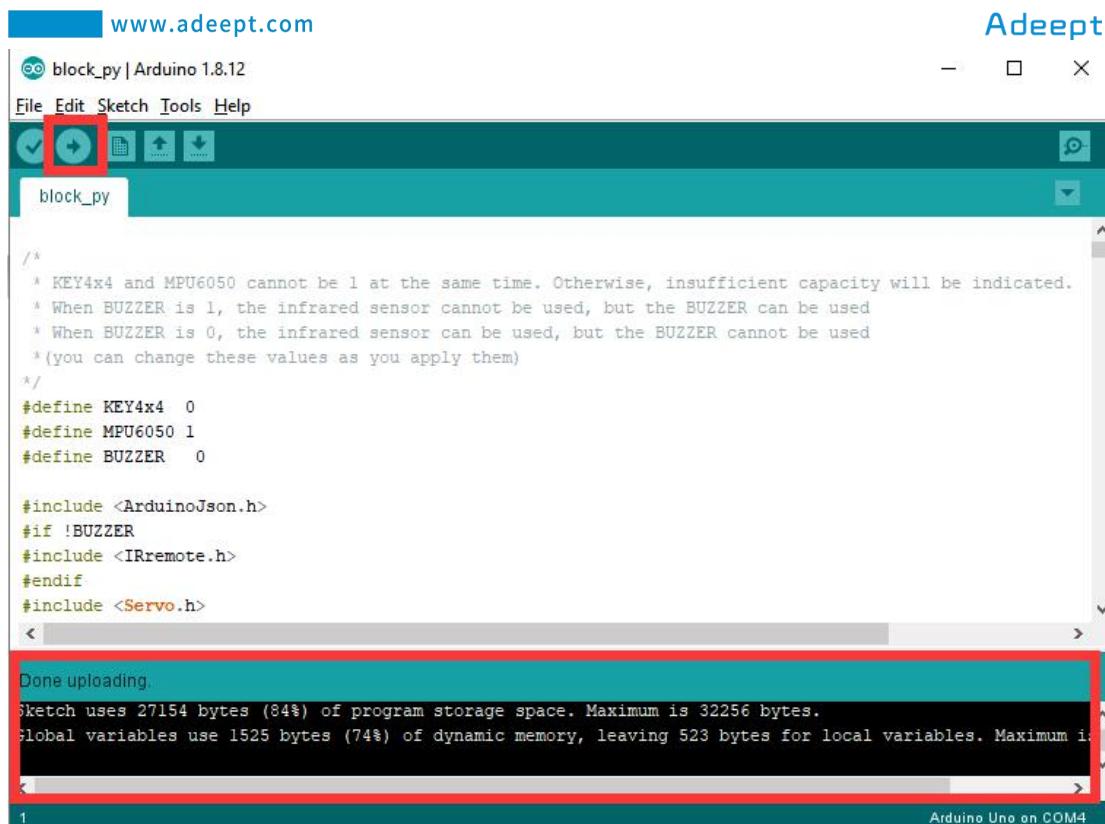
2. Using graphical code blocks to program to make Frequency

Meter on Arduino UNO

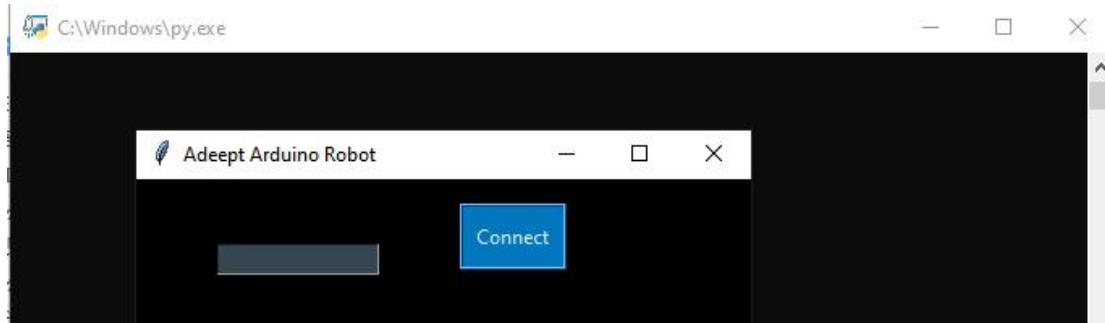
(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

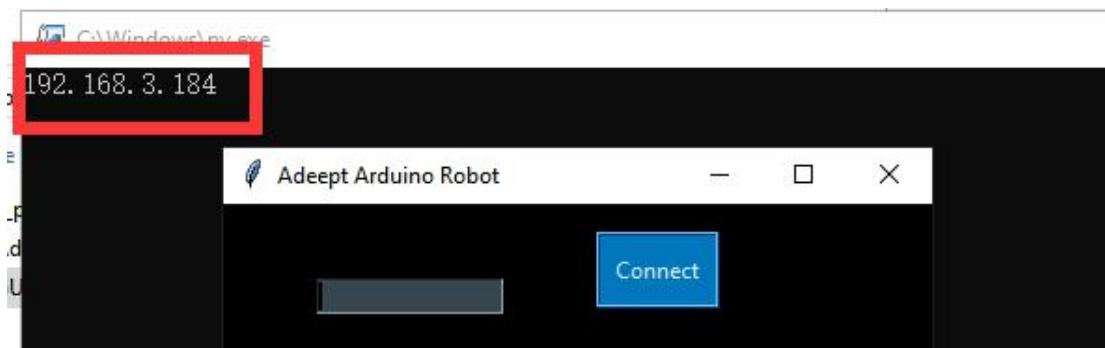
1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:



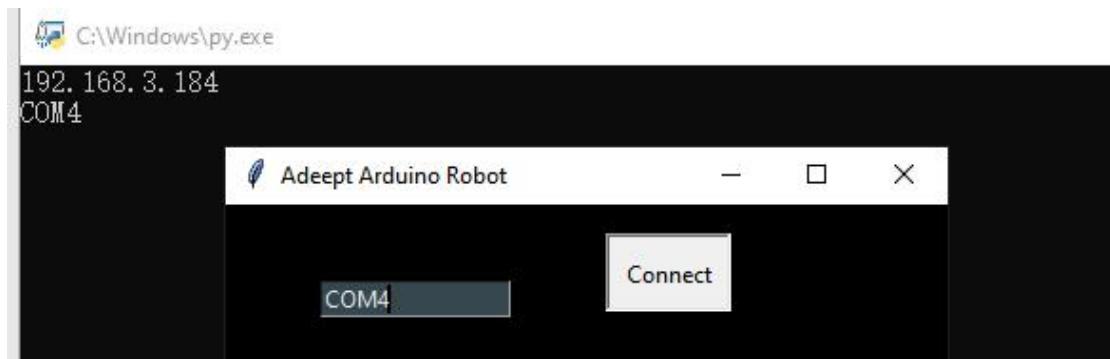
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3.Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



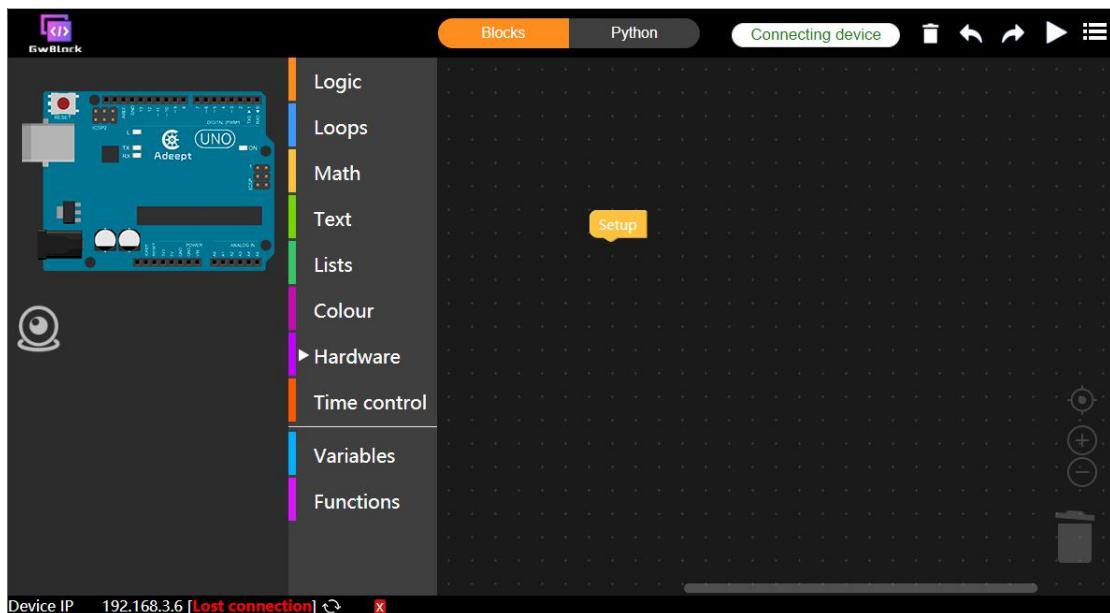
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



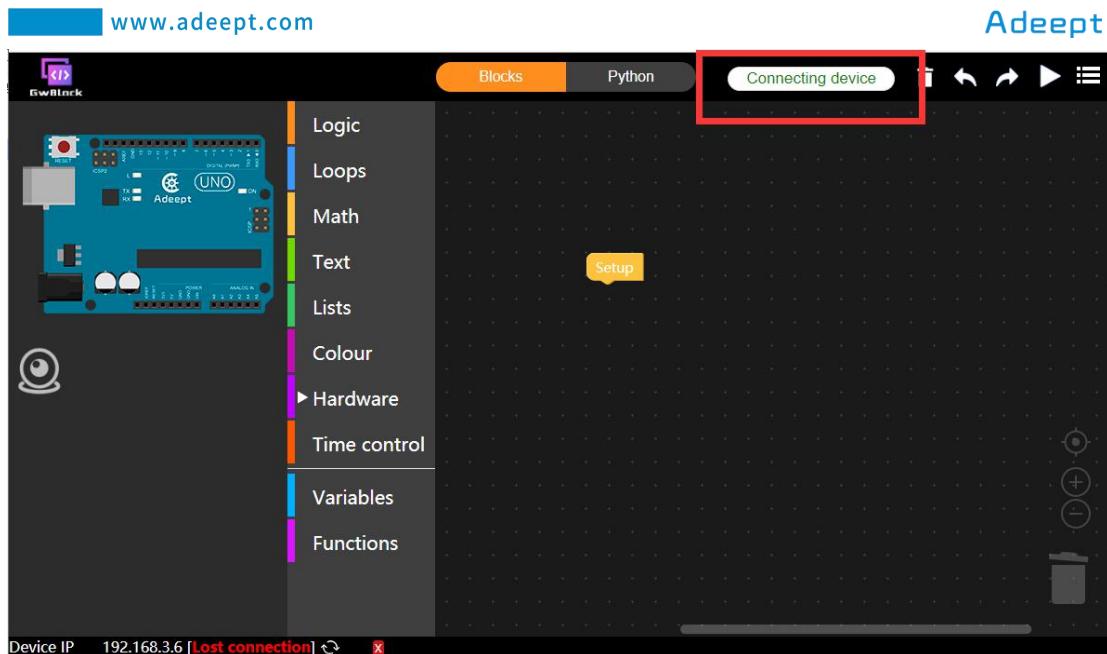
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

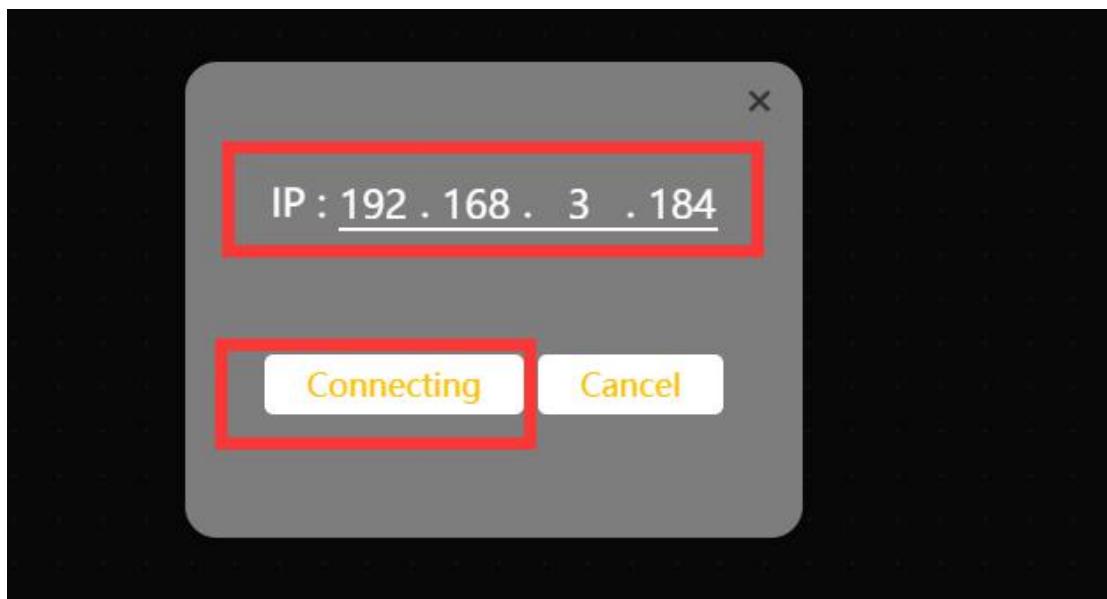
After successfully entering the website, the interface is as follows:



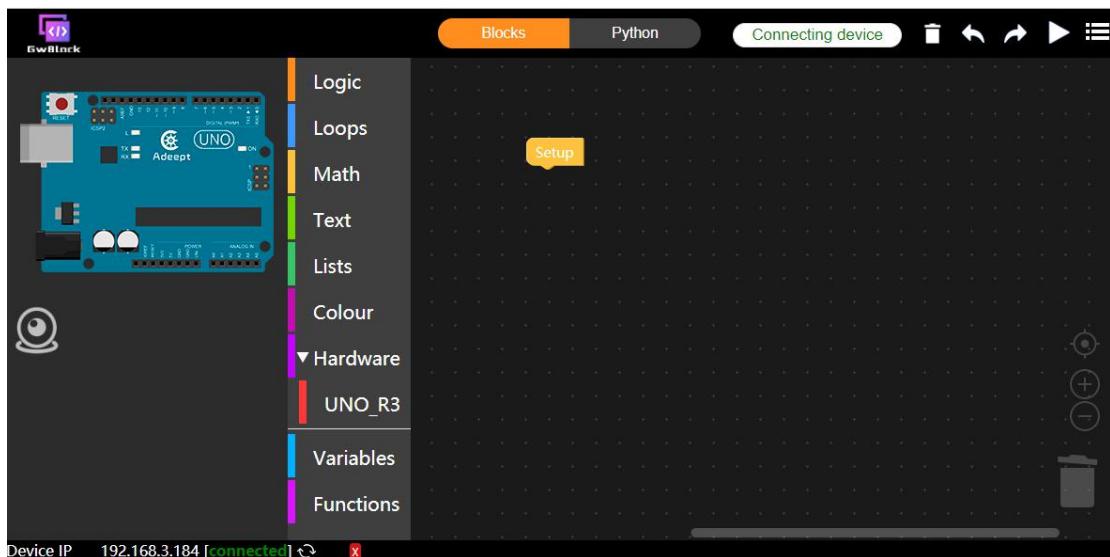
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:



(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

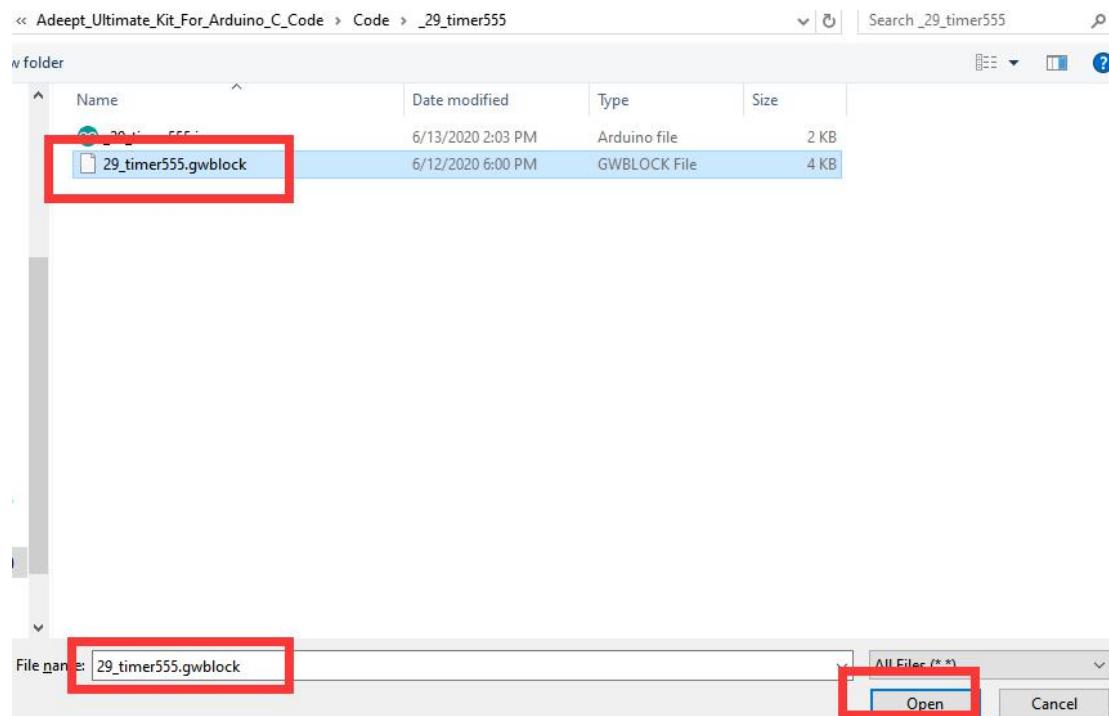
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_29_timer555

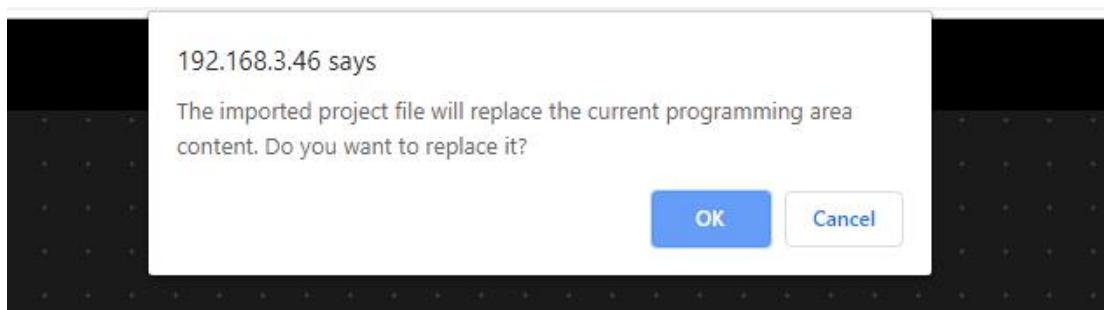
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

5. Select the "29_timer555.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



6.Click OK.It will show as below:

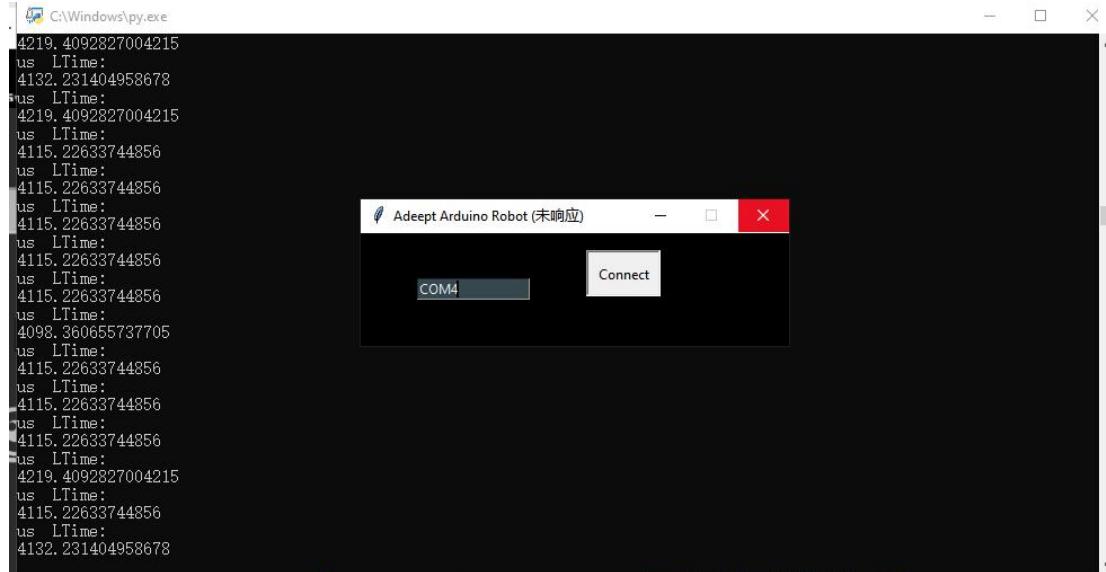


7.It will show as below after successfully opening:

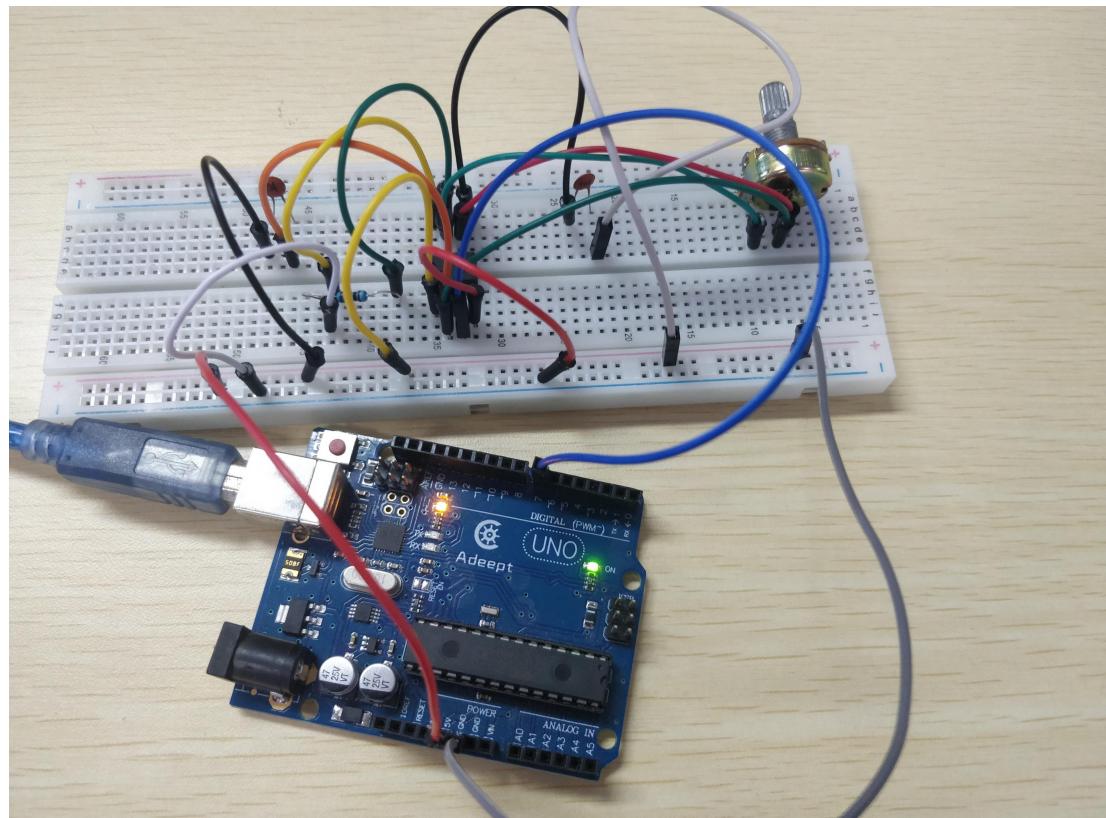


8. Click the button  on the upper right corner.After running the program

successfully, open the GUI info v1.0.py window. When you rotate the potentiometer, you will find that the corresponding value will also be displayed on the window, indicating that our experimental test is successful.



9. The physical connection diagram of the experiment is as follows:



(3)Core code program

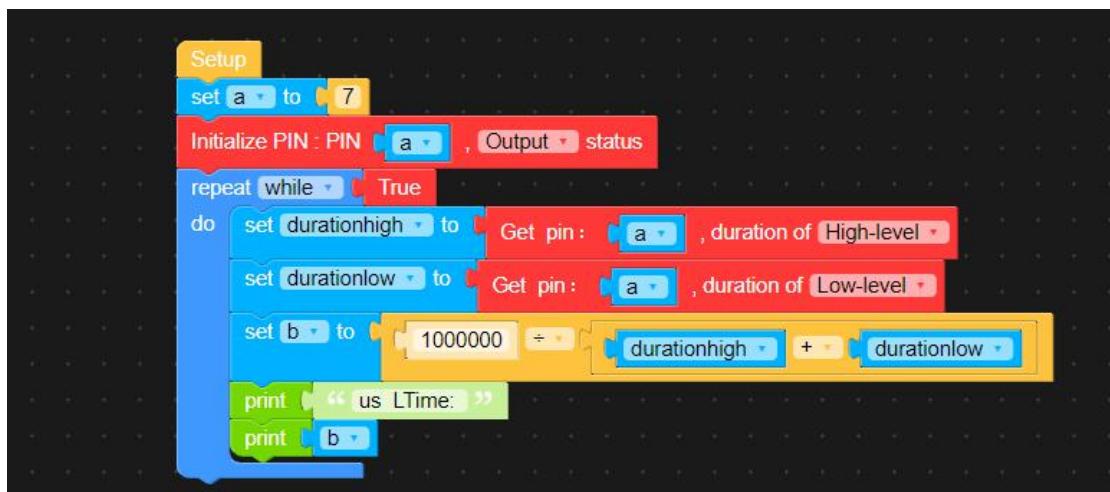
After the above hands-on operation, you must be very interested to know how we

program to use Potentiometer and NE555 Timer to make Frequency Meter on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

In the GwBlock graphical editor, all code programs are executed from **Setup**. The data obtained by the calculation with

 instruction is stored in the variable b,

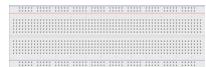
and then the value of the variable b is printed and outputted on the command window with the instruction .



Lesson 30 The Application of the PIR Movement Sensor

In this lesson, we will learn how to use PIR movement sensor to detect biological intrusions.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	1	
LED	1	
PIR Movement Sensor	1	
Male to Female Jumper Wires	Several	

2. The introduction of PIR Movement Sensor

(1) PIR Movement Sensor

The PIR movement sensor we use in this class is HCSR501. It is a kind of sensor that can sense the infrared radiation of the target and make measurements based on the physical properties of infrared. The figure below is the passive infrared sensor we use in this class, also known as human infrared sensor. It has three pins, namely VCC,

OUT and GND. Two yellow knobs directly opposite the pin can adjust the sensitivity and reception distance of the sensor. It can detect every living and non-living thing whose temperature is above absolute zero. In addition, the PIR sensor has two potentiometers: one for adjusting the sensor's sensitivity (or rather, the sensor's sensing distance), and one for adjusting how long the output stays high when any human movement is detected.



(2) The working principle of the PIR Movement Sensor

PIR movement sensor, also known as human infrared sensor, it can detect the human body in the form of heat emission of infrared radiation, so can be used to detect the presence of people or animals. The human body has a constant body temperature, generally at 37 degrees, so it will emit a specific wavelength of about 10UM of infrared. The passive infrared probe is used to detect about 10UM of infrared emitted by the human body. The infrared ray about 10UM emitted by the human body is enhanced through the Fresnel filter and then gathered to the infrared sensor. Infrared induction source usually adopts pyroelectric element, which will lose the charge balance when it receives the human body's infrared radiation temperature changes, and release the charge outward. After the subsequent circuit is detected and processed, the alarm signal can be generated.

Within the range of the setting, the high level will be output when someone is detected to be active, otherwise the low level will be output. It can repeatedly trigger the output high level. After sensing the human body, keep the output high level for a delay period and no longer sense, and re-detect after the delay is over. If someone is

active within its sensing range, its output will be It keeps high level until the person leaves, and then changes the high level to low level (the sensor module will automatically extend the delay time for each activity detected again after the end of a delay period, and take the time of the last activity as the starting point of the delay time).

(1) Pyroelectric effect

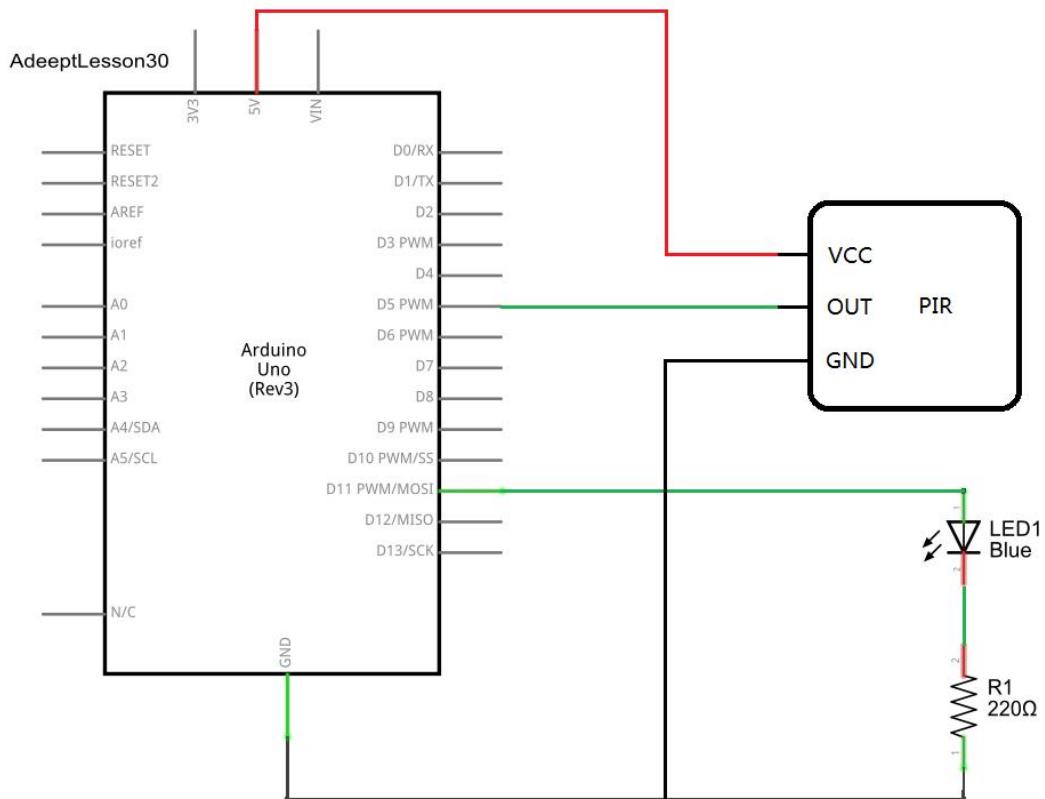
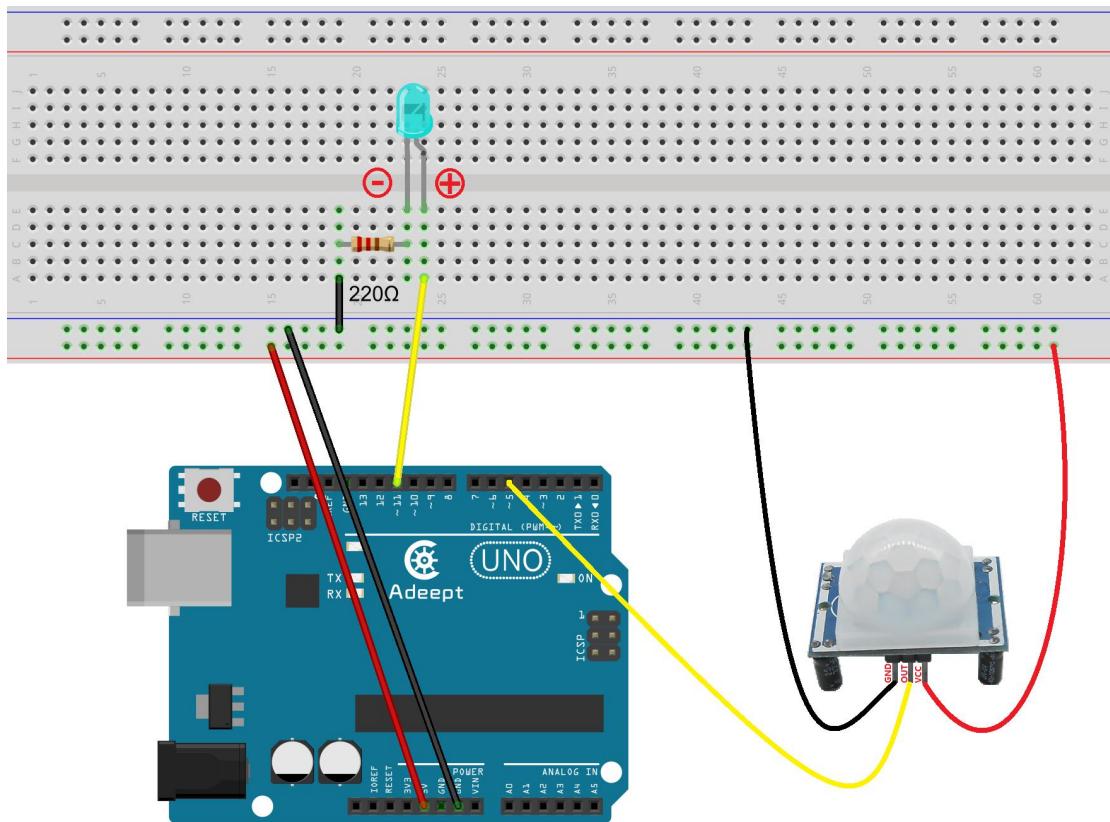
When some crystals are heated, an equal but opposite amount of charge will be generated at both ends of the crystal. This phenomenon of electrical polarization due to thermal change is called pyroelectric effect.

(2) Fresnel lens

The Fresnel lens is made according to the Fresnel principle. The Fresnel lens is divided into two forms: refraction and reflection. Second, the detection area is divided into several bright areas and dark areas, so that moving objects entering the detection area can generate infrared signals on PIR in the form of temperature changes, so that PIR can generate electrical signals of changes. The sensitivity of pyroelectric human body infrared sensor (PIR) is greatly increased.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:



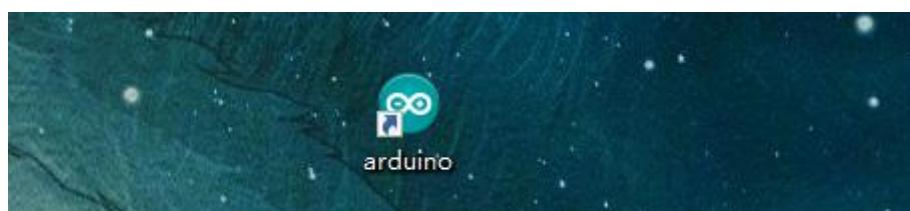
4. PIR Movement Sensor

We provide two different methods to use PIR movement sensor to detect biological intrusions. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

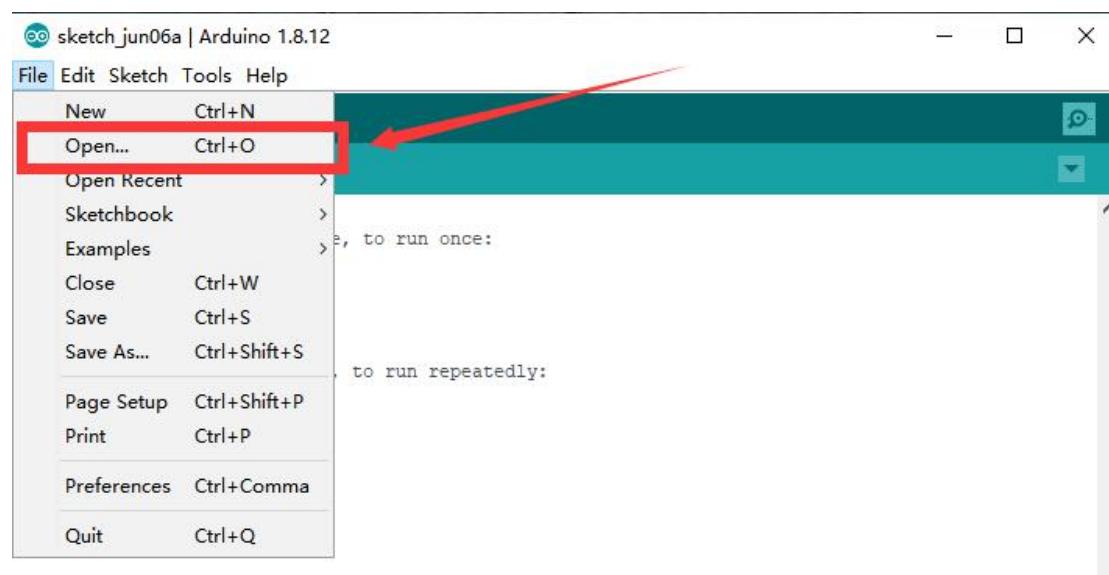
1. Using C language to program to detect biological intrusions on Arduino UNO

(1) Compile and run the code program of this course

1. Open the Arduino IDE software, as shown below:

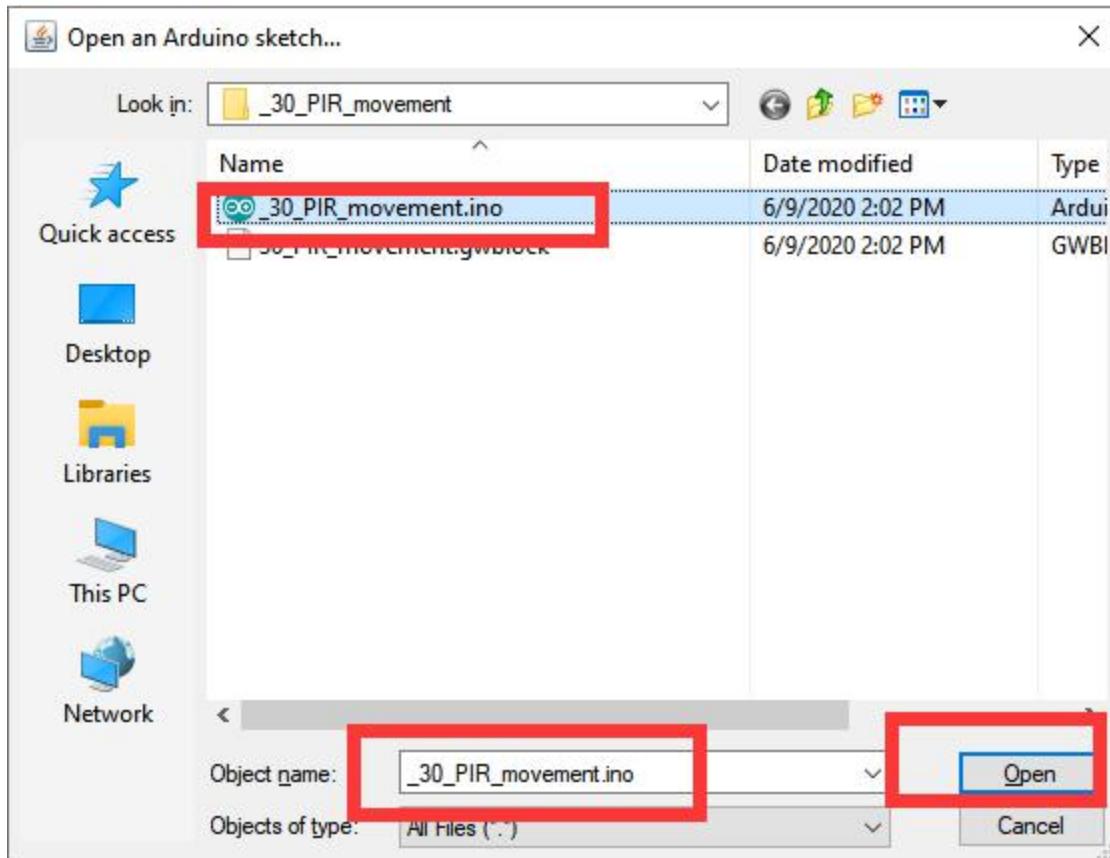


2. Click Open in the File drop-down menu:

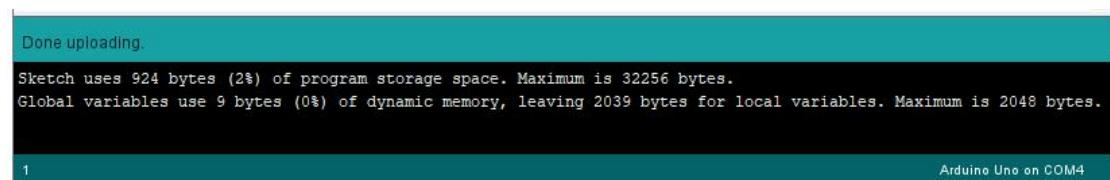


3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the

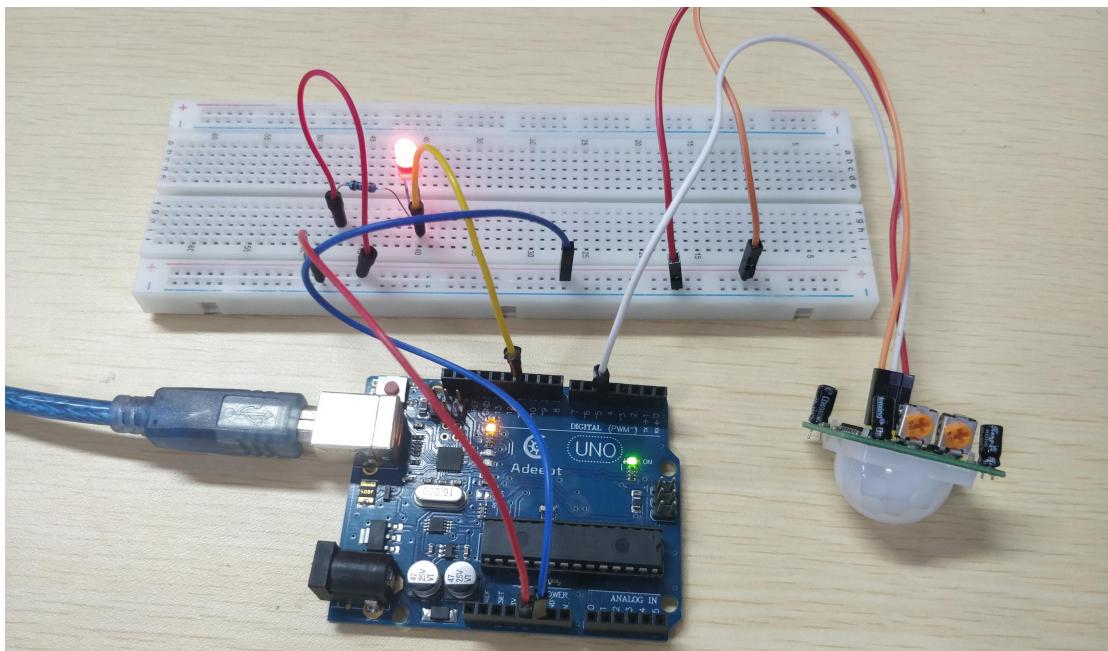
Code\30_PIR_movement directory. Select 30_PIR_movement.ino. This file is the code program we need in this course. Then click Open.



4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.



5. After successfully running the program, when you are close to the PIR movement sensor, the LED will be on, indicating that a biological intrusion has been detected; when you are away from the PIR movement sensor, the LED will be off, indicating that no biological intrusion has been detected. The physical connection diagram of the experiment is as follows:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to use PIR movement sensor to detect biological intrusions on Arduino UNO. We will introduce how our core code can be achieved:

1.In the setup() function, set the ledpin pin to OUTPUT mode and PIRpin to INPUT mode with the pinMode() function;

```
void setup() {
    // put your setup code here, to run once:
    pinMode( ledpin,OUTPUT);      //initialize the led pin as output
    pinMode( PIRpin,INPUT);       //initialize the PIR pin as input
}
```

2.In the loop() function,determine whether the PIR movement sensor detects a biological approaching with if (digitalRead(PIRpin)==LOW), if digitalRead(PIRpin)==LOW does not detect that someone is approaching, then the LED will be off with digitalWrite(ledpin , LOW); if an intrusion is detected,the LED will be on with digitalWrite(ledpin, HIGH).

```
void loop() {  
  
    if(digitalRead(PIRpin)==LOW) //Detecting whether the body movement information  
    {  
        digitalWrite(ledpin,LOW); //LED OFF  
    }else  
    {  
        digitalWrite(ledpin,HIGH); //LED ON  
    }  
}
```

2.Using graphical code blocks to program to detect biological intrusions on Arduino UNO

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

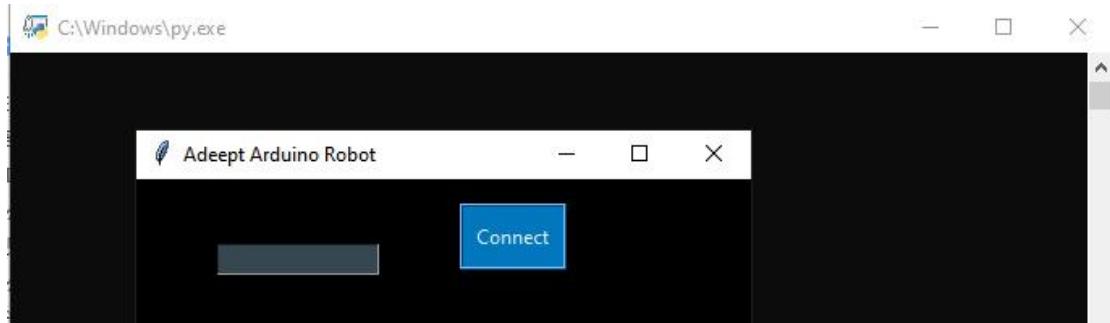
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

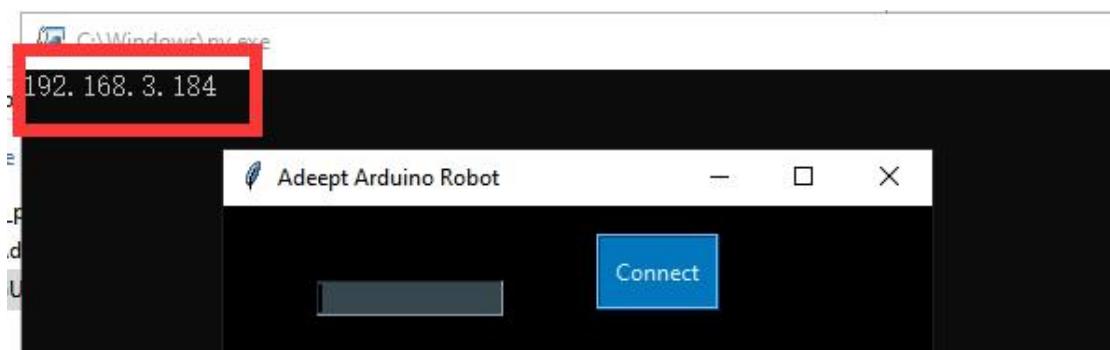
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

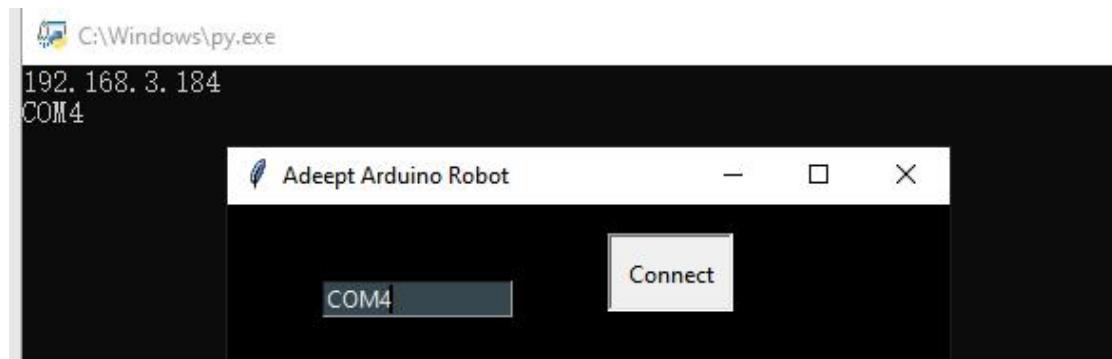
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



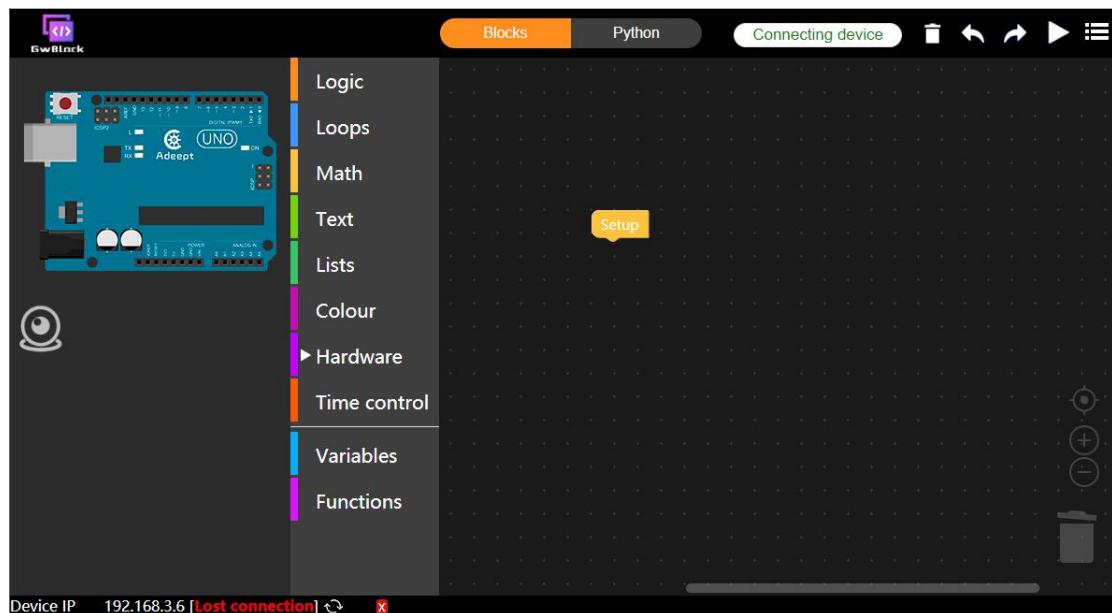
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



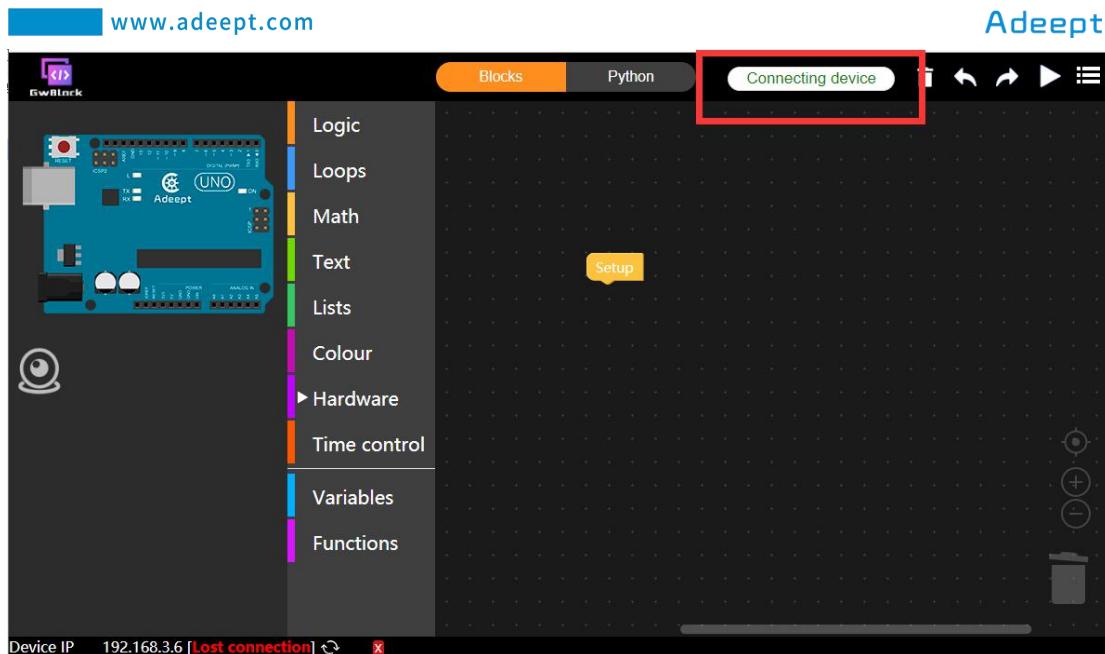
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

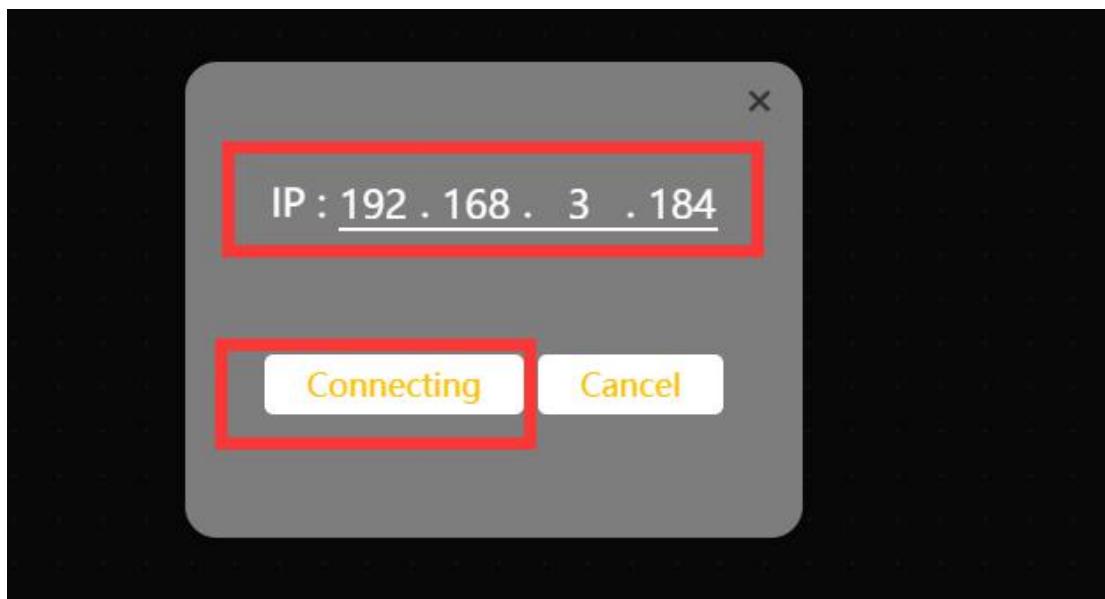
After successfully entering the website, the interface is as follows:



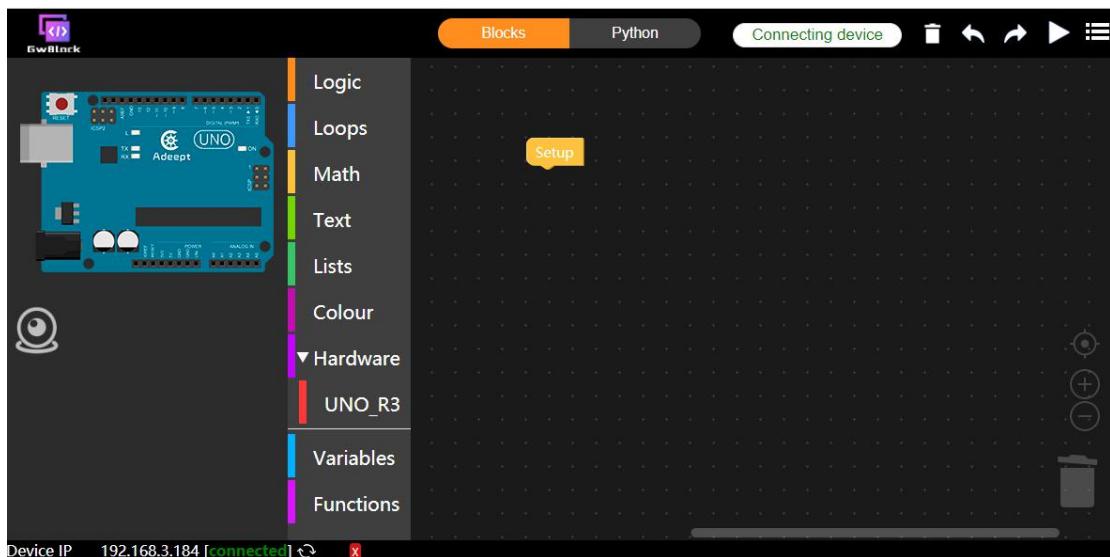
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

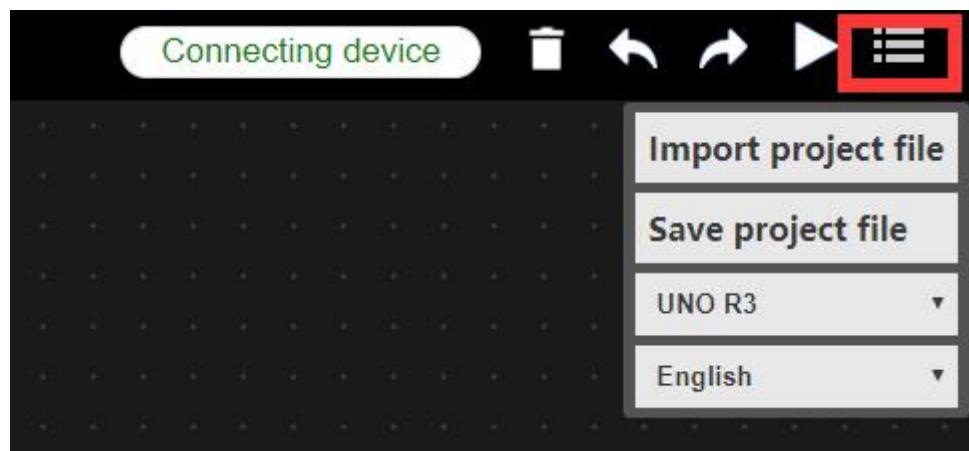


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

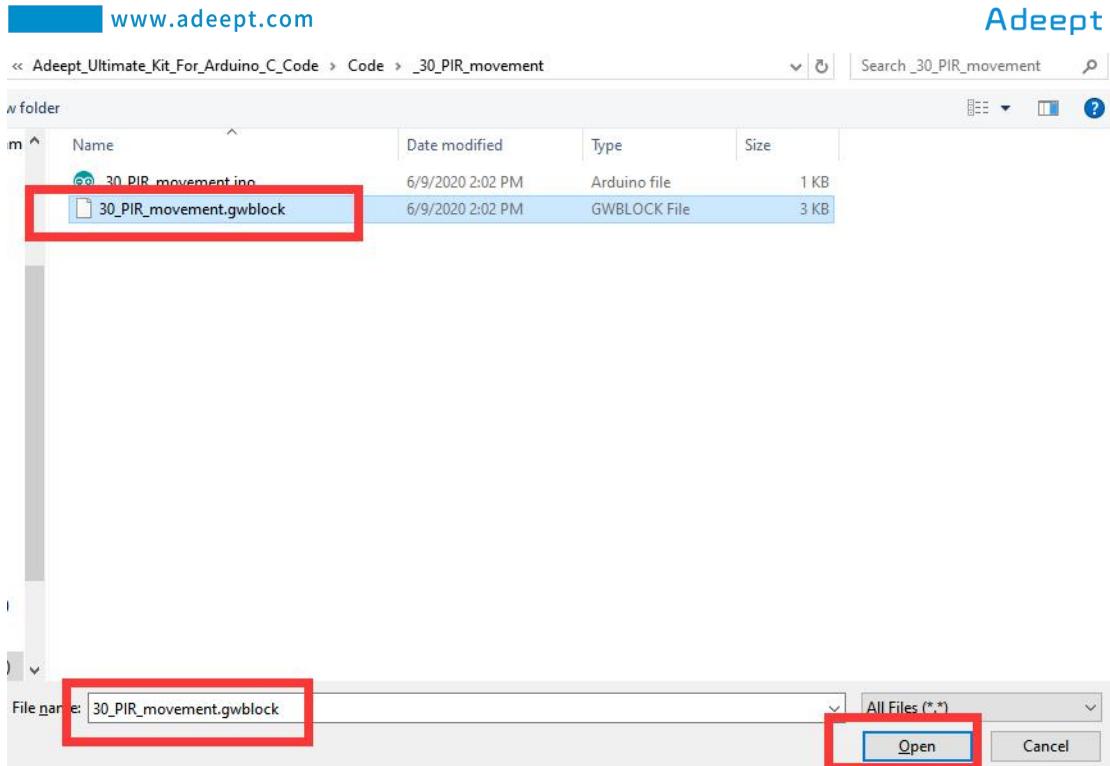
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_30_PIR_movement

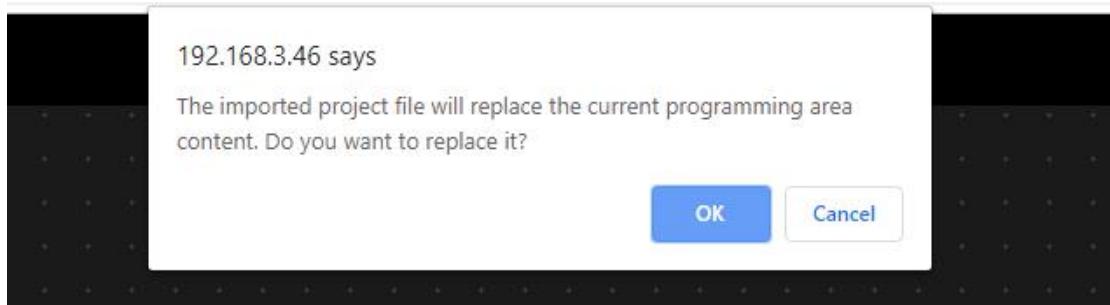
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

5. Select the "30_PIR_movement.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



6.Click OK.It will show as below:

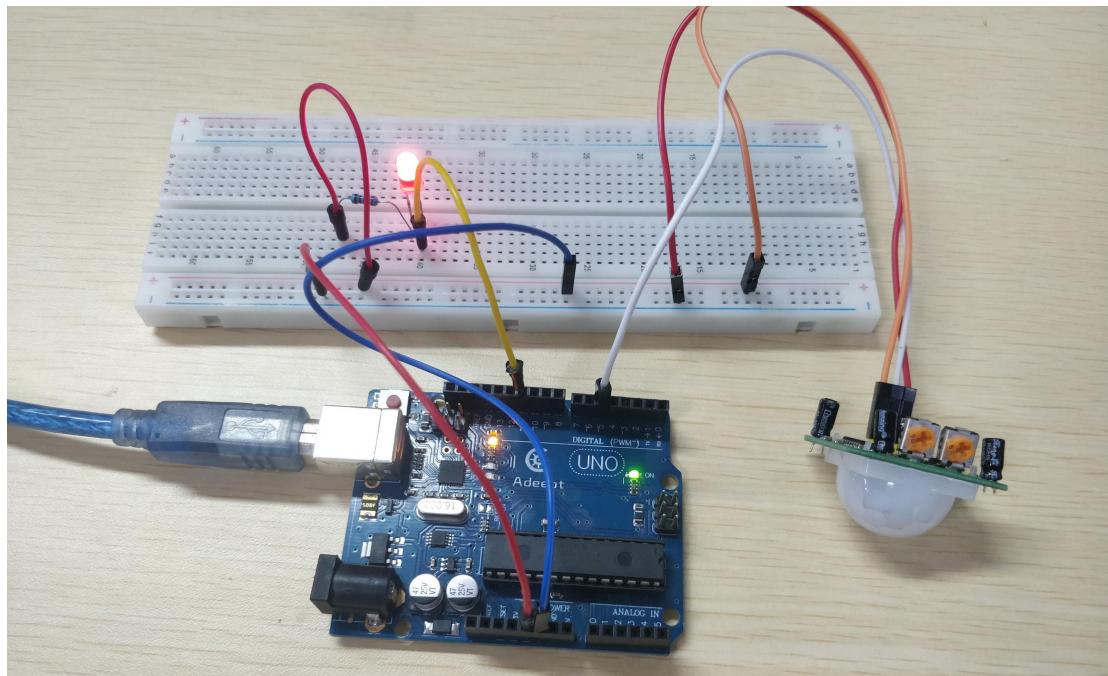


7.It will show as below after successfully opening:



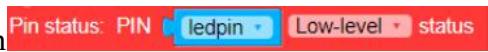
8. Click the button  on the upper right corner.After successfully running the

program, when you are close to the PIR movement sensor, the LED will be on, indicating that a biological intrusion has been detected; when you are away from the PIR movement sensor, the LED will be off, indicating that no biological intrusion has been detected. The physical connection diagram of the experiment is as follows:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to use PIR movement sensor to detect biological intrusions on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

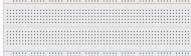
In the GwBlock graphical editor, all code programs are executed from **Setup**. In the while loop statement, it is judged by the instruction  whether the PIR movement sensor detects biological intrusion. When PIRpin is equal to 0, it means that no biological intrusion is detected. At this time, the LED is turned off by the instruction ; when PIRpin is not equal to 0, it means that a biological intrusion is detected, and the LED is controlled to light up via commands .



Lesson 31 Using IR Remoter Controller to control Relay

In this lesson, we will study how to use IR Remoter Controller to control Relay.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
IR Receiver HX1838	1	
Remote Controller	1	
Male to Female Jumper Wires	3	
Relay Module	1	

2. The introduction of IR remoter controller and relay

(1)IR remoter controller

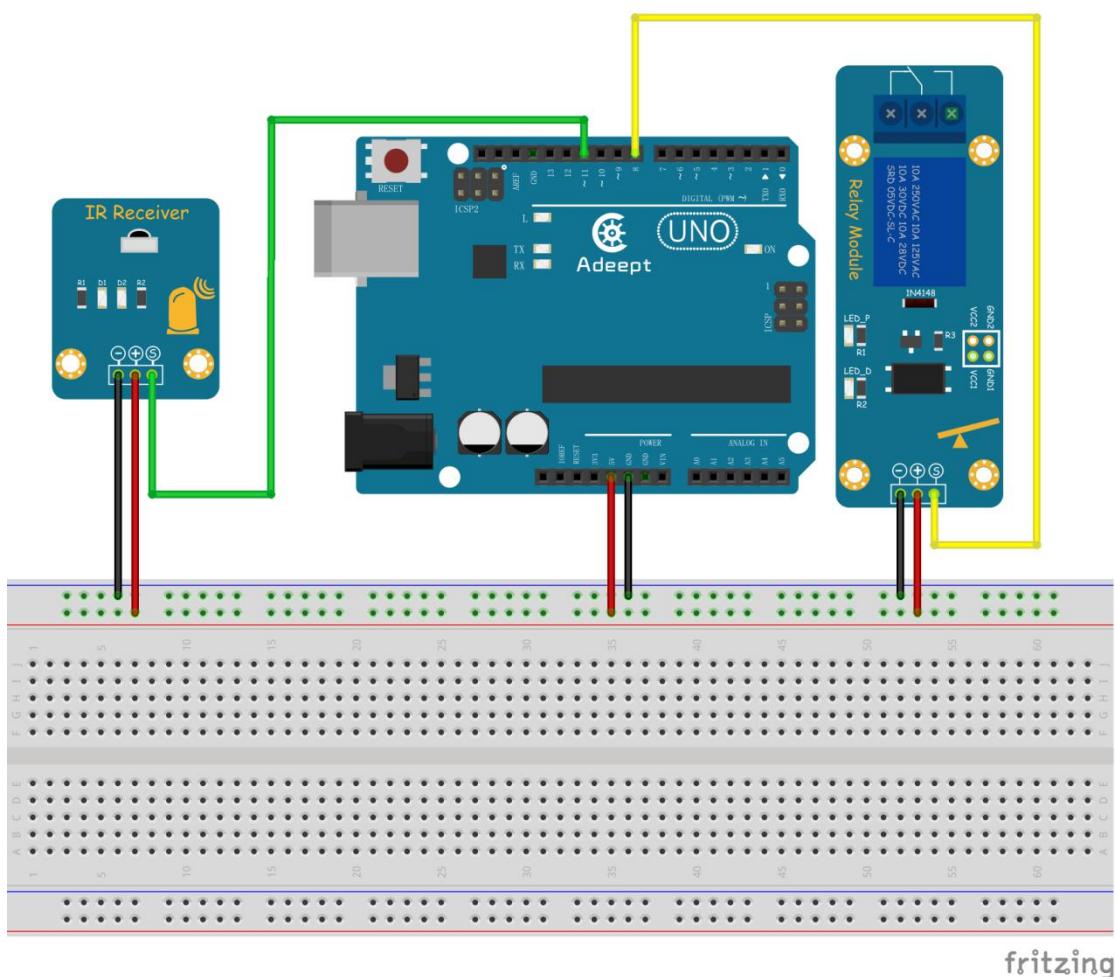
Please refer to Lesson 17, for we have introduced the IR remoter controller in detail in Lesson 17.

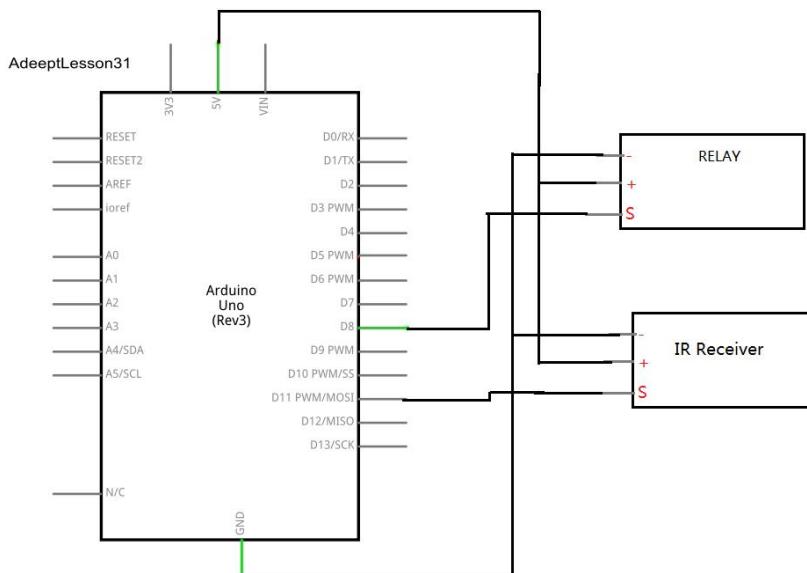
(2) relay

Please refer to Lesson 4, for we have introduced the relay in detail in Lesson 4.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:





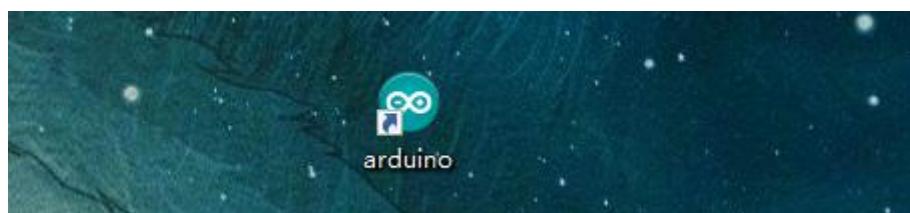
4. Using IR Remoter Controller to control Relay

We provide two different methods to use IR Remoter Controller to control Relay. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

1.Using C language to program to use IR Remoter Controller to control Relay on Arduino UNO

(1)Compile and run the code program of this course

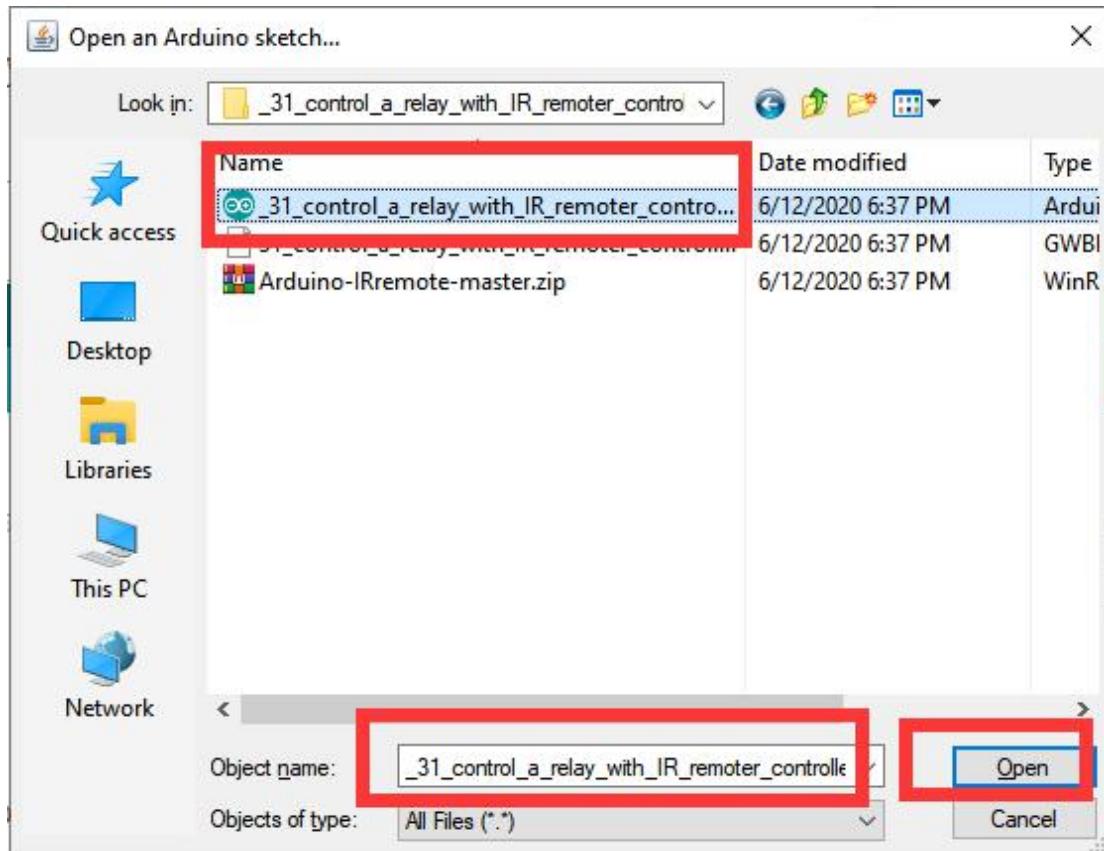
1. Open the Arduino IDE software, as shown below:



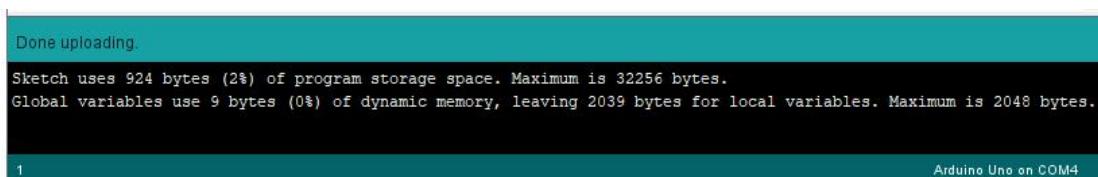
2. Click Open in the File drop-down menu:



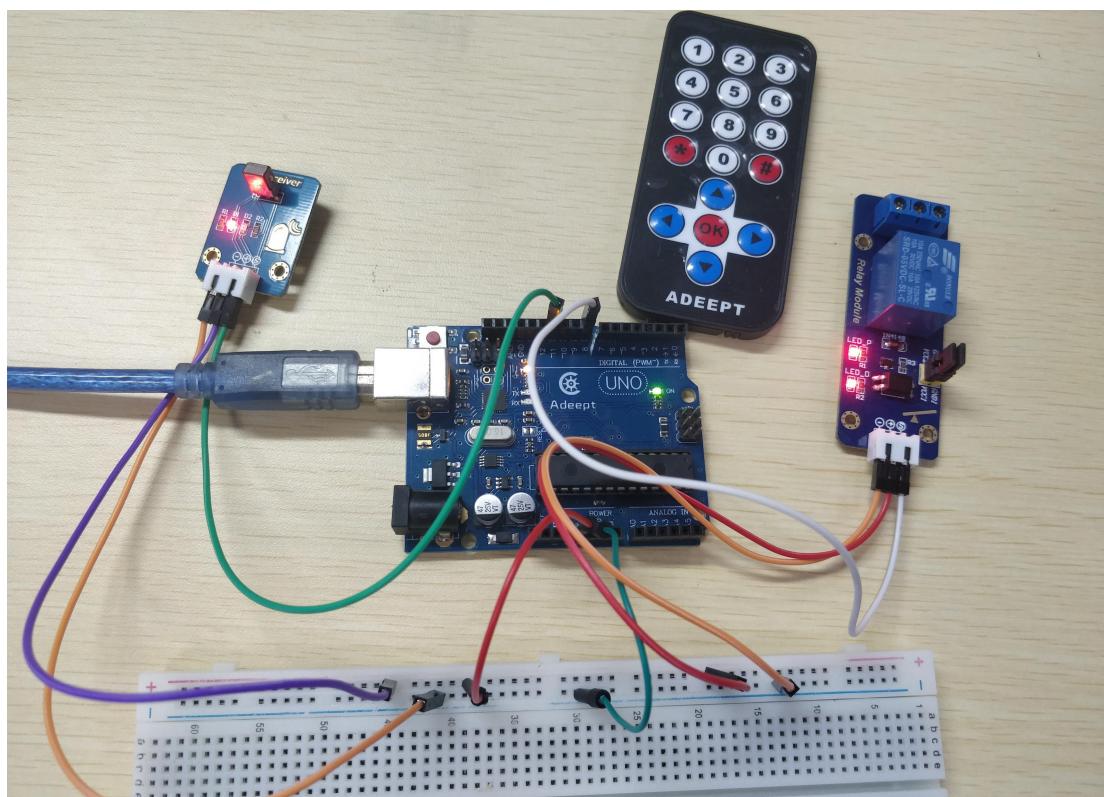
4. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\31_control_a_relay_with_IR_remoter_controller directory. Select31_control_a_relay_with_IR_remoter_controller.ino. This file is the code program we need in this course. Then click Open.



4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.



5. After running the program successfully, when you click the number 1 on the remote control, it means to control the relay switch, then the LED_D indicator on the relay will go out; when you click the number 0 on the remote control, it means that the relay will be controlled to close and energize. The LED_D indicator of the relay will be lit up. The physical connection diagram of the experiment is as follows:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to use IR Remoter Controller to control Relay on Arduino UNO. We will introduce how our core code can be achieved:

1. In the `setup()` function, set the `relayPin` pin to OUTPUT mode with the

pinMode() function. Initialize the IR remoter controller by the irrecv.enableIRIn() function, that is, turn on the IR remoter controller.

2.In the loop() function, use if(results.value==16750695) to determine whether the 0 button of the remote controller is pressed. If button 0 is pressed, digitalWrite(relayPin, LOW) will control the relay to turn on; If button 1 is pressed, then digitalWrite(relayPin, HIGH) will control the relay to turn on.

```
void setup()
{
    pinMode(relayPin, OUTPUT); //initialize the relayPin as an output
    irrecv.enableIRIn(); // Initialization infrared receiver
}

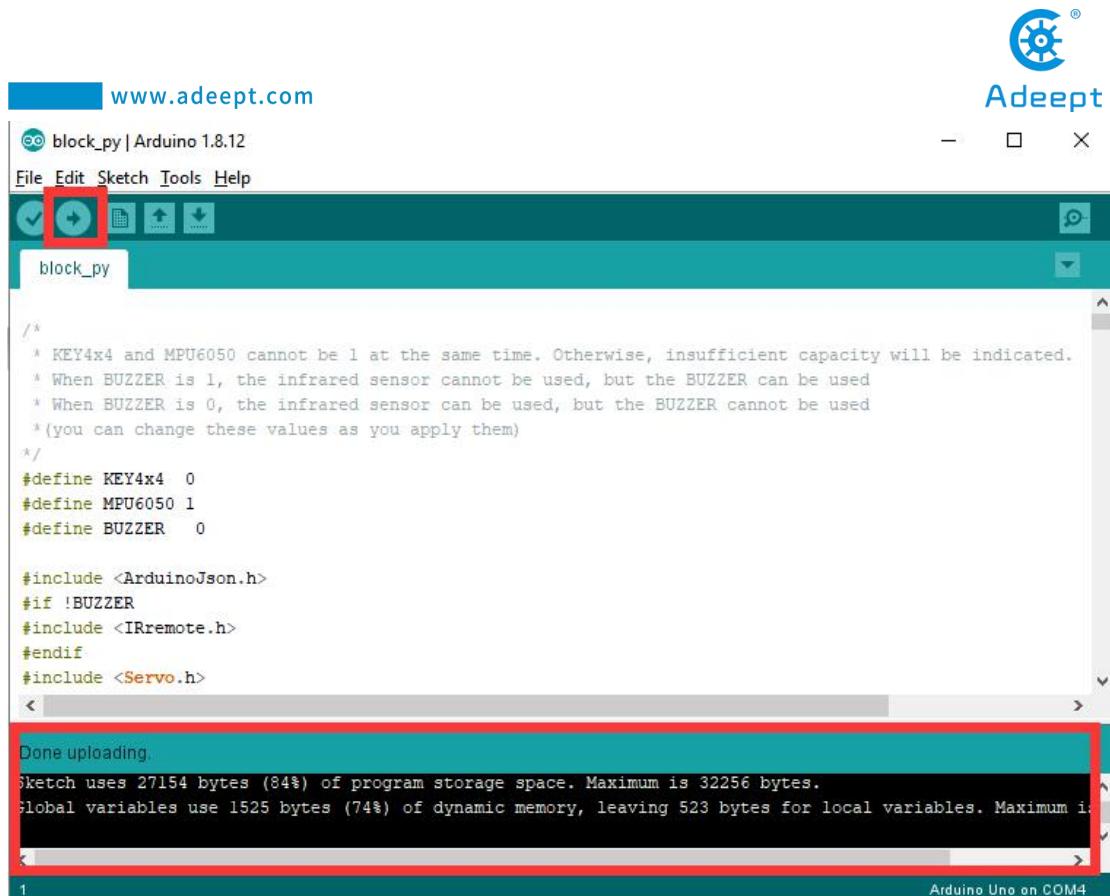
void loop()
{
    if (irrecv.decode(&results)) {
        if(results.value==16750695)//0
        {
            digitalWrite(relayPin, LOW); //drive the relay is turn OFF
        }
        if(results.value==16753245)//1
        {
            digitalWrite(relayPin, HIGH); //drive the relay is turn ON
        }
        irrecv.resume(); // Receiving the next value
    }
}
```

2.Using graphical code blocks to program to use IR Remoter Controller to control Relay on Arduino UNO

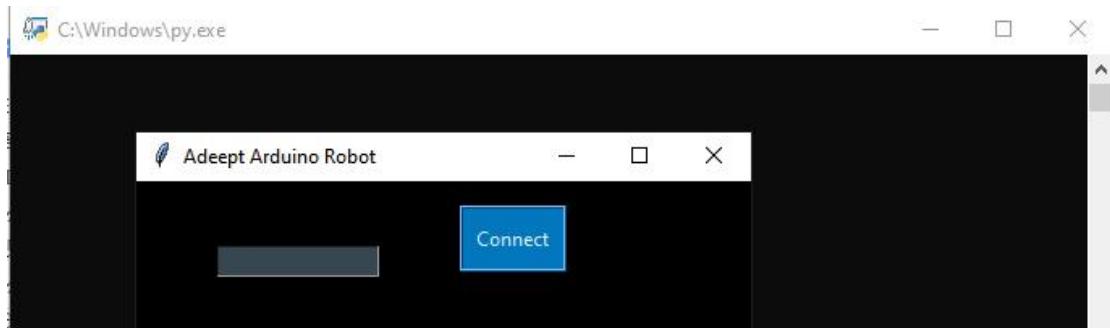
(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

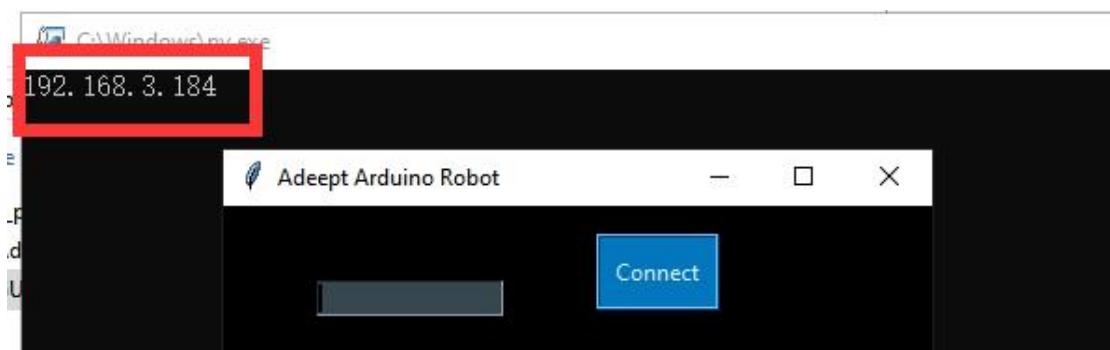
1.Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:



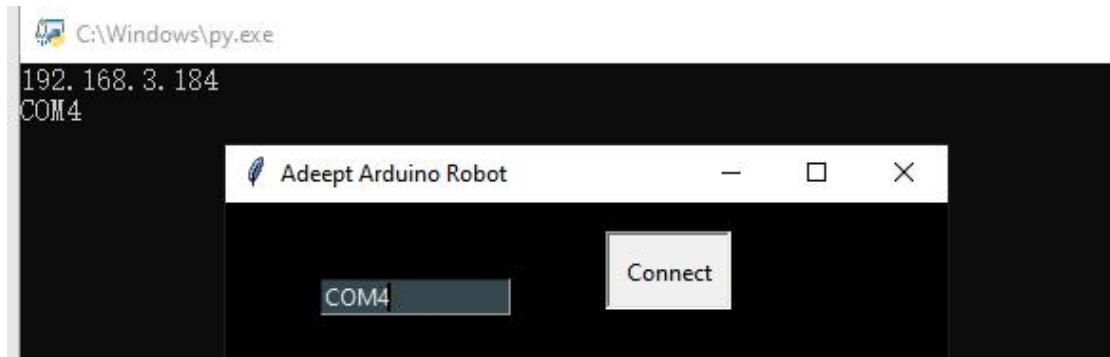
2.Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again.Then open the websocket folder. Double-click to open the GUI info v1.0.py file.It will show as below:



3.Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



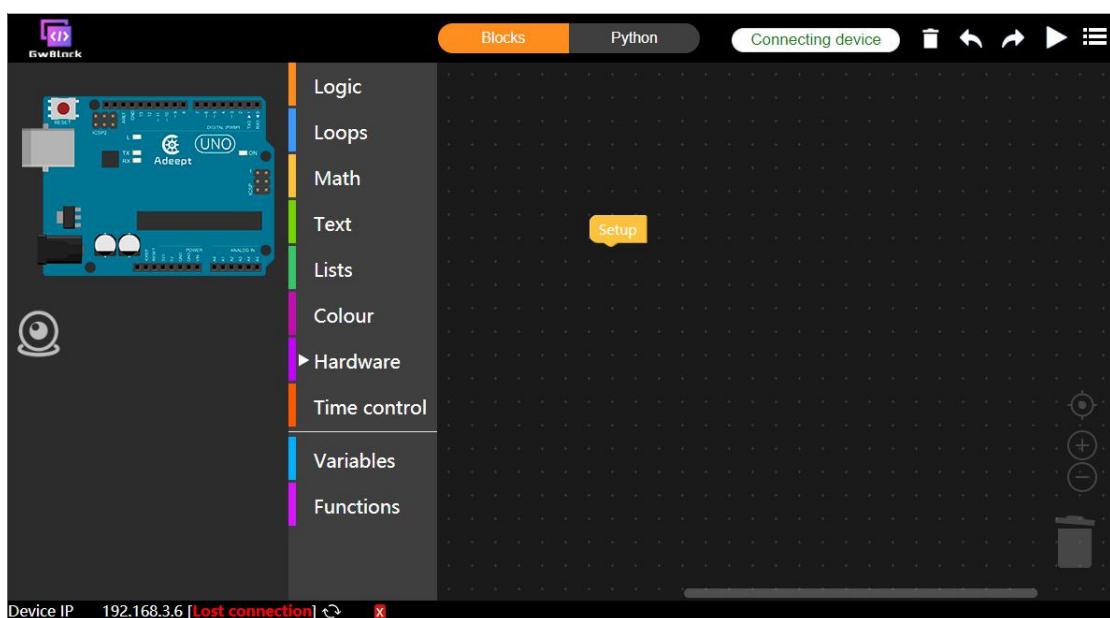
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



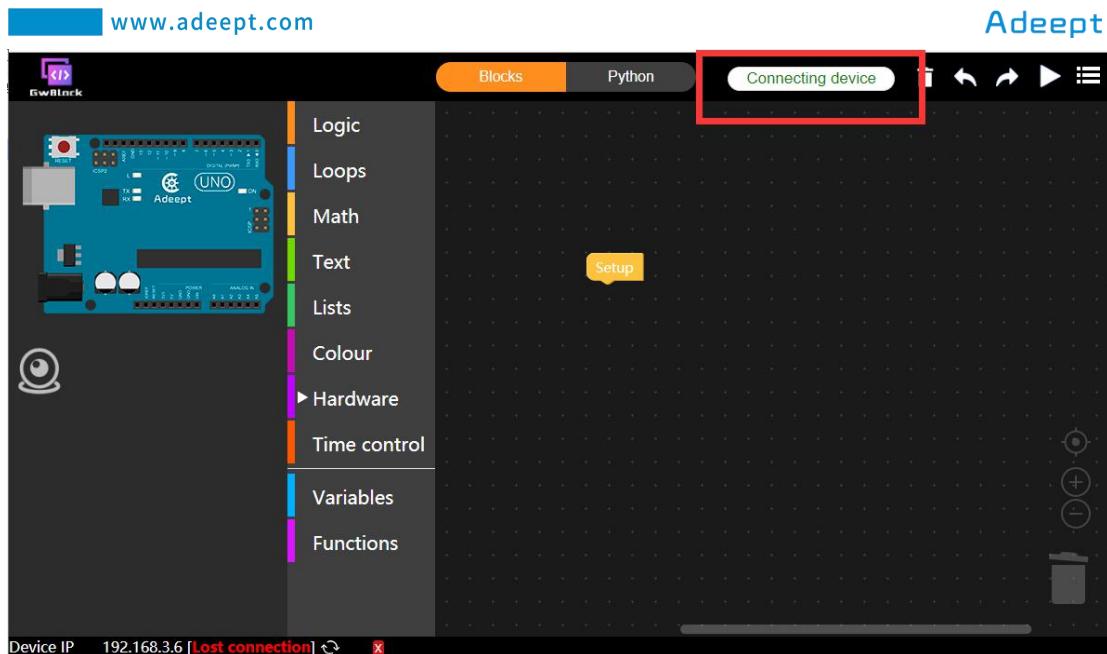
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

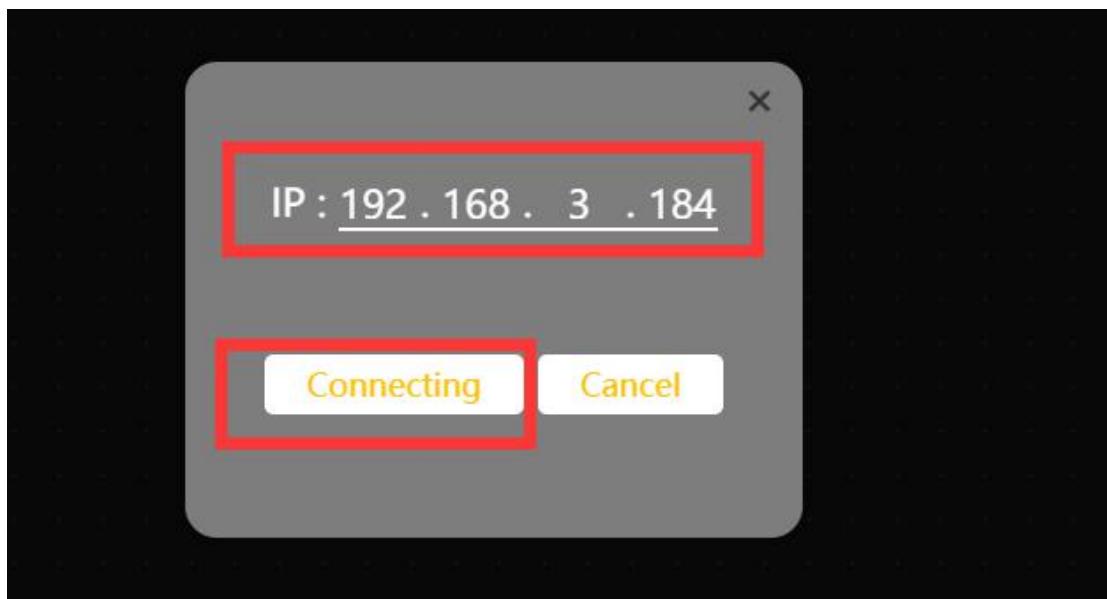
After successfully entering the website, the interface is as follows:



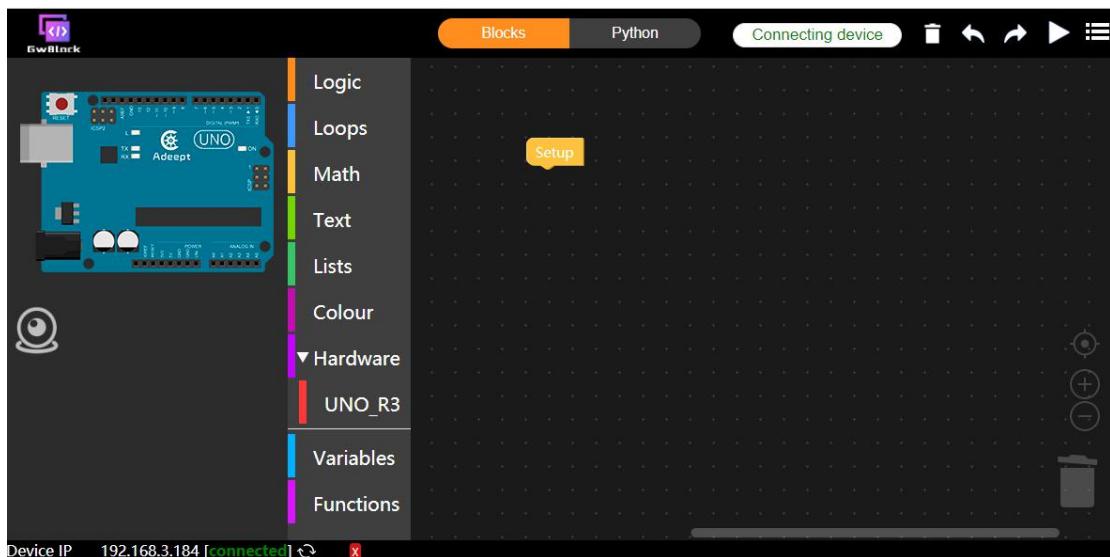
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:



(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

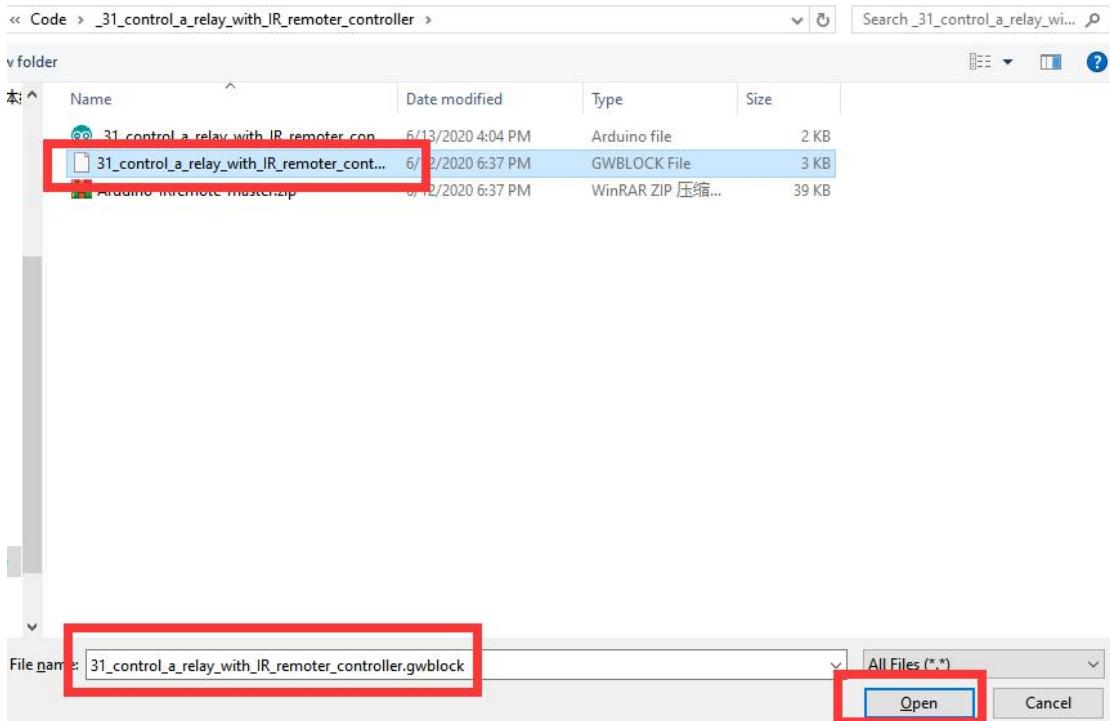
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_31_control_a_relay_with_IR_remoter_controller

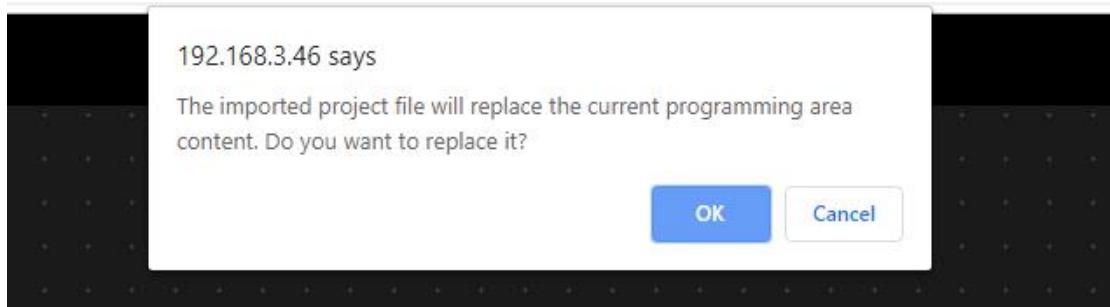
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

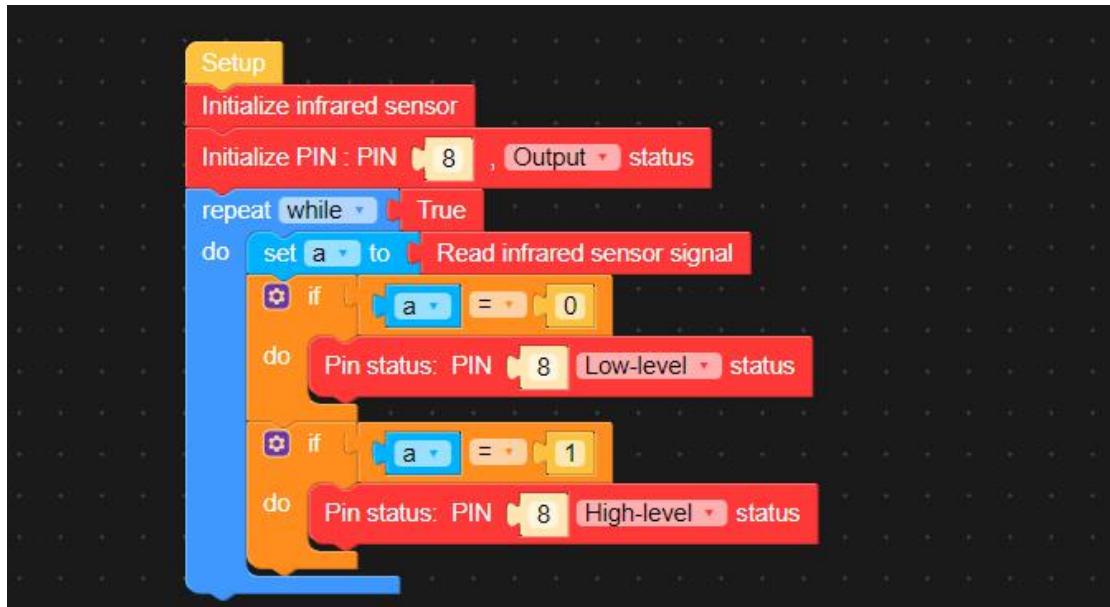
5. Select the "31_control_a_relay_with_IR_remoter_controller.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



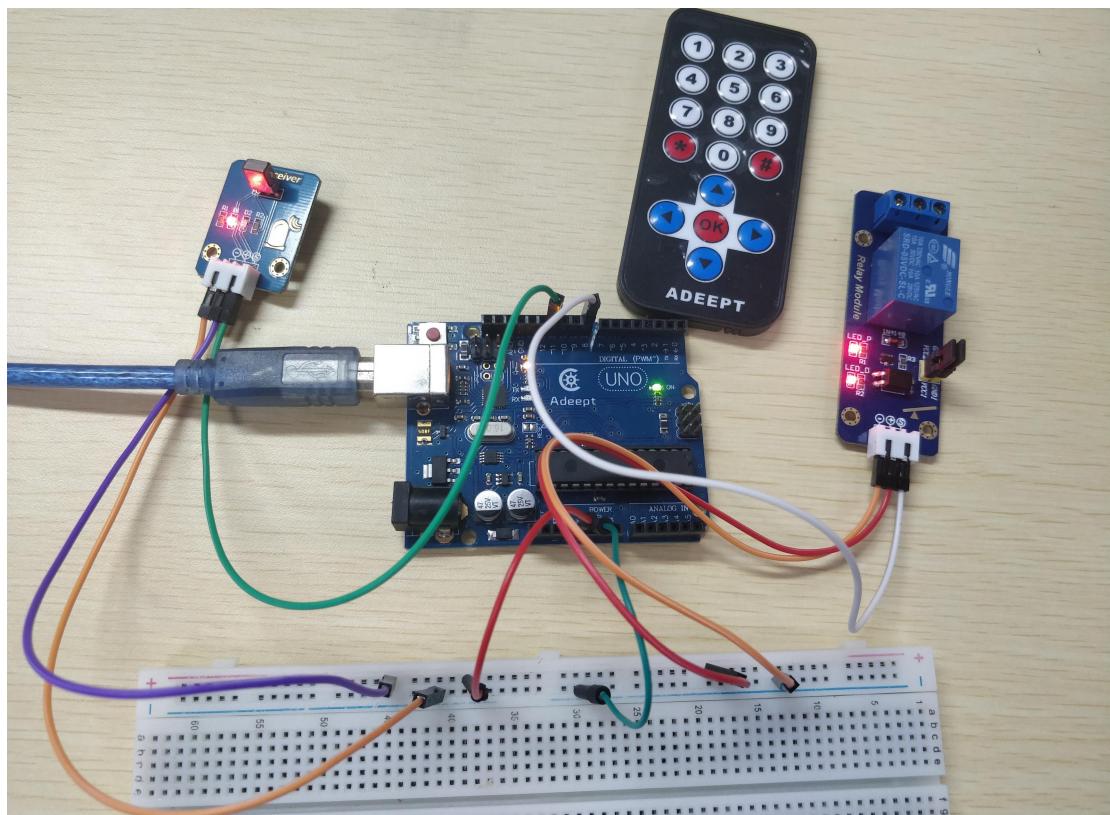
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



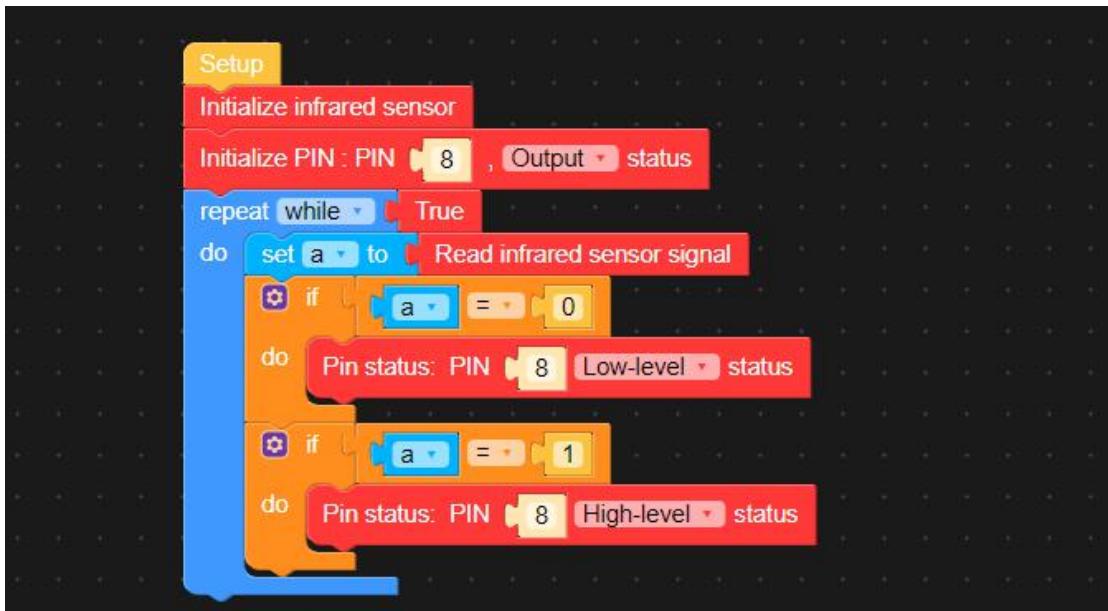
8. Click the button  on the upper right corner. After successfully running the program, it means that our experimental test is successful. The physical connection diagram of the experiment is as follows:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to use IR Remoter Controller to control Relay on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

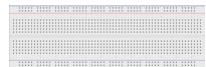
In the GwBlock graphical editor, all code programs are executed from **Setup**. First, you need to use the command **Initialize infrared sensor** to initialize the IR remoter controller; in the while loop, first use the command **Read infrared sensor signal** to read the signal value of the infrared receiver. Use the if statement to determine the value 0 or 1 of the key pressed by the remote controller. Control the relay to be turned on and off with the command **Pin status: PIN 8 Low-level status** and **Pin status: PIN 8 High-level status**.



Lesson 32 Controlling RGB LED with IR Remoter Controller

In this lesson, we will learn how to use IR Remoter Controller to control RGB LED.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
Resistor(220Ω)	3	
RGB LED	1	
IR Receiver HX1838	1	
Remote Controller	1	
Male to Female Jumper Wires	3	

2. IR Remoter Controller and RGB LED

(1) IR Remoter Controller

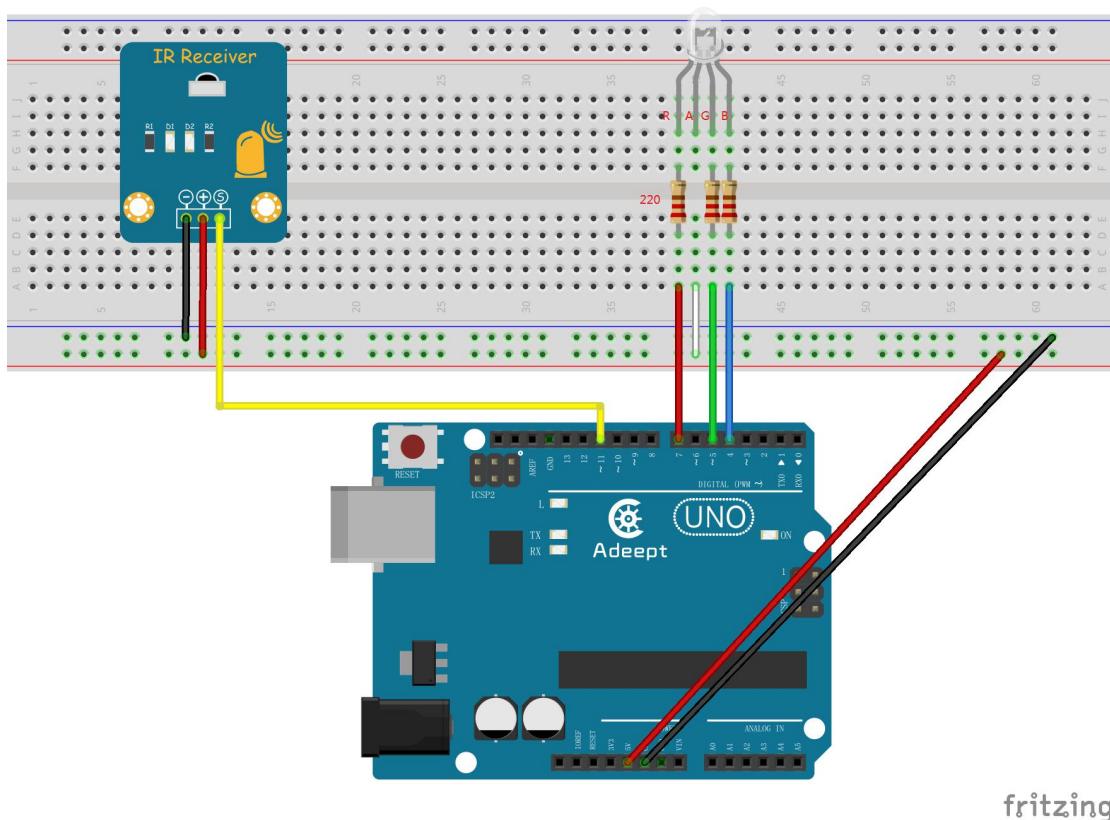
Please refer to Lesson 17, for we have introduced the IR remoter controller in detail in Lesson 17.

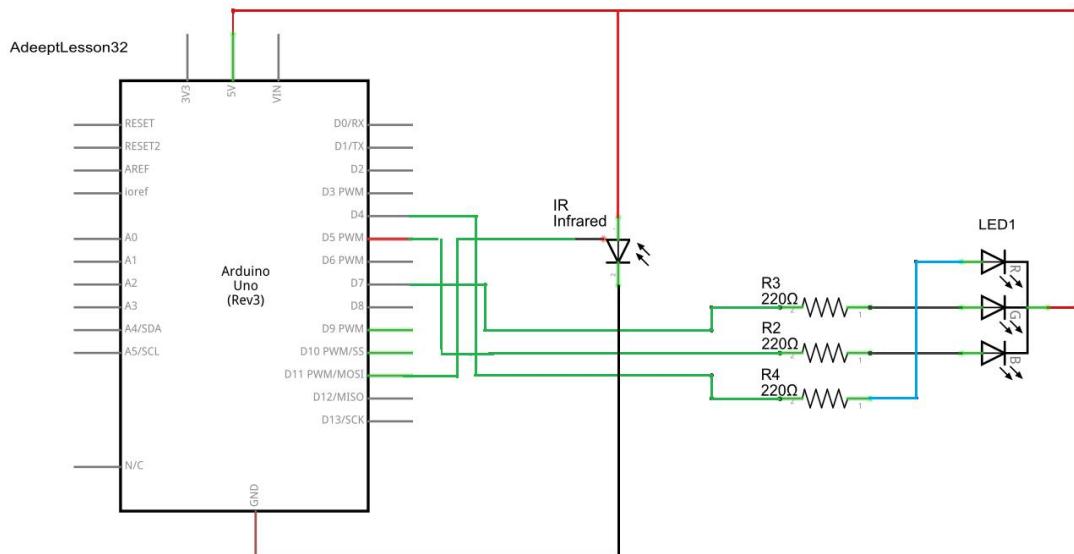
(2)RGB LED

Please refer to Lesson 9, for we have introduced the RGB LED in detail in Lesson 9.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:





4. Using IR Remoter Controller to control RGB LED

We provide two different methods to use IR Remoter Controller to control RGB LED. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

1.Using C language to program to use IR Remoter Controller to control RGB LED on Arduino UNO

(1)Compile and run the code program of this course

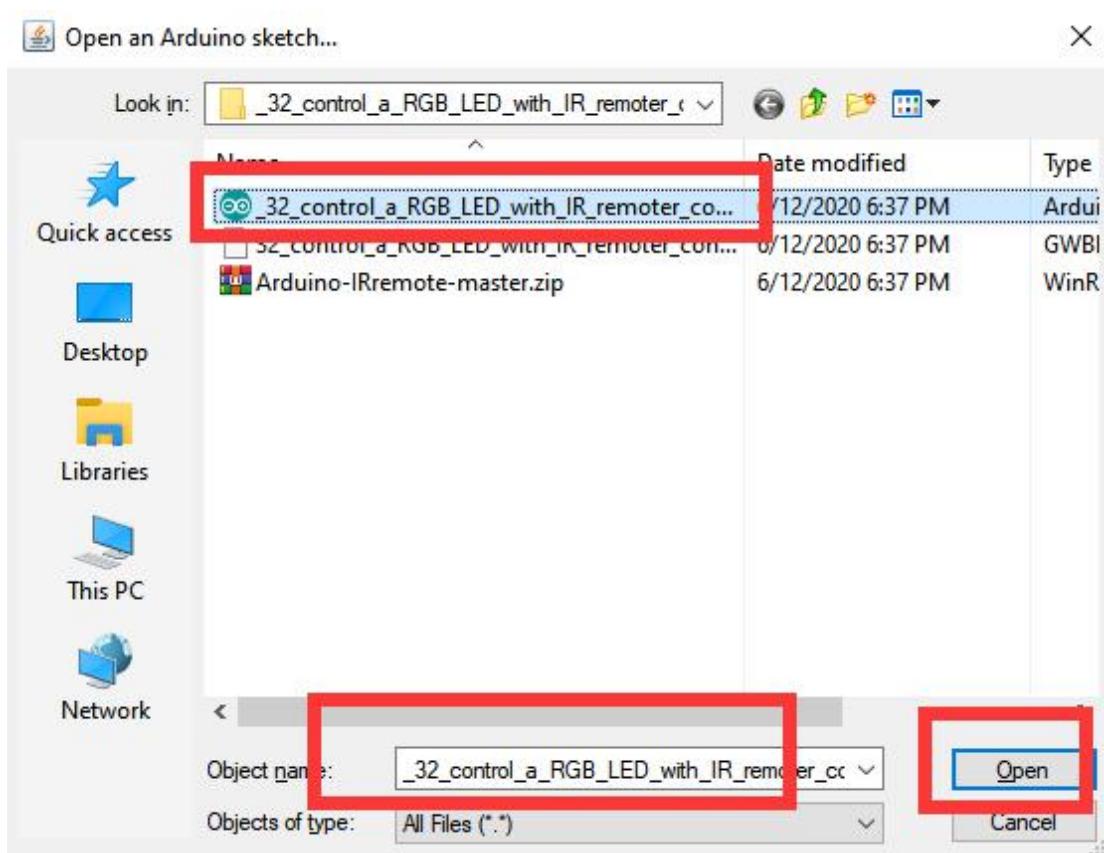
1.Open the Arduino IDE software, as shown below:



2.Click Open in the File drop-down menu:



5. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\32_control_a_RGB_LED_with_IR_remoter_controller directory. Select_32_control_a_RGB_LED_with_IR_remoter_controller.ino. This file is the code program we need in this course. Then click Open.



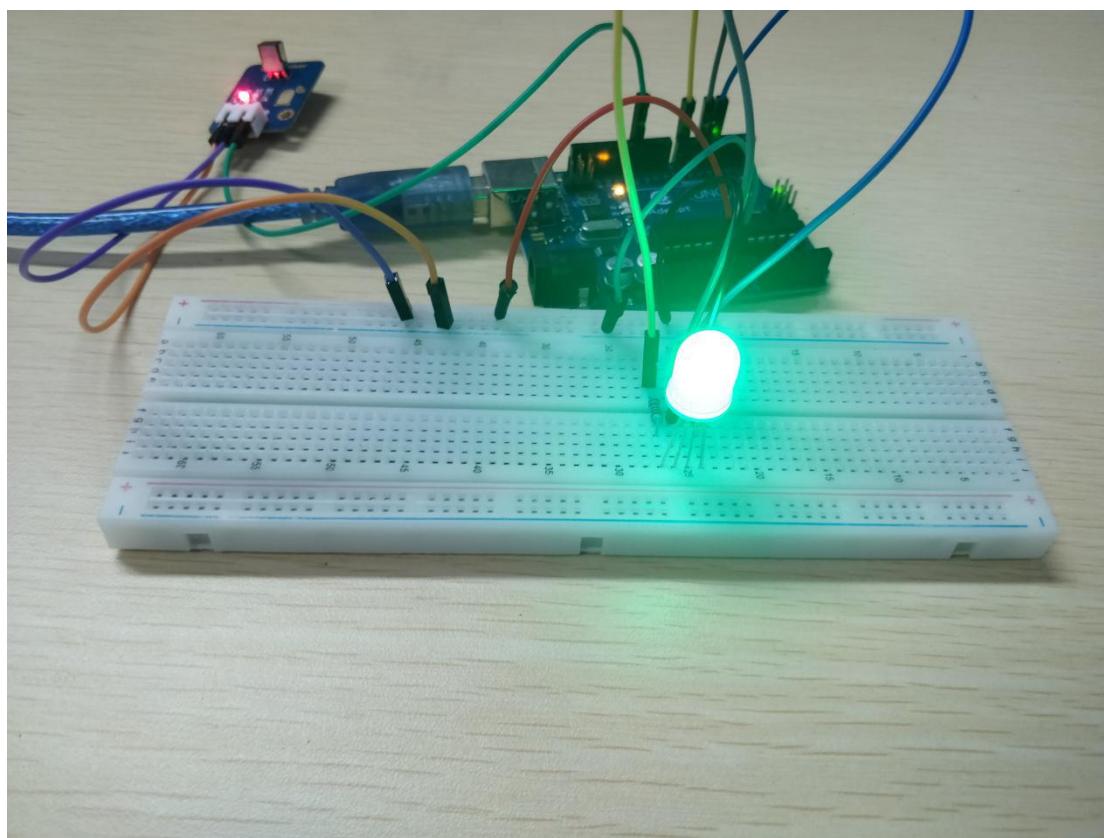
4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1                               Arduino Uno on COM4
```

5. After successfully running the program, we set the program to change the color of the RGB LED with numbers from 0 to 9 on the remote controller. When you press number 1 on the remote controller, the RGB LED will be turned on to red. When you press the number 2, the RGB LED will turn to green. When you press the number 3, the RGB LED will change to blue. It turns blue. When you press the number 0, the RGB LED will go out. The physical connection diagram of the experiment is as follows:



(2) Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to use IR Remoter Controller to control RGB LED on

Arduino UNO. We will introduce how our core code can be achieved:

1. In the setup() function, set the R, G, and B of the RGB LED to the OUTPUT mode with the pinMode() function. Initialize the IR remoter controller with the irrecv.enableIRIn() function, that is, turn on the IR remoter controller.

```
void setup()
{
    pinMode(redPin, OUTPUT); // sets the redPin to be an output
    pinMode(greenPin, OUTPUT); // sets the greenPin to be an output
    pinMode(bluePin, OUTPUT); // sets the bluePin to be an output
    irrecv.enableIRIn(); //Initialization infrared receiver
}
```

2. In the loop() function, read the signal value of the infrared receiver through the if judgment. Control the RGB LED to become the corresponding color with the color(R, G, B) function. If the values of R, G, and B are different, then the the color displayed by RGB LED will be different. If you press button 1 on the remote controller and set the displayed color value to red with the color(255,0,0) function, the RGB LED will show red.

```

void loop()
{
    if (irrecv.decode(&results)) {
        if(results.value==16750695)//0
        {
            color(0,0,0); // turn the RGB LED off
        }
        if(results.value==16753245)//1
        {
            color(255,0,0); // turn the RGB LED red
        }
        if(results.value==16736925)//2
        {
            color(0,255,0); // turn the RGB LED green
        }
        if(results.value==16769565)//3
        {
            color(0,0,255); // turn the RGB LED blue
        }
        if(results.value==16720605)//4
        {
            color(255,255,0); // turn the RGB LED yellow
        }
        if(results.value==16712445)//5
        {
            color(255,255,255); // turn the RGB LED white
        }
        if(results.value==16761405)//6
        {
            color(128,0,255); // turn the RGB LED purple
        }
    }
}

```

2.Using graphical code blocks to program to use IR Remoter

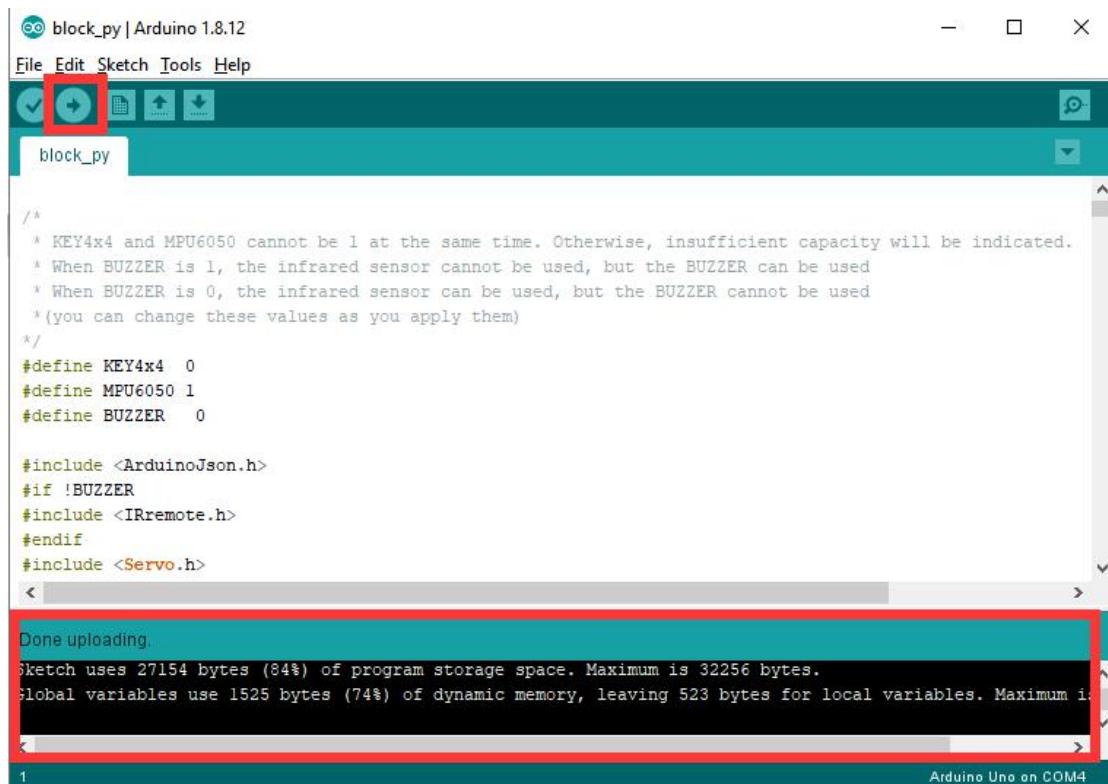
Controller to control RGB LED on Arduino UNO

(1)Connecting to GwBlock graphical editor

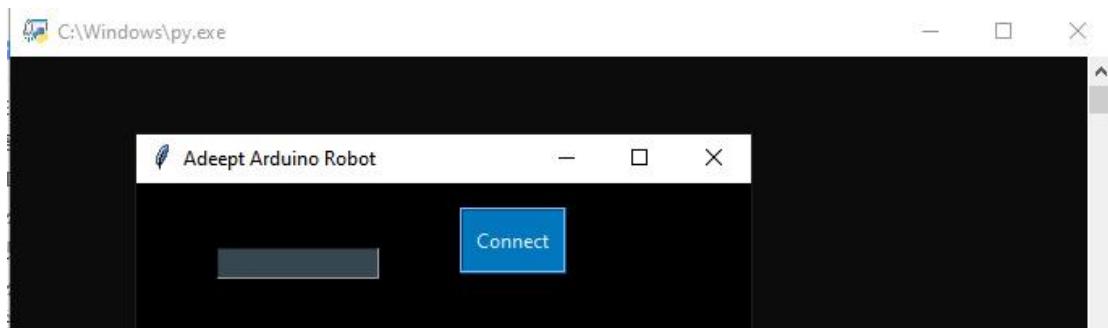
In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1.Open the directory of the folder we provide to the user:

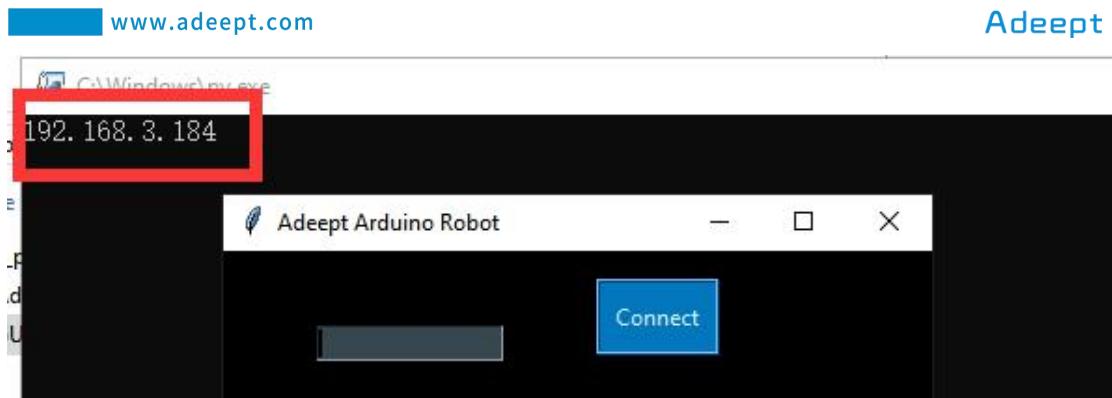
Adept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:



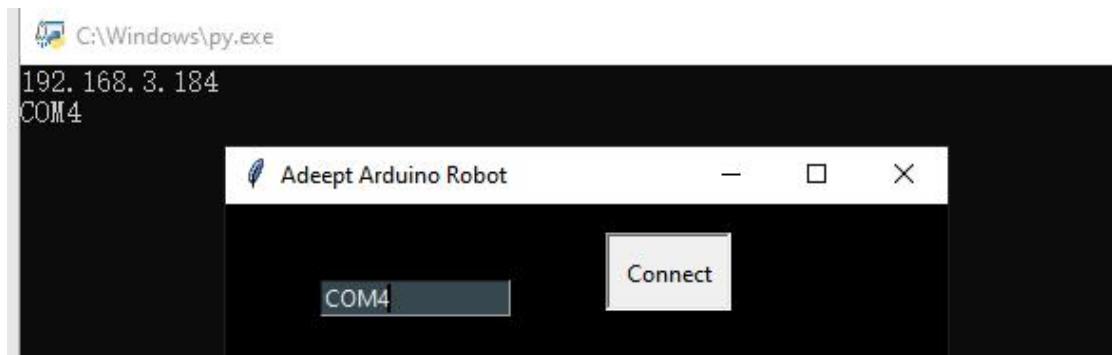
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3.Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



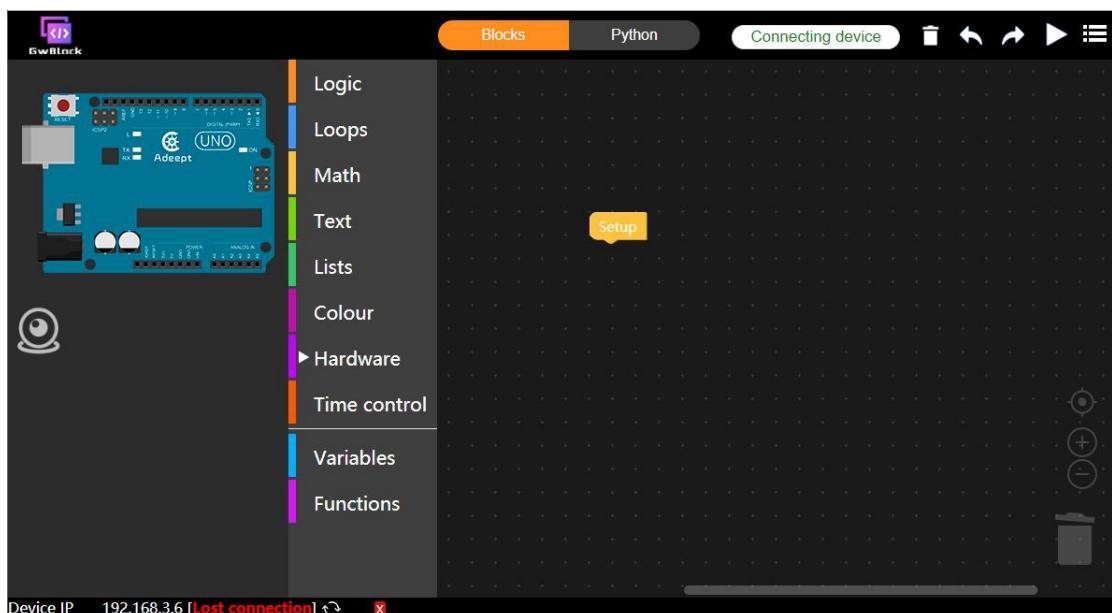
4. Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



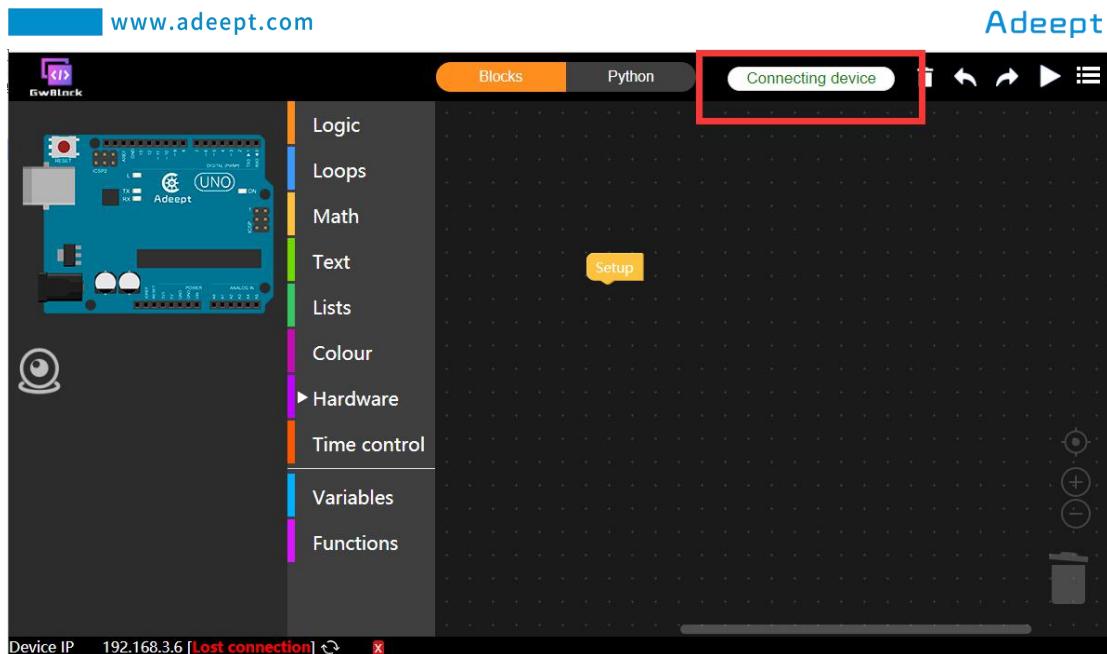
5. Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

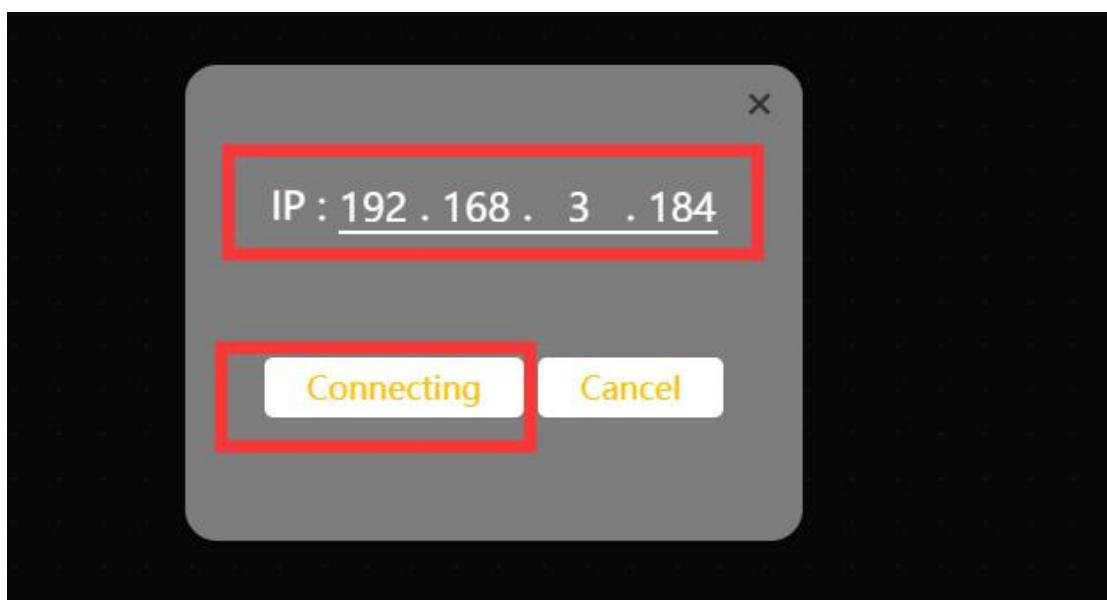
After successfully entering the website, the interface is as follows:



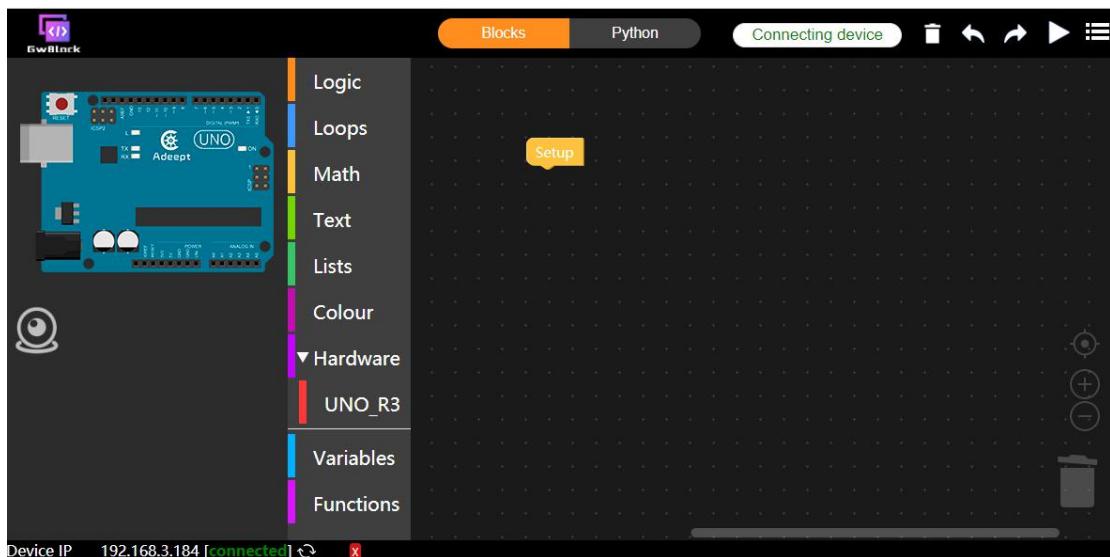
6. Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:

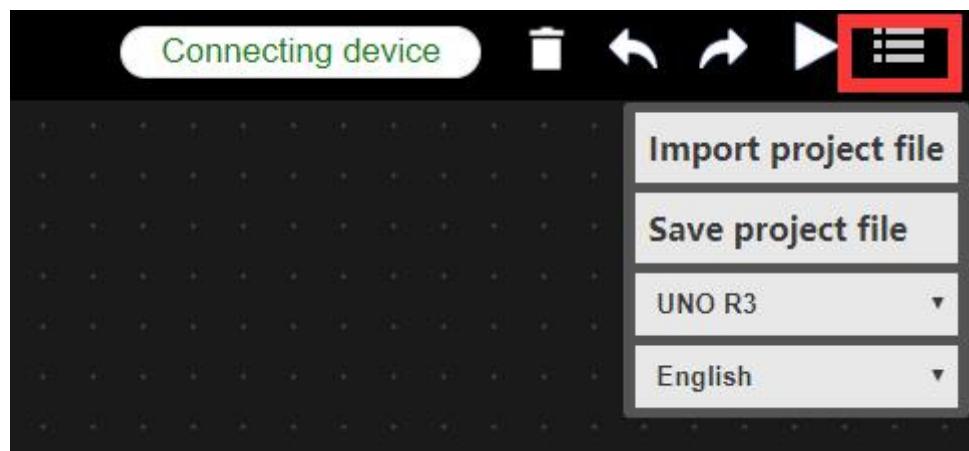


(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button  in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

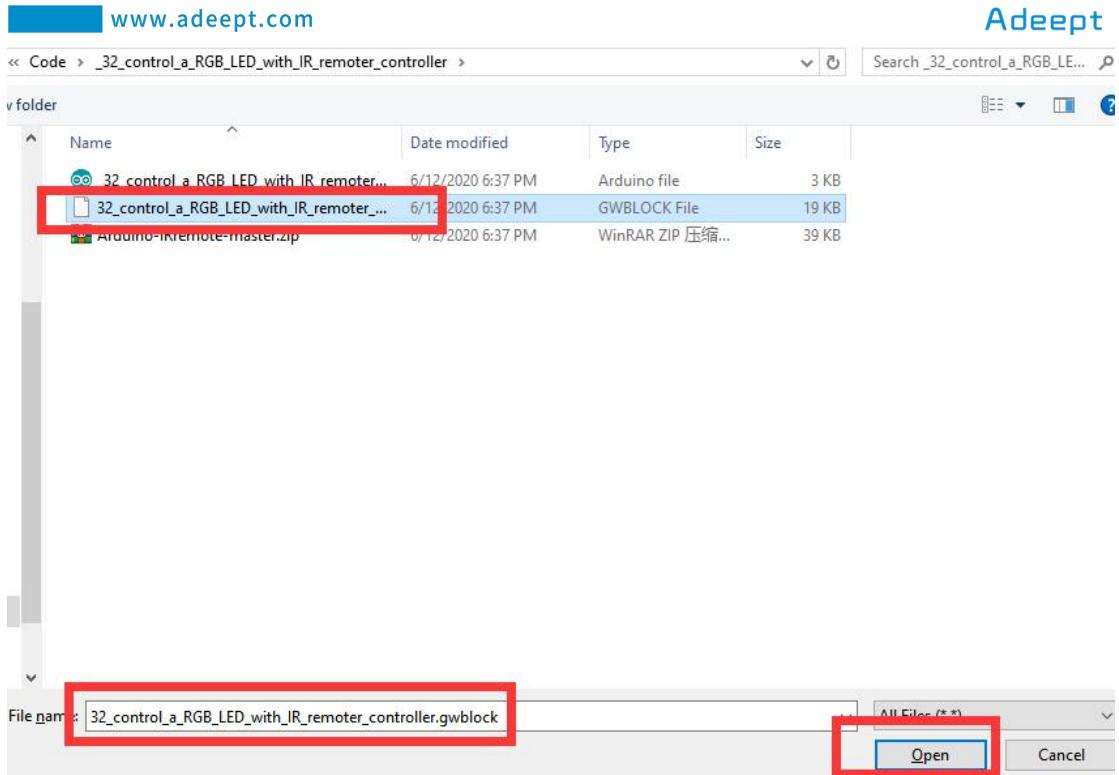
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_32_control_a_RGB_LED_with_IR_remoter_controller

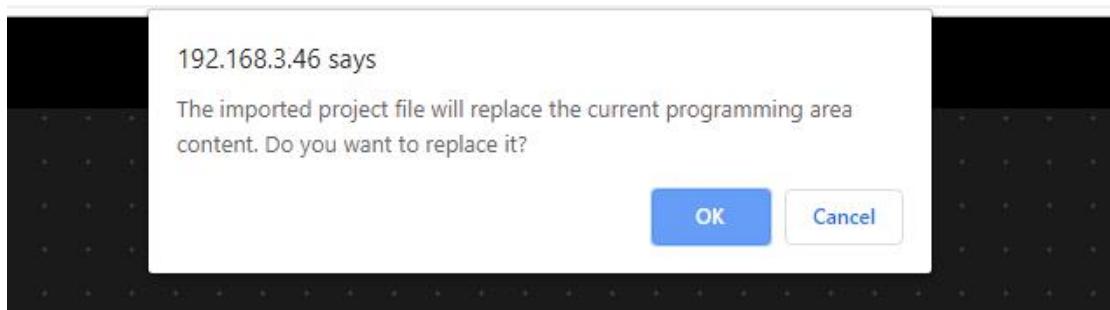
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

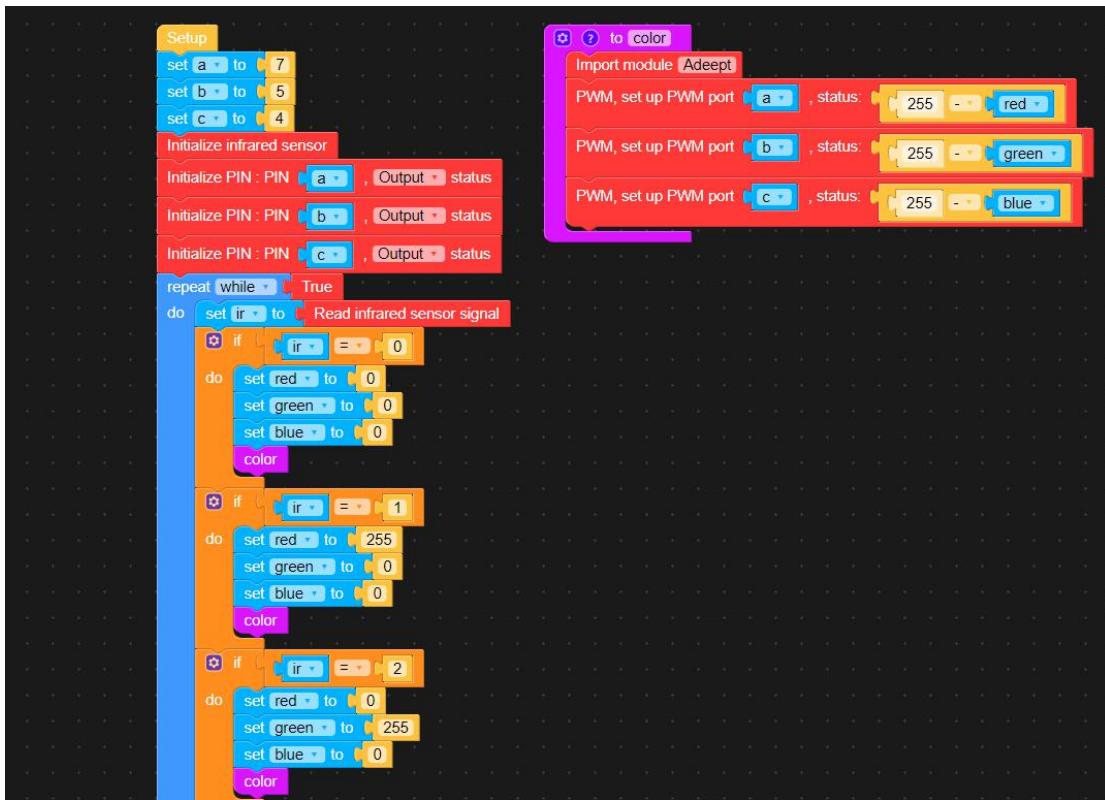
5. Select the "32_control_a_RGB_LED_with_IR_remoter_controller.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



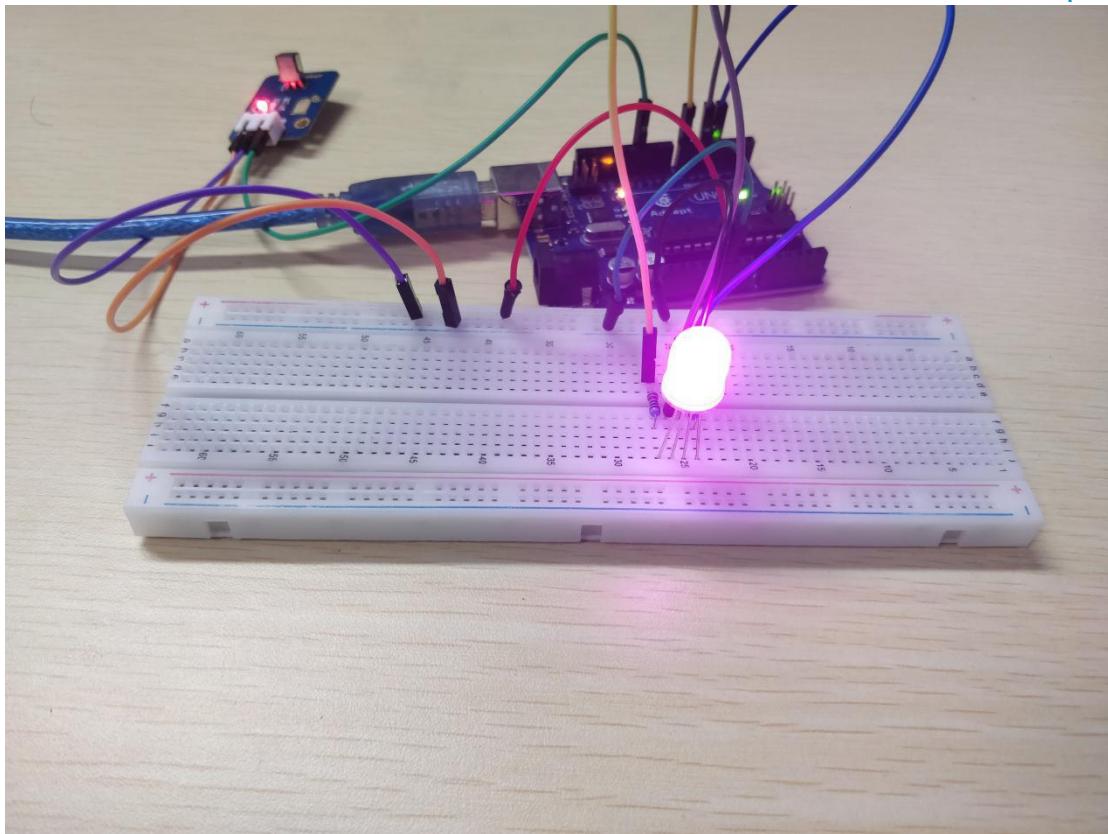
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



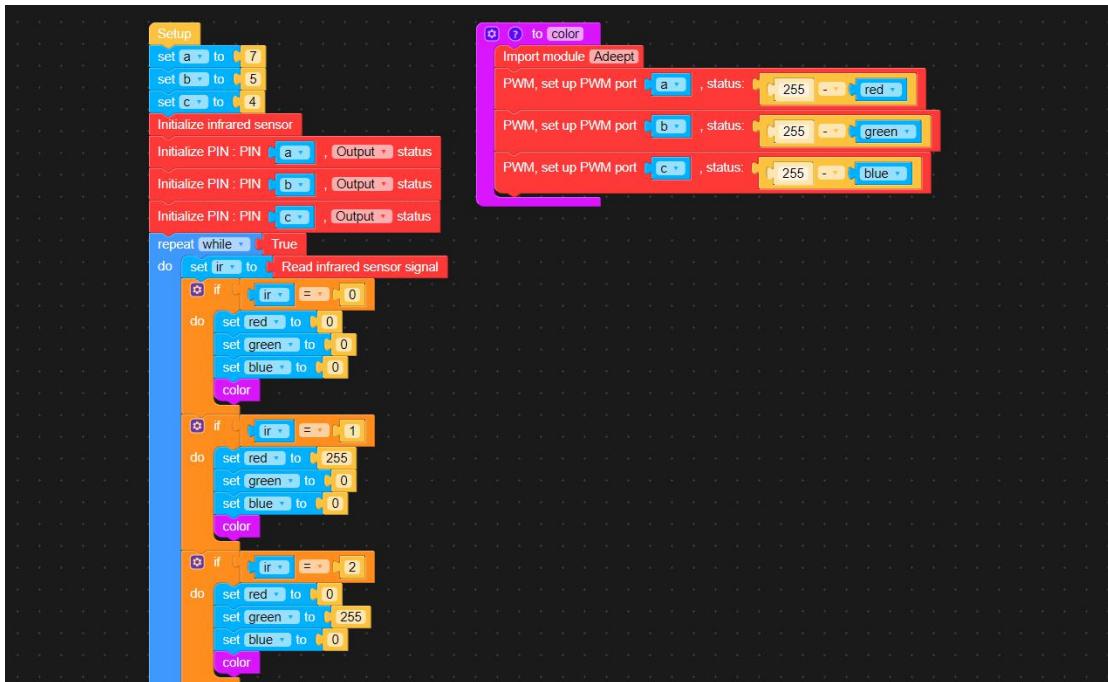
8. Click the button  on the upper right corner. When you press the number from 0 to 9 on the remote controller, the RGB LED color will change, indicating that our experimental test is successful. The physical connection diagram of the experiment is as follows:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to use IR Remoter Controller to control RGB LED on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

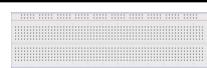
In the GwBlock graphical editor, all code programs are executed from **Setup**. First, you need to use the command **Initialize infrared sensor** to initialize the infrared receiving sensor; at the same time, in the loop, first use the command **Read infrared sensor signal** to read the signal value of the infrared receiver, and then use the command **[ir] = 0** to determine whether the button 0 of the remote controller is clicked. If you click the button 0, the command  will modify the color of RGB LED.



Lesson 33 Controlling Stepper Motor with IR Remoter Controller

In this lesson, we will learn how to use IR Remoter Controller to control stepper motor.

1. Components used in this course

Components	Quantity	Picture
Arduino UNO	1	
Breadboard	1	
USB Cable	1	
jumper wire	Several	
ULN2003-based Stepper Motor Driver	1	
Stepper Motor	1	
IR Receiver HX1838	1	
Remote Controller	1	
Male to Female Jumper Wires	Several	

2. IR Remoter Controller and Stepper Motor

(1) IR Remoter Controller

Please refer to Lesson 17, for we have introduced the IR remoter controller in detail

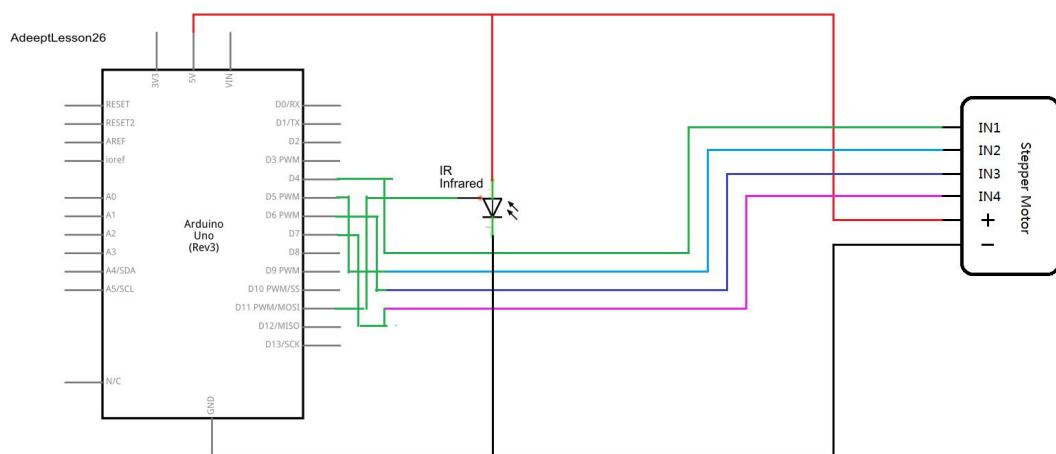
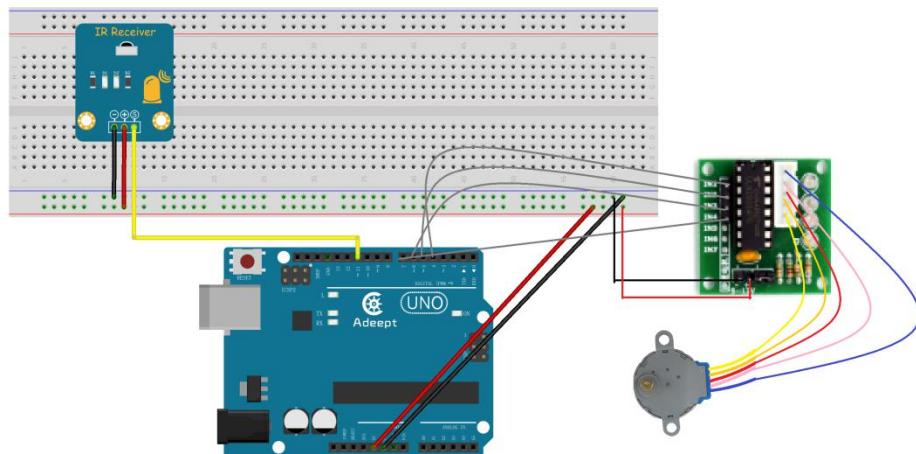
in Lesson 17.

(2) Stepper Motor

Please refer to Lesson 26, for we have introduced the Stepper Motor in detail in Lesson 26.

3.Wiring diagram (Circuit diagram)

Before the experiment, we connect them in the circuit as shown in the following figure. When connecting the circuit, we should pay attention to the difference between positive and negative electrodes. As shown in the following figure:



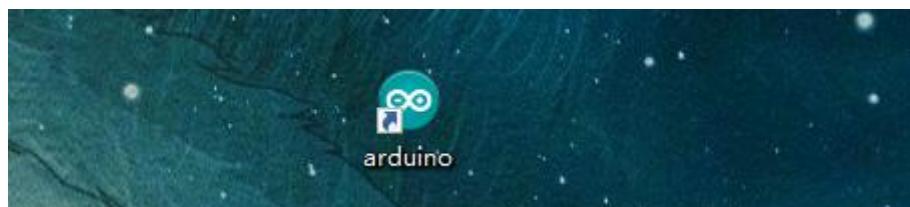
4. Using IR Remoter Controller to control Stepper Motor

We provide two different methods to use IR Remoter Controller to control stepper motor. One is to program on the Arduino UNO with C language through the Arduino IDE. You need to master the C language; the other is to program on the Arduino UNO with the graphical code block through GwBlock IDE. This method is very suitable for beginners who have not mastered the advanced programming languages C and C++. We will introduce these methods respectively.

1. Using C language to program to use IR Remoter Controller to control stepper motor on Arduino UNO

(1) Compile and run the code program of this course

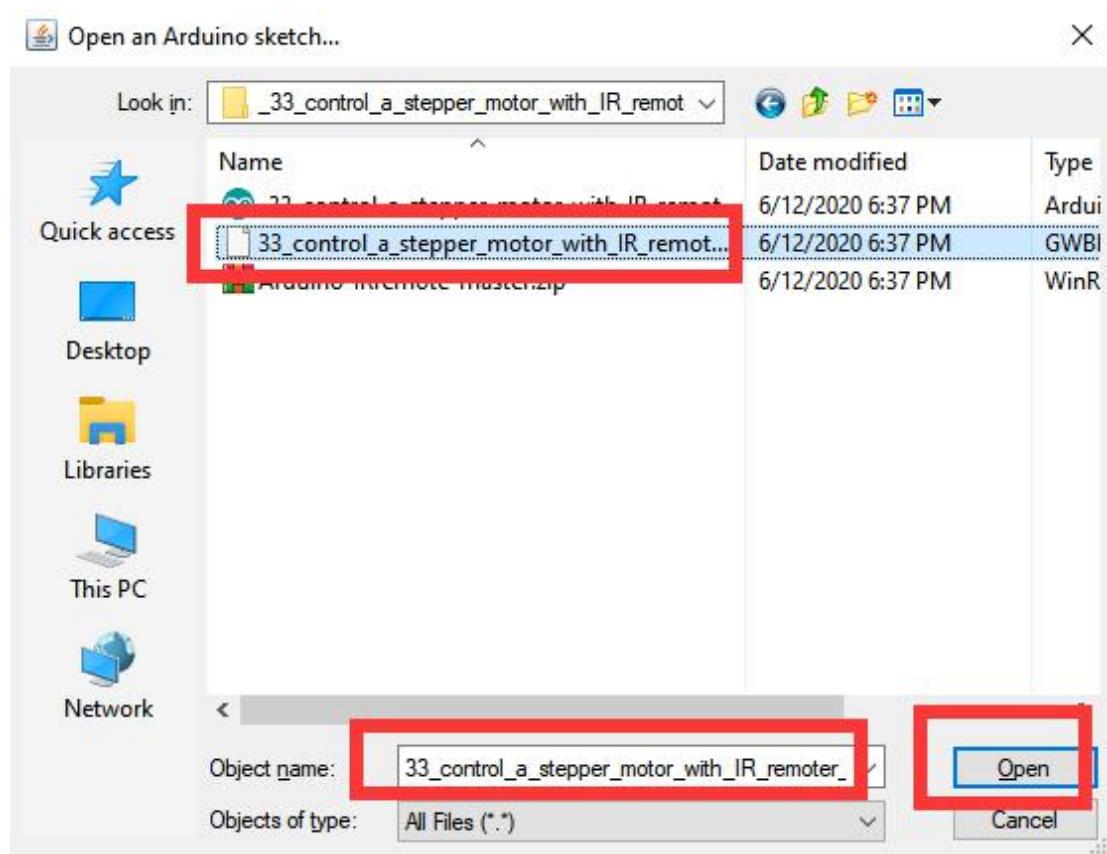
1. Open the Arduino IDE software, as shown below:



2. Click Open in the File drop-down menu:



3. Find the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 that we provide to the user. Open the folder Adeept_Ultimate_Kit_For_Arduino_C_Code in it. Enter the Code\33_control_a_stepper_motor_with_IR_remoter_controller directory. Select 33_control_a_stepper_motor_with_IR_remoter_controller.ino. This file is the code program we need in this course. Then click Open.



4. After opening, click  to upload the code program to the Arduino UNO. If there is no error warning in the console below, it means that the Upload is successful.

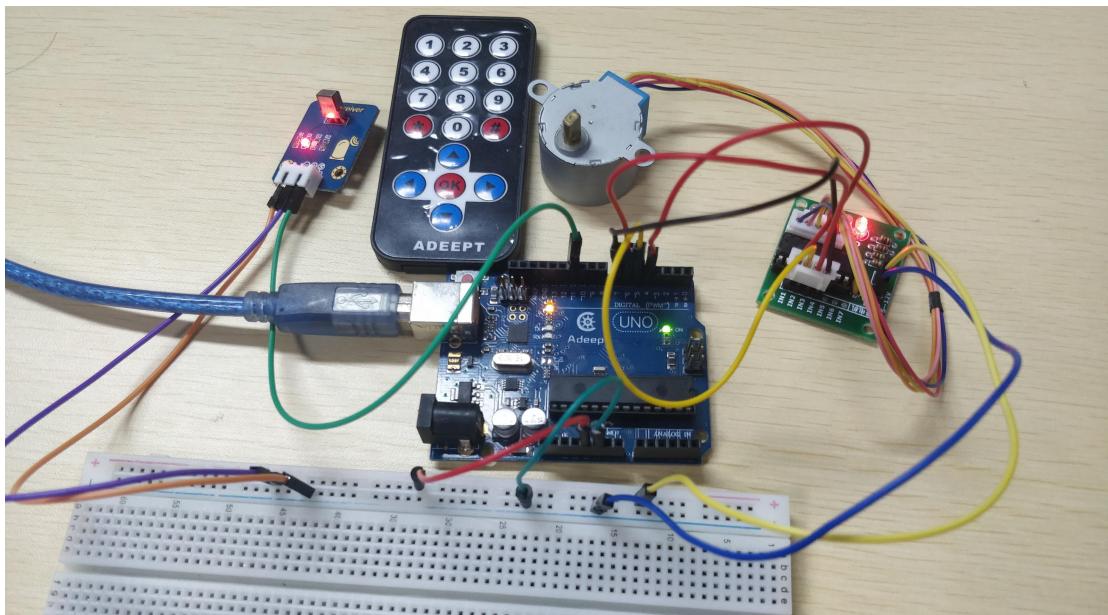
```
Done uploading.

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.

1
Arduino Uno on COM4
```

5. After successfully running the program, we set the numbers from 0 to 9 on the remote controller to control the state of the stepper motor in the program. When you press the numbers from 1 to 9 on the remote controller, the stepper motor will move. When you press 0, the stepper motor will stop. The physical connection diagram of

the experiment is as follows:



(2)Core code program

After the above hands-on operation, you must be very interested to know how we use C language to program to use IR Remoter Controller to control stepper motor on Arduino UNO. We will introduce how our core code can be achieved:

1.In the setup() function, set the Pin0, Pin1, Pin2, Pin3 pins to OUTPUT mode with the pinMode() function; use the irrecv.enableIRIn() function to initialize the infrared receiver.

```
void setup()
{
    pinMode(Pin0, OUTPUT); //Set digital 4 port mode, the OUTPUT for the output
    pinMode(Pin1, OUTPUT); //Set digital 5 port mode, the OUTPUT for the output
    pinMode(Pin2, OUTPUT); //Set digital 6 port mode, the OUTPUT for the output
    pinMode(Pin3, OUTPUT); //Set digital 7 port mode, the OUTPUT for the output
    irrecv.enableIRIn(); //Initialization infrared receiver
}
```

2.In the loop() function, judge the key value received by the infrared receiver with the switch statement and control the stepper motor to rotate. If you press button 1 on IR remoter controller, the stepper motor will rotate clockwise at a step speed of 15;

3.Press the "*" button on the remote controller to change the direction of rotation of the stepper motor. At this time, press the key value 1-9 on the remote controller, the stepper motor will rotate counterclockwise; if you press the "#" button, you can

control the stepper motor to rotate clockwise. At this time, press the button value 1-9 on the remote controller, the stepper motor will rotate clockwise.;

```

void loop()
{
    if (irrecv.decode(&results)) {
        switch(results.value){
            case 16738455/**/:
                dir = 1;
                break;
            case 16756815/**/:
                dir = 0;
                break;
            case 16750695/**/:
                Speed(15); //Stepper motor speed = 1 slow (note:speed from 1 to 15)
                Step(0); //Stepper motor stop
                break;
            case 16753245/**/:
                Speed(15); //Stepper motor speed = 15 fast (note:speed from 1 to 15)
                if(dir==0){
                    Step(512); //Stepper motor forward 512 steps ---- 360 angle
                }else{
                    Step(-512); //Stepper motor backward 512 steps ---- 360 angle
                }
                break;
            case 16736925/**/:
                Speed(15); //Stepper motor speed = 15 fast (note:speed from 1 to 15)
                if(dir==0){
                    Step(1024); //Stepper motor forward 1024 steps ---- 2*360 angle
                }else{
                    Step(-1024); //Stepper motor backward 1024 steps ---- 2*360 angle
                }
                break;
        }
    }
}

```

2. Using graphical code blocks to program to control stepper motor with IR Remoter Controller on Arduino UNO

(1)Connecting to GwBlock graphical editor

In the previous course "graphical programming of Arduino", we have introduced in detail how to connect GwBlock. Here we will briefly explain the steps.

1. Open the directory of the folder we provide to the user: Adeept_Ultimate_Kit_For_Arduino_V2_0\block_py. Double-click to open the block_py.ino file (open with Arduino). Then click  and upload the program to the Arduino UNO. After successful Upload, it will show as below:

www.adeept.com

block_py | Arduino 1.8.12

File Edit Sketch Tools Help

block_py

```

/*
 * KEY4x4 and MPU6050 cannot be 1 at the same time. Otherwise, insufficient capacity will be indicated.
 * When BUZZER is 1, the infrared sensor cannot be used, but the BUZZER can be used
 * When BUZZER is 0, the infrared sensor can be used, but the BUZZER cannot be used
 * (you can change these values as you apply them)
 */
#define KEY4x4 0
#define MPU6050 1
#define BUZZER 0

#include <ArduinoJson.h>
#if !BUZZER
#include <IRremote.h>
#endif
#include <Servo.h>
<
```

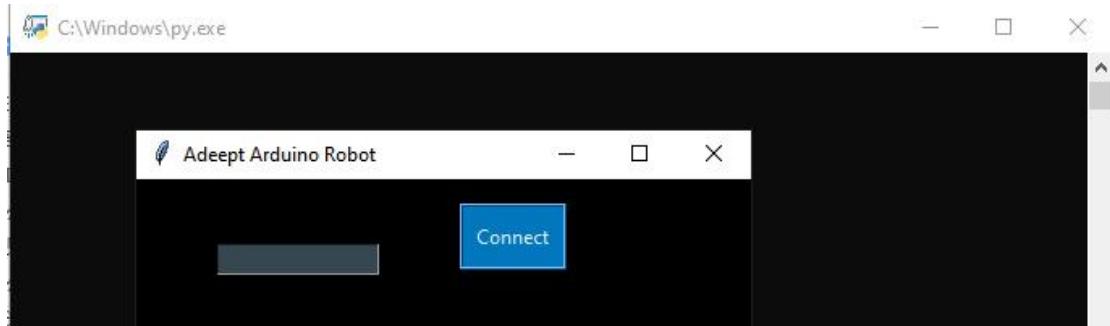
Done uploading.

Sketch uses 27154 bytes (84%) of program storage space. Maximum is 32256 bytes.

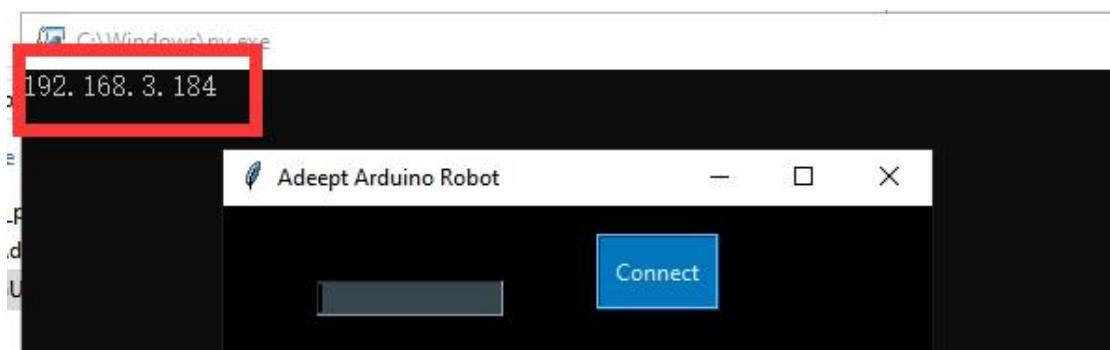
Global variables use 1525 bytes (74%) of dynamic memory, leaving 523 bytes for local variables. Maximum is 1023 bytes.

1 Arduino Uno on COM4

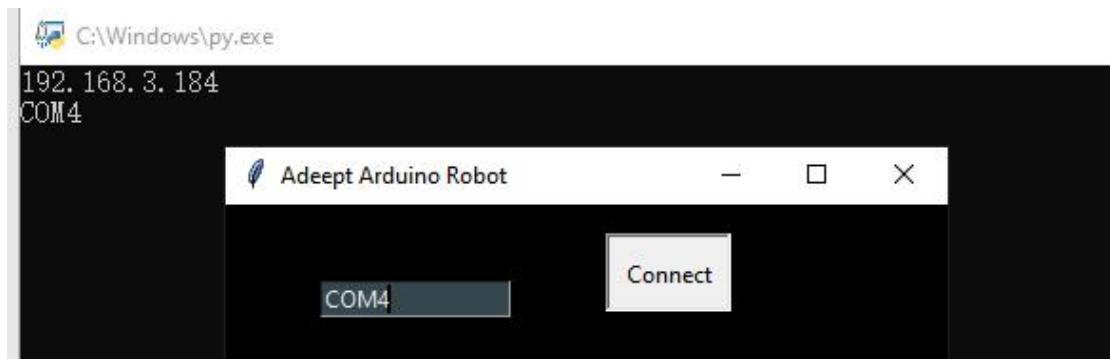
2. Open the folder Adeept_Ultimate_Kit_For_Arduino_V2_0 again. Then open the websocket folder. Double-click to open the GUI info v1.0.py file. It will show as below:



3. Remember the IP address: 192.168.3.184 (each user's IP is different) which will be used later. It will show as below:



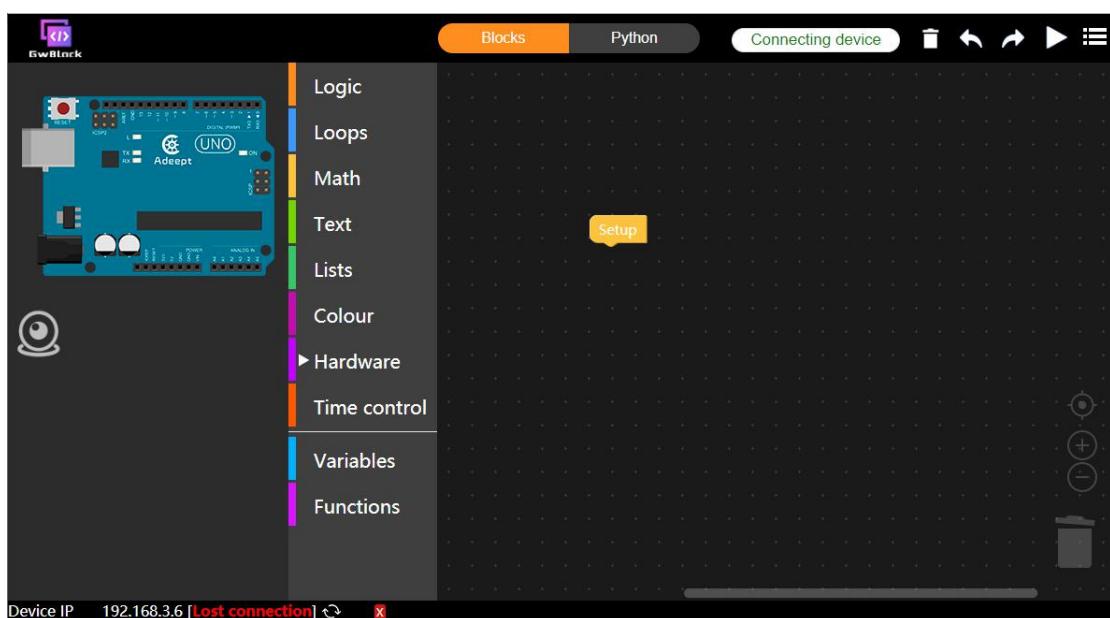
4.Enter the connected port number: COM4 in the input box of Adeept Arduino Robot, and click the Connect button. It will show as below:



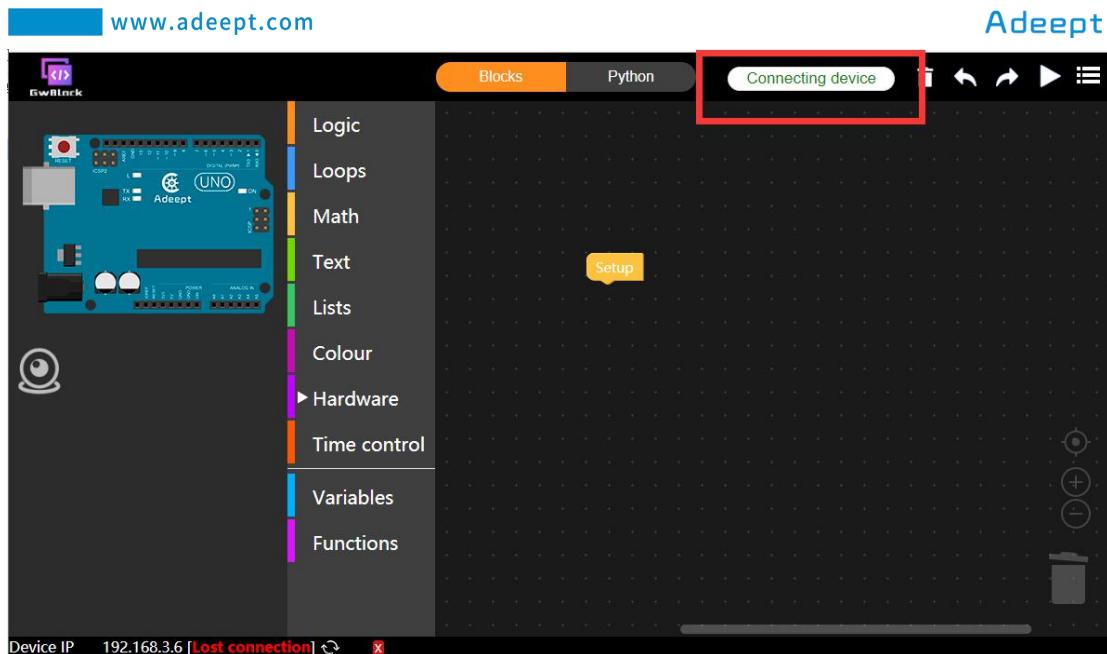
5.Enter the URL of the GwBlock graphical editor in the browser:

http://www.adeept.com/gwblock/?hd_mo=uno_r3.

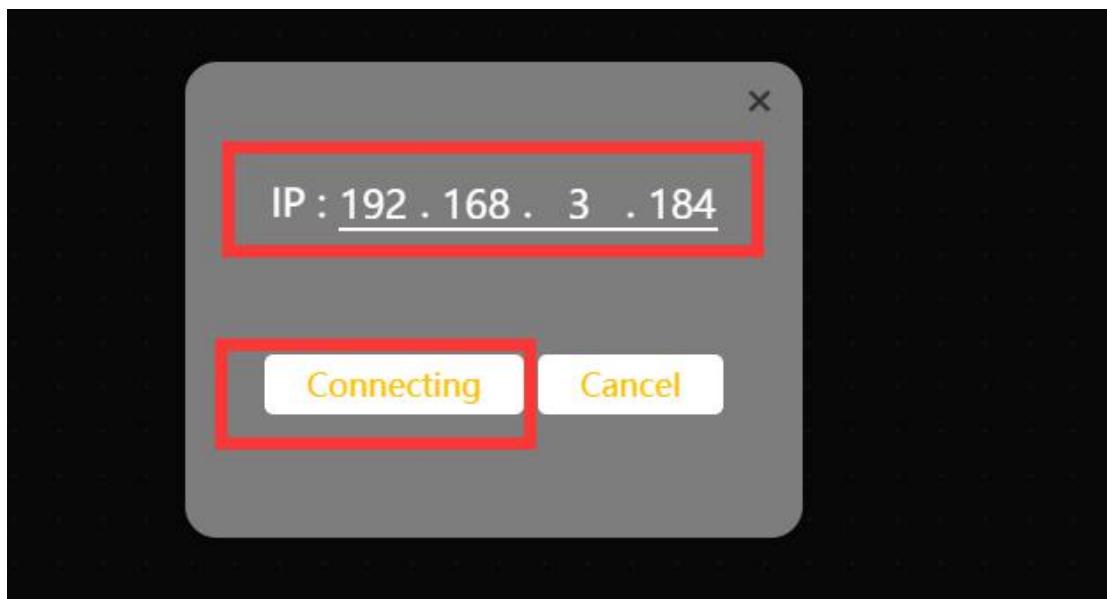
After successfully entering the website, the interface is as follows:



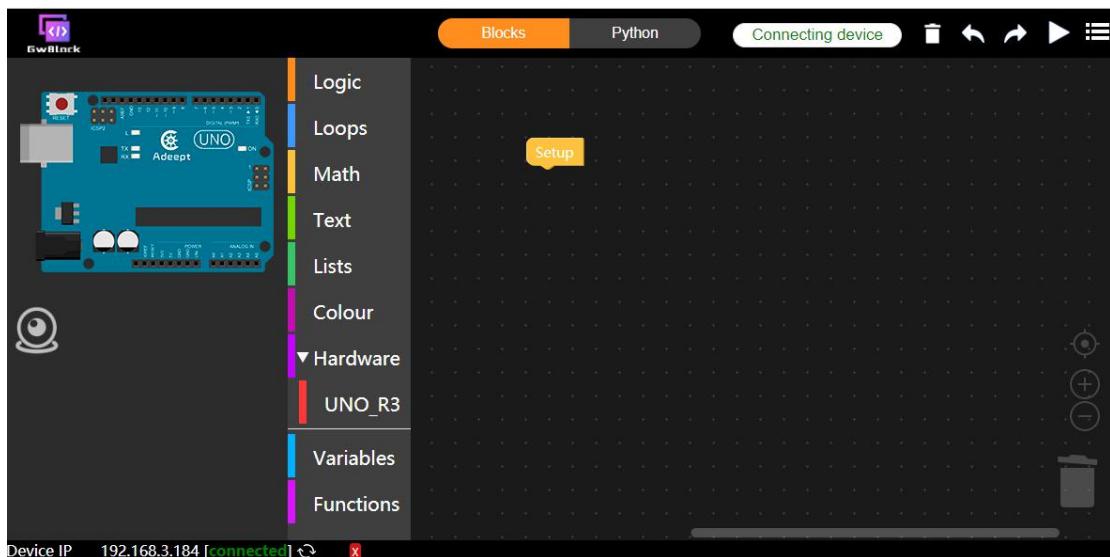
6.Click the "Connecting device" button in the upper right corner. It will show as below:



7.In the pop-up box, enter the IP address recorded in step 3: 192.168.3.184.Then click the Connecting . It will show as below:



8.After the successful connection, a green connected prompt will appear in the lower left corner. It Indicates that we have successfully connected to the GwBlock graphical editor. It will show as below:



(2)Run the code program for this course

Now let us learn how to use the GwBlock graphical editor to open and run the program for the course.

(1)Open and run the program for this course

1. After successfully connecting to the GwBlock graphical editor, click the button

 in the upper right corner. It will show as below:



2. Click Import project file to import the external project file. After opening, a blank page will appear. You need to modify the lower right corner area and select All Files. It will show as below:



3. The file will then be displayed. It will show as below:

Name	Date modified	Type
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder
Arduino libraries	6/4/2020 3:52 PM	File folder
block_py	6/8/2020 5:13 PM	File folder
websocket	6/9/2020 10:37 AM	File folder

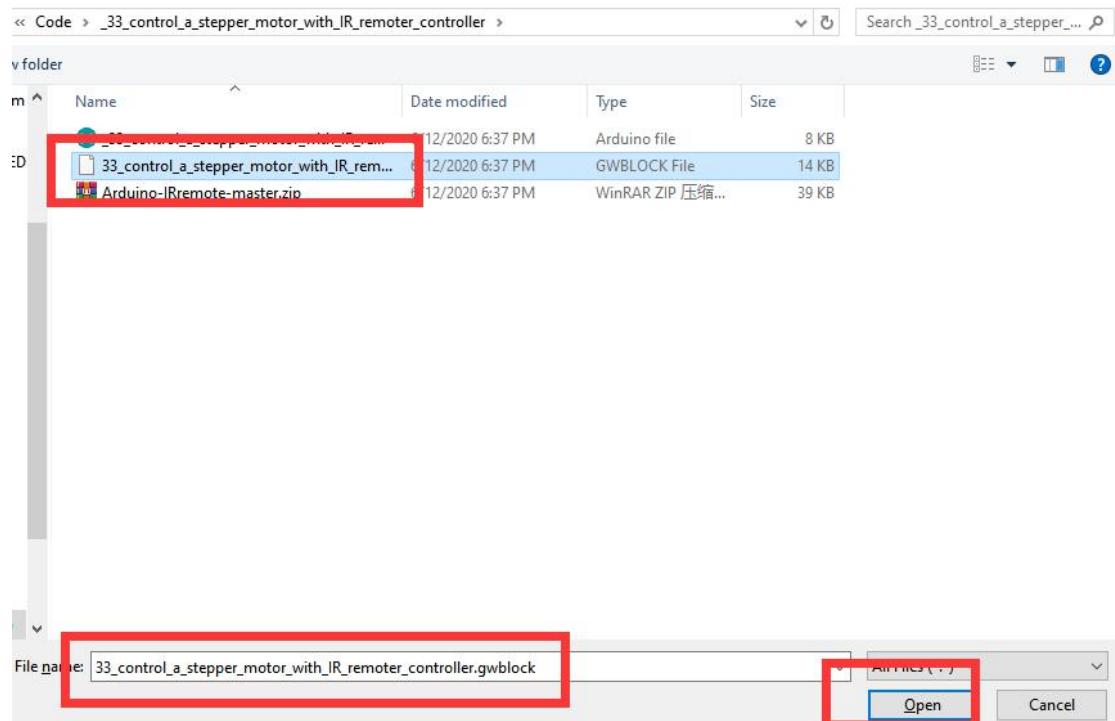
4. Open the Adeept_Ultimate_Kit_For_Arduino_V2_0 file that we provide to users, and then open the folder:

Adeept_Ultimate_Kit_For_Arduino_C_Code\Code_33_control_a_stepper_motor_with_IR_remoter_controller

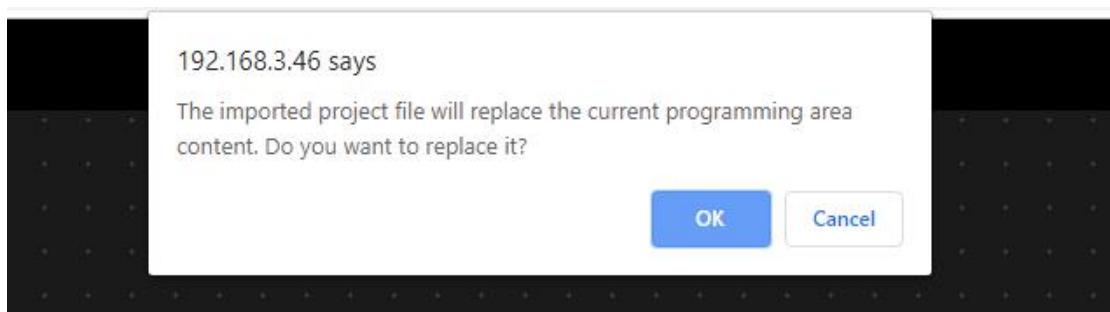
It will show as below:

Name	Date modified	Type	Size
Adeept_Ultimate_Kit_For_Arduino_C_Code	6/6/2020 1:22 PM	File folder	
Arduino libraries	6/4/2020 3:52 PM	File folder	
block_py	6/8/2020 5:13 PM	File folder	

5. Select the "_33_control_a_stepper_motor_with_IR_remoter_controller.gwblock" file. This file is the graphical code program for our lesson. Click "Open" in the lower right corner. It will show as below:



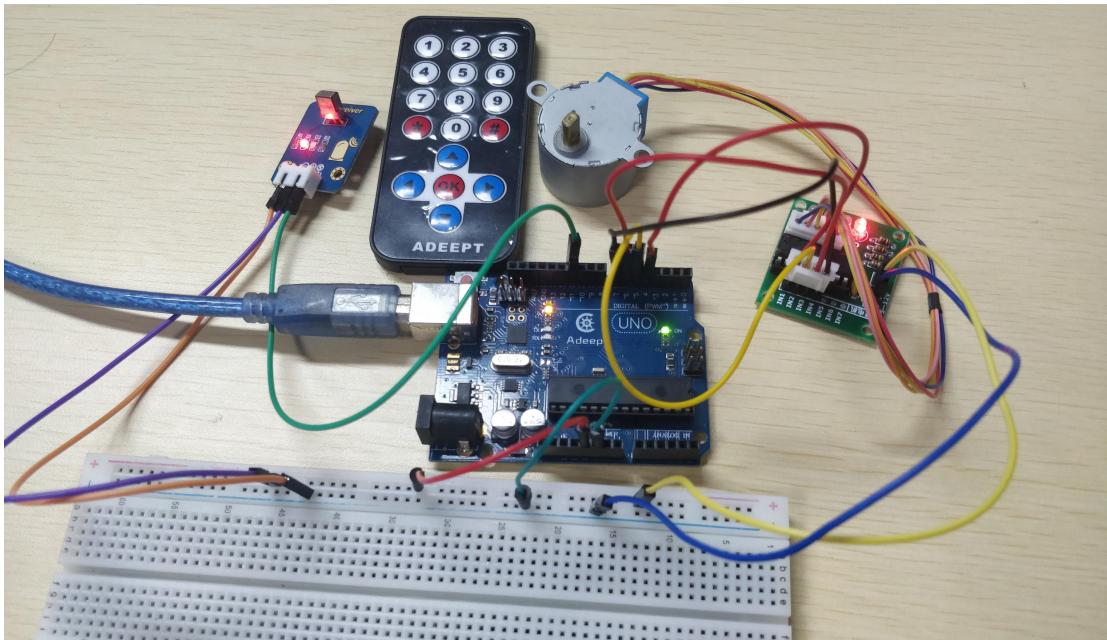
6.Click OK.It will show as below:



7.It will show as below after successfully opening:



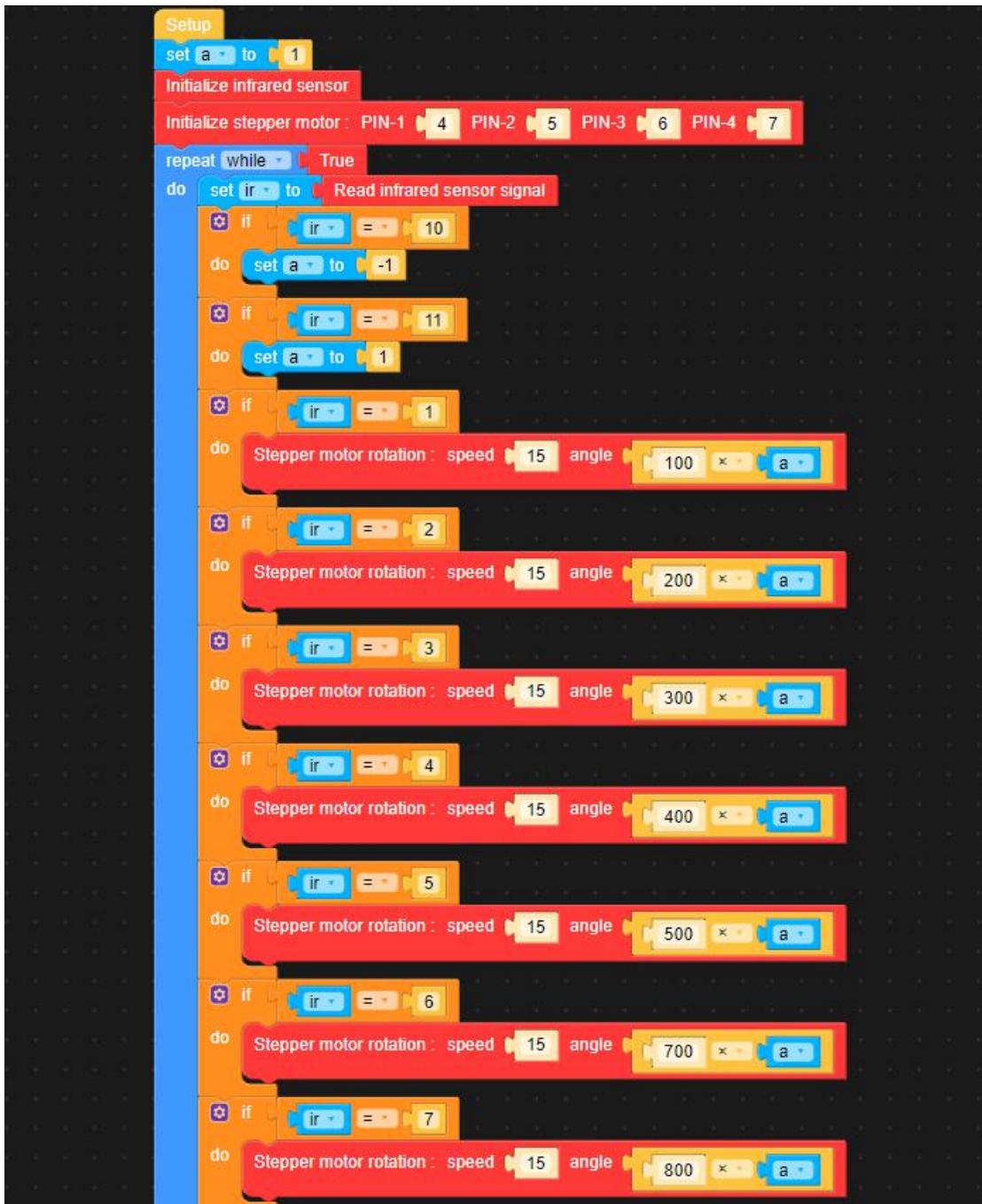
8. Click the button  on the upper right corner. When you press the numbers from 1 to 9 on the remote controller, you can control the stepper motor to rotate, indicating that our experimental test is successful. The physical connection diagram of the experiment is as follows:



(3)Core code program

After the above hands-on operation, you must be very interested to know how we program to use IR Remoter Controller to control stepper motor on Arduino UNO with graphical code blocks. We will introduce how our core code can be achieved:

In the GwBlock graphical editor, all code programs are executed from **Setup**. First, you need to use the command **Initialize infrared sensor** to initialize the infrared receiving sensor; in the while loop, first use the command **Read infrared sensor signal** to read the signal value of the infrared receiver, and use the command **ir = 10** to judge if the button "*" is pressed, then the stepper motor is controlled to rotate counterclockwise with the command **set a to -1**; use the command **ir = 11** to judge if the button "#" is pressed, the stepper motor is controlled to rotate clockwise with the command **set a to 1**; at this time, if you press the button 1 of the remote controller, the command **Stepper motor rotation : speed 15 angle 100 * a** will control the stepper motor to rotate counterclockwise or clockwise at a step speed of 15.



Lesson 34 Controlling the Size of a Circle by Potentiometer

1. Overview

In this lesson, we will collect the potentiometer data by programming the Arduino UNO Board, and then send the data to the Processing through serial communication to change the size of a circle.

3. Components

- 1 * Arduino UNO
- 1 * USB Cable
- 1 * 10kΩ Potentiometer
- 1 * Breadboard
- Several jumper wires

3. Principle

The experiment consists of two parts: first, acquire the data from Arduino; second, process the data.

Arduino key function:

•`write()`

Writes binary data to the serial port. This data is sent as a byte or series of bytes; to send the characters representing the digits of a number use the `print()` function instead.

Syntax:

`Serial.write(val)`

`Serial.write(str)`

Serial.write(buf, len)

Parameters:

val: a value to send as a single byte

str: a string to send as a series of bytes

buf: an array to send as a series of bytes

len: the length of the buffer

Returns:

byte

write() will return the number of bytes written, though reading that number is optional

Processing key function:

•Name: size()

Description:

Defines the dimension of the display window in units of pixels. The size() function must be the first line of code, or the first code inside setup(). Any code that appears before the size() command may run more than once, which can lead to confusing results.

The system variables width and height are set by the parameters passed to this function. If size() is not used, the window will be given a default size of 100x100 pixels.

Syntax:

size(w, h)

size(w, h, renderer)

Parameters:

w int: width of the display window in units of pixels

h int: height of the display window in units of pixels

renderer

String: Either P2D, P3D, or PDF

Returns:

void

- Name: `println()`

Description:

The `println()` function writes to the console area, the black rectangle at the bottom of the Processing environment. This function is often helpful for looking at the data a program is producing. Each call to this function creates a new line of output. More than one parameter can be passed into the function by separating them with commas. Alternatively, individual elements can be separated with quotes ("") and joined with the addition operator (+).

Before Processing 2.1, `println()` was used to write array data to the console. Now, use `printArray()` to write array data to the console.

Note that the console is relatively slow. It works well for occasional messages, but does not support high-speed, real-time output (such as at 60 frames per second).

Syntax:

`println()`

`println(what)`

`println(variables)`

Parameters:

what Object, String, float, char, boolean, or byte: data to print to console

variables Object[]: list of data, separated by commas

Returns:

void

- Name: `background()`

Description:

The `background()` function sets the color used for the background of the Processing window. The default background is light gray. This function is typically used within `draw()` to clear the display window at the beginning of each frame, but it can be used inside `setup()` to set the background on the first frame of animation or if the background need only be set once.

An image can also be used as the background for a sketch, although the image's width and height must match that of the sketch window. Images used with `background()` will ignore the current `tint()` setting. To resize an image to the size of the sketch window, use `image.resize(width, height)`.

It is not possible to use the transparency alpha parameter with background colors on the main drawing surface. It can only be used along with a PGraphics object and `createGraphics()`.

Syntax:

`background(rgb)`

`background(rgb, alpha)`

`background(gray)`

`background(gray, alpha)`

`background(v1, v2, v3)`

`background(v1, v2, v3, alpha)`

`background(image)`

Parameters:

rgb int: any value of the color datatype

alpha float: opacity of the background

gray float: specifies a value between white and black

v1 float: red or hue value (depending on the current color mode)

v2 float: green or saturation value (depending on the current color mode)

v3 float: blue or brightness value (depending on the current color mode)

image PImage: PImage to set as background (must be same size as the sketch window)

Returns:

void

•**Name:** fill()

Description:

Sets the color used to fill shapes. For example, if you run fill(204, 102, 0), all subsequent shapes will be filled with orange. This color is either specified in terms of the RGB or HSB color depending on the current colorMode(). (The default color space is RGB, with each value in the range from 0 to 255.)

When using hexadecimal notation to specify a color, use "#" or "0x" before the values (e.g., #CCFFAA or 0xFFCCFFAA). The # syntax uses six digits to specify a color (just as colors are typically specified in HTML and CSS). When using the hexadecimal notation starting with "0x", the hexadecimal value must be specified with eight characters; the first two characters define the alpha component, and the remainder define the red, green, and blue components.

The value for the "gray" parameter must be less than or equal to the current maximum value as specified by colorMode(). The default maximum value is 255.

Syntax:

fill(rgb)

fill(rgb, alpha)

fill(gray)

fill(gray, alpha)

fill(v1, v2, v3)

fill(v1, v2, v3, alpha)

Parameters:

rgb int: color variable or hex value

alpha float: opacity of the fill

gray float: number specifying value between white and black

v1 float: red or hue value (depending on current color mode)

v2 float: green or saturation value (depending on current color mode)

v3 float: blue or brightness value (depending on current color mode)

Returns:

void

•**Name:ellipse()**

Description:

Draws an ellipse (oval) to the screen. An ellipse with equal width and height is a circle. By default, the first two parameters set the location, and the third and fourth parameters set the shape's width and height. The origin may be changed with the ellipseMode() function.

Syntax:

ellipse(a, b, c, d)

Parameters:

a float: x-coordinate of the ellipse

b float: y-coordinate of the ellipse

c float: width of the ellipse by default

d float: height of the ellipse by default

Returns:

Void

Note:

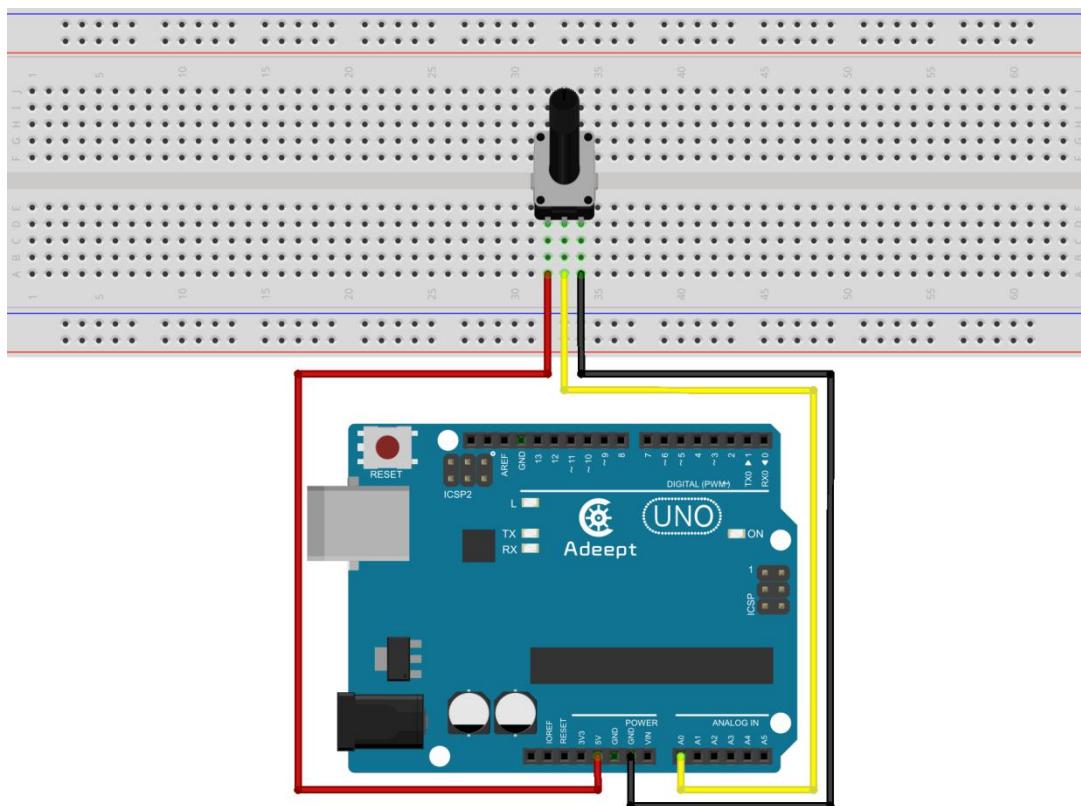
1. In this experiment, my Arduino UNO board is connected to my computer port

COM26. But it may differ in your case. So please adjust it according to your actual situation.

2. If Processing prompts that you need to install the related function library, please do it.

4. Procedures

Step 1: Build the circuit



Step 2: Program

Step 3: Compile the program and upload to Arduino UNO board

Step 4: Run the Processing software (Processing_Potentiometer.pde)

Now, turn the knob of the potentiometer, and you will see a blue circle size to change on the computer.

www.adeept.com

Processing_Potentiometer | Processing 3.0a10

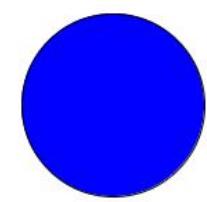
File Edit Sketch Debug Tools Help

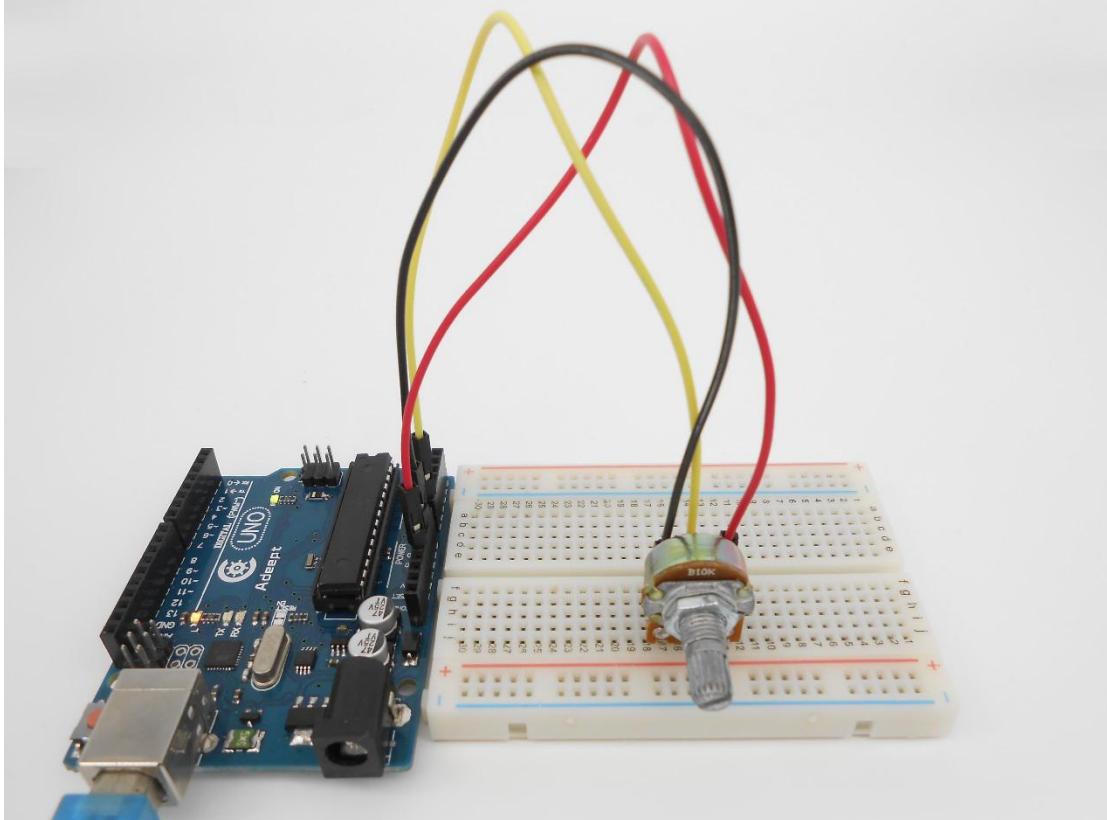
Java

Processing_Potentiometer

```
1 //////////////////////////////////////////////////////////////////
2 File name: Processing_Potentiometer.pde
3 Description: Arduino and processing interactive
4           The size of the potentiometer volta
5 Website: www.adeept.com
6 E-mail: support@adeept.com
7 Author: Tom
8 Date: 2015/05/02
9 //////////////////////////////////////////////////////////////////
10 import processing.serial.*;
11
12 Serial serial;
13 int sensorValue;
14
15 void setup() {
16   // set the canvas size is 400 x 400
17   size(400, 400);
18
19   // Open the serial port and set the baud rate
20   // This experiment arduino board is connected
21   // adjust according to actual situation.
22   serial = new Serial(this, "COM26", 9600);
23 }
24
25 void draw() {
26   if ( serial.available() > 0 ) {
27     // Read from serial production sensor value.
28     sensorValue = serial.read();
29   }
30 }
```

Console Errors





Lesson 35 Controlling the 3D Model by PS2 Joystick

1. Overview

In this lesson, we will collect the state of a joystick by programming the Arduino UNO Board, and then send the data to the Processing through the serial communication.

2. Components

- 1 * Arduino UNO
- 1 * USB Cable
- 1 * PS2 Joystick
- 1 * Breadboard
- Several jumper wires

3. Principle

The experiment consists of two parts: first, acquire the data from Arduino; second, process the data.

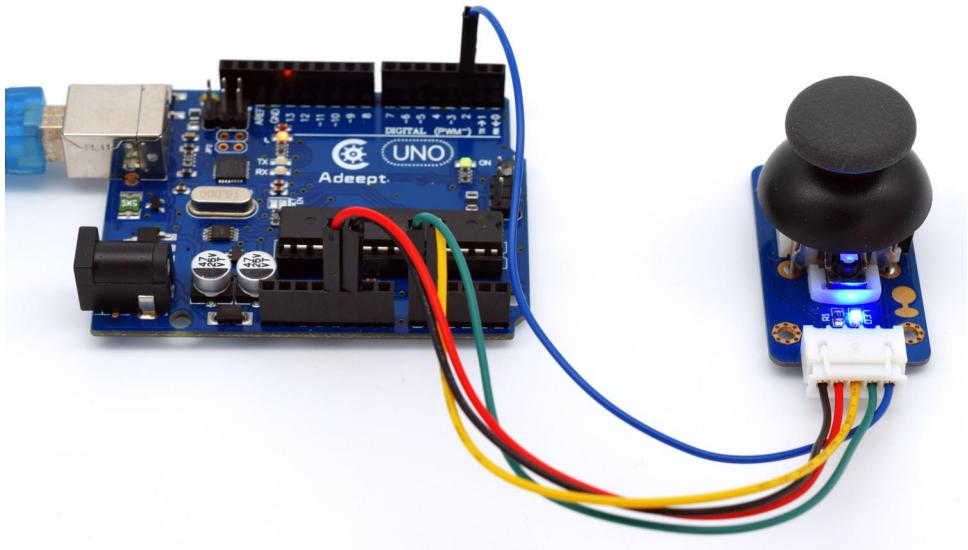
Here use the Arduino UNO board to collect data of the joystick state, and upload the data to the computer through the serial port. The data will be processed by Processing and shown with 3D image.

Note:

1. In this experiment, my Arduino UNO board is connected to my computer port COM26. But it may differ in your case. So please adjust it according to your actual situation.
2. If the Processing does not run normally, you may need to install the related function libraries.

4. Procedures

Step 1: Build the circuit

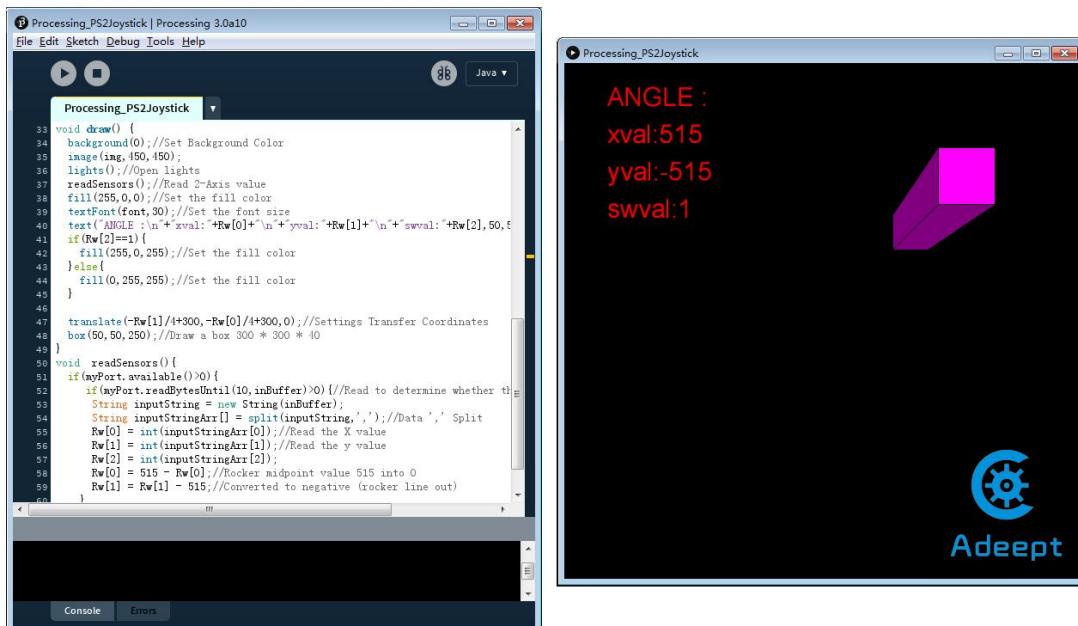


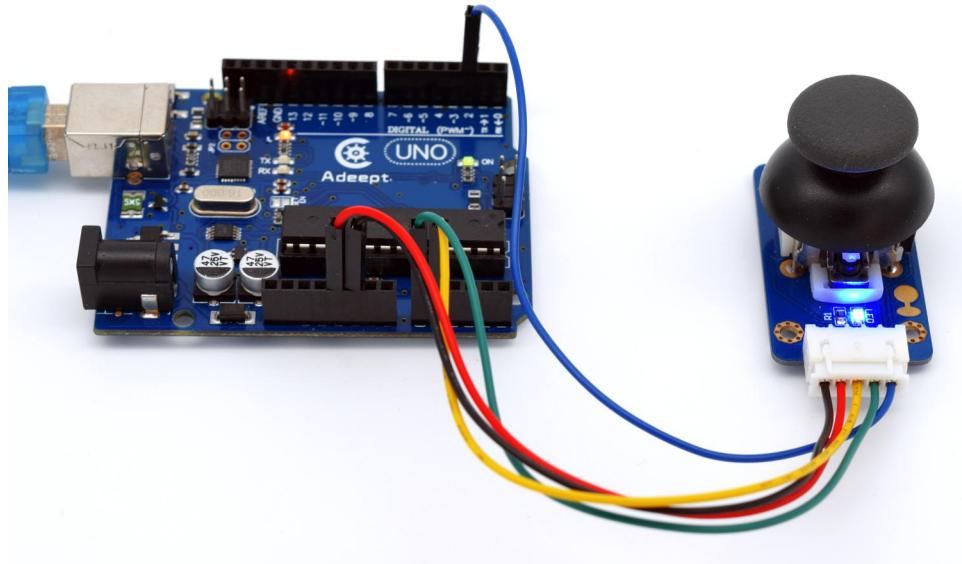
Step 2: Program

Step 3: Compile the program and upload to Arduino UNO board

Step 4: Run the Processing software (Processing_PS2Joystick.pde)

Move the joystick, and the 3D model will follow movement changes accordingly on your computer.





Lesson 36 Snake Game

1.Overview

In this lesson, we will make a Snake Game based on the Processing, and play the game with two buttons.

2.Components

- 1 * Arduino UNO
- 1 * USB Cable
- 2 * Button
- 1 * Breadboard
- Several jumper wires

3.Principle

The experiment consists of two parts: first, acquire the data from Arduino; second, process the data.

Play the Snake Game:

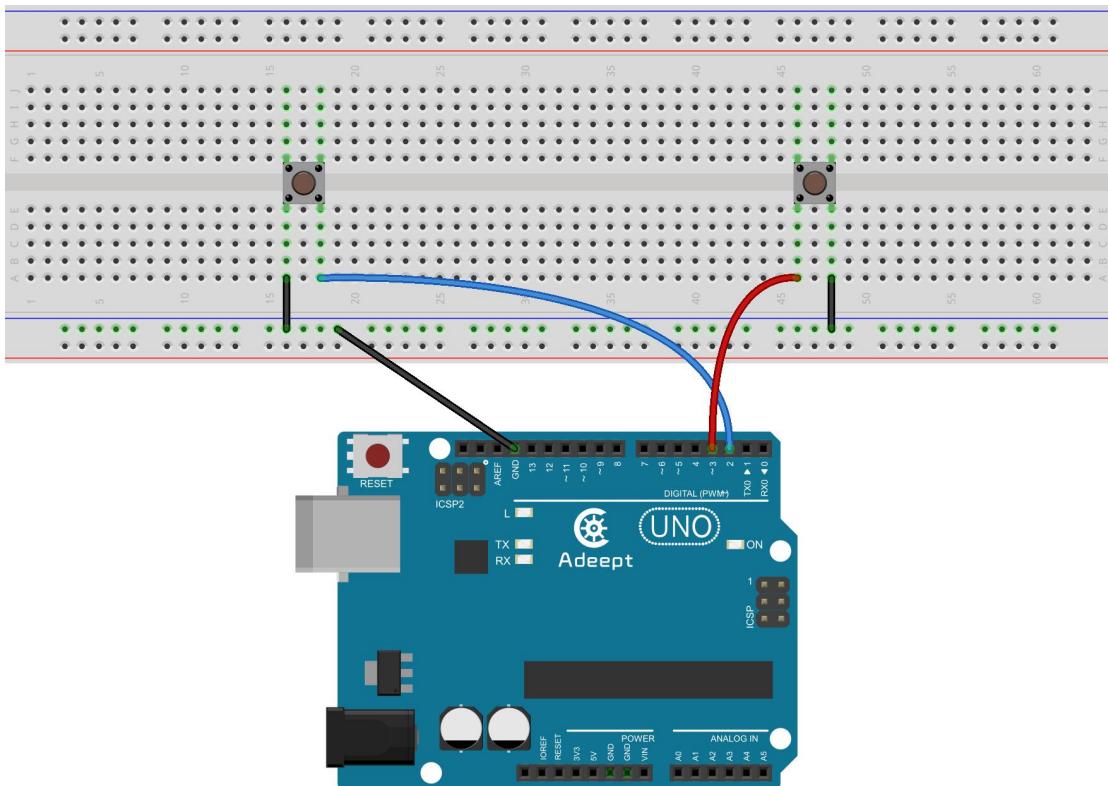
1. When you press the right button, the snake will move to the right.
2. Press the left button, and the snake will move to the left.

Note:

1. You need to install the Sound library.
2. In this experiment, my Arduino UNO board is connected to my computer port COM26. But it may differ in your case. So please adjust it according to your actual situation.
3. If the Processing does not run normally, you may need to install the related function libraries.

4.Procedures

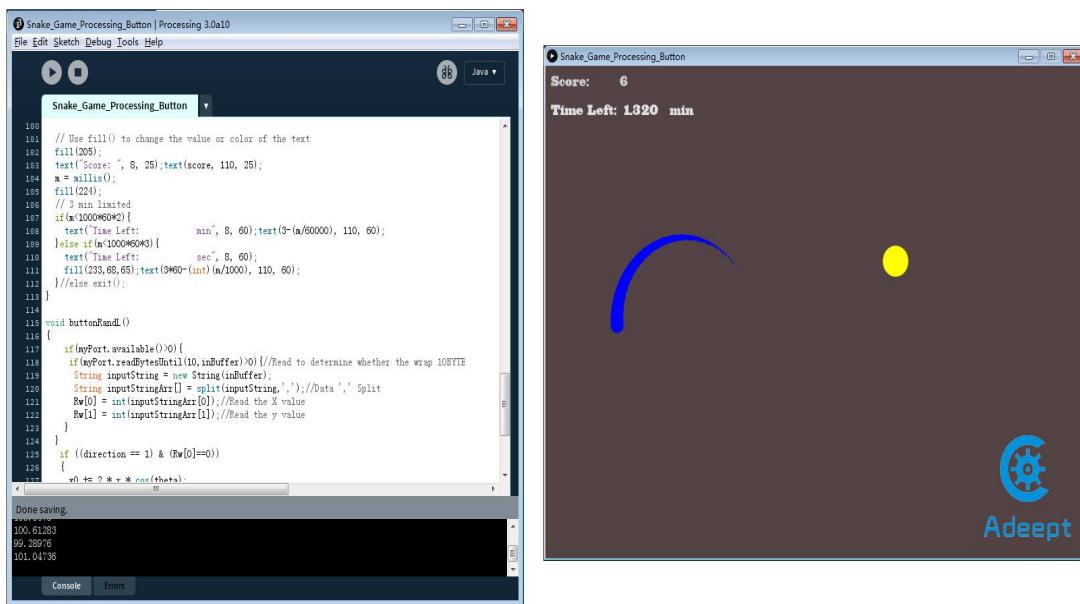
Step 1: Build the circuit

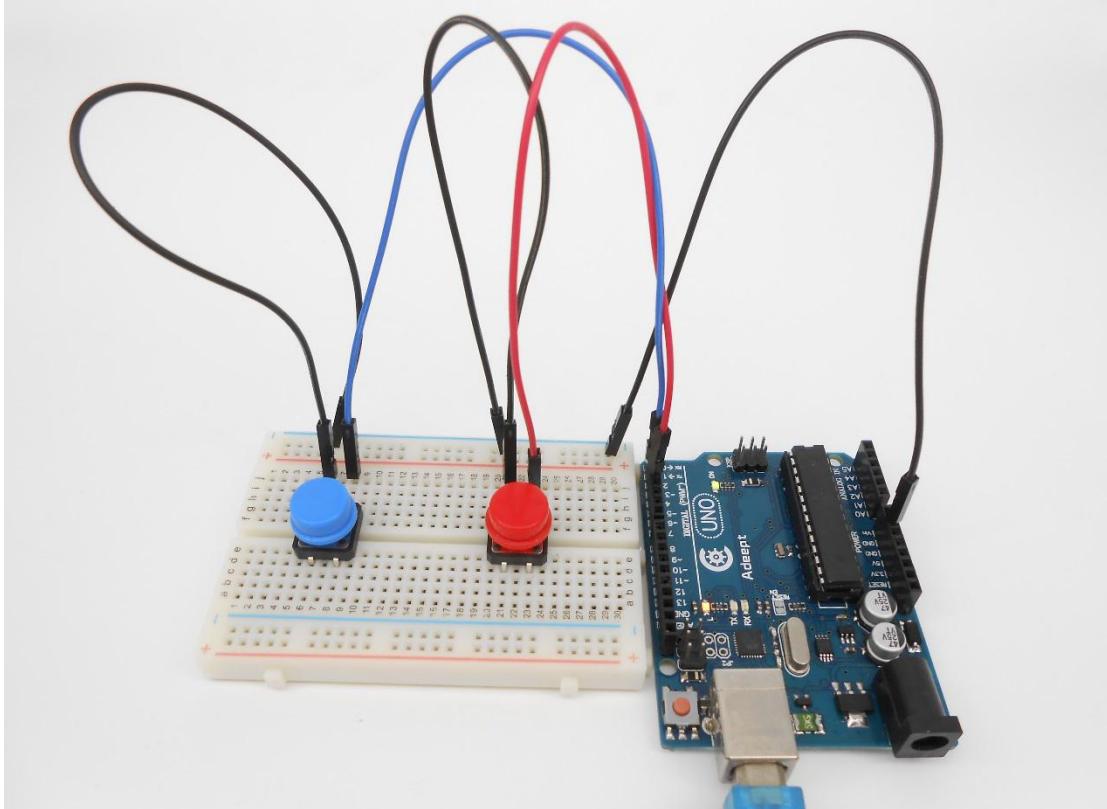


Step 2: Program

Step 3: Compile the program and upload to Arduino UNO board

Step 4: Run the Processing software (Snake_Game_Processing_Button.pde)





Lesson 37 Star Wars

1. Overview

In this lesson, we will make a star wars game based on the Processing and play the game with a PS2 joystick and a button connected to the Arduino UNO.

1. Components

- 1 * Arduino UNO
- 1 * USB Cable
- 1 * Button
- 1 * PS2 Joystick
- 1 * Breadboard
- Several jumper wires

4. Principle

The experiment consists of two parts: first, acquire the data from Arduino; second, process the data.

Play the Star Wars Game:

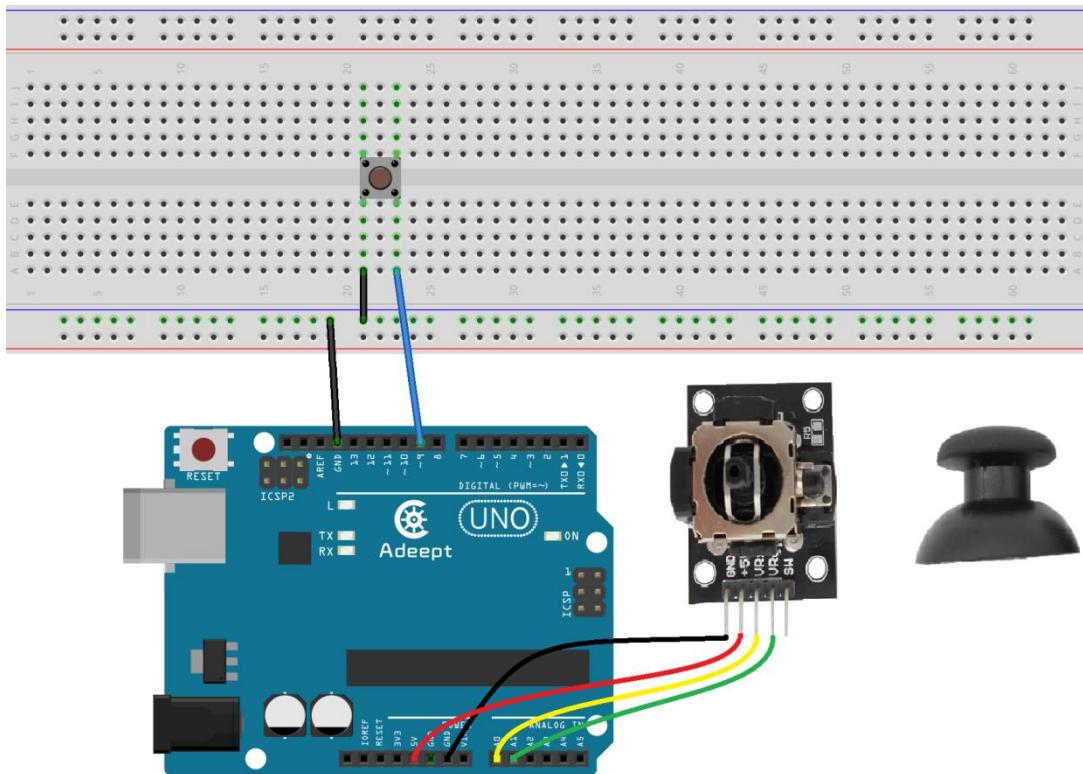
- ① When you press the button, the fighter aircraft will fire bullets
- ② When you operating the PS2 Joystick, you can change the position of movement of the fighter aircraft.

Note:

1. In this experiment, my Arduino UNO board is connected to my computer port COM26. But it may differ in your case. So please adjust it according to your actual situation.
2. If the Processing does not run normally, you may need to install the related function libraries.

5. Procedures

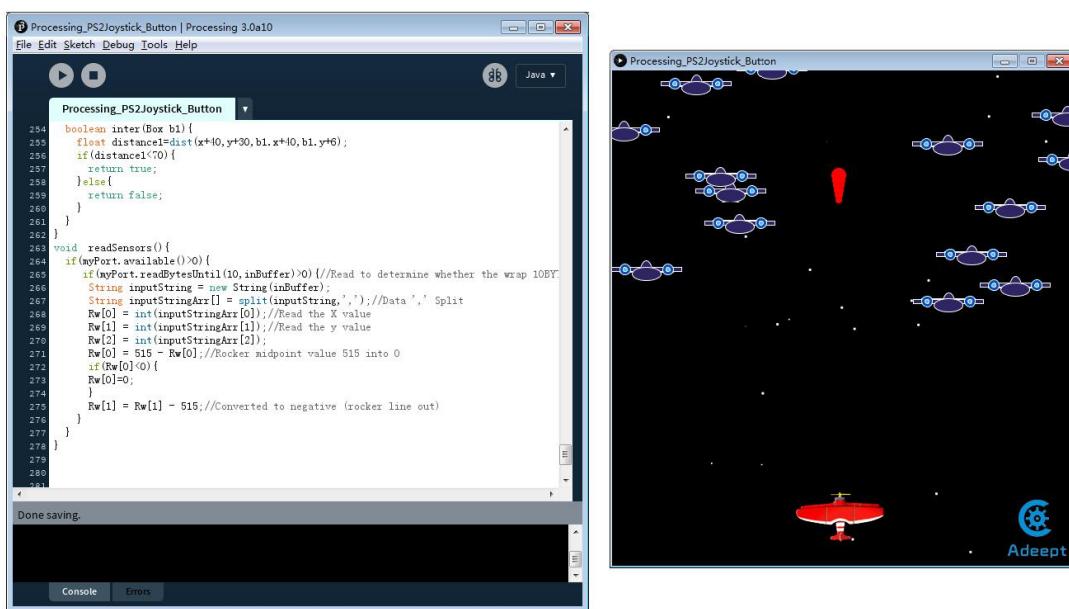
Step 1: Build the circuit

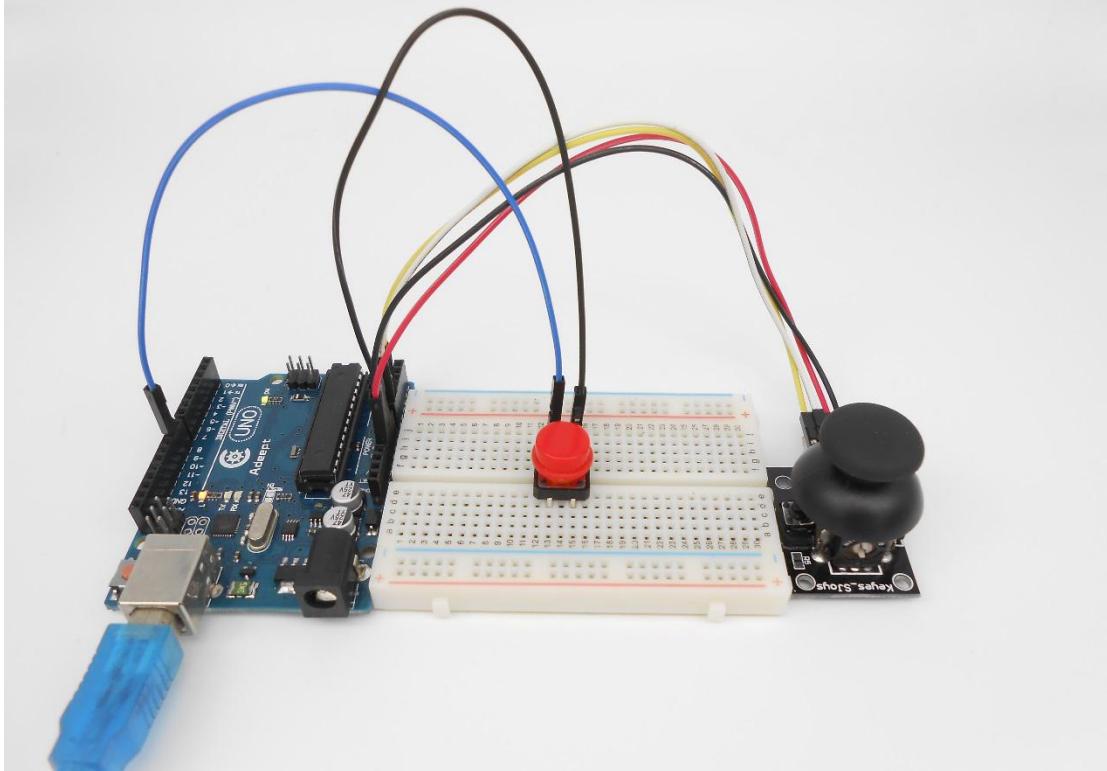


Step 2: Program

Step 3: Compile the program and upload to Arduino UNO board

Step 4: Run the Processing software (Processing PS2Joystick Button.pde)







Adeept

STEM Education Products and Service Provider

Shenzhen Adeept Technology Co., Ltd.

: www.adeept.com

: support@adeept.com

: Rm.1315, Shihong Building, No.2095 Bixin Rd.,
Longgang Dist., Shenzhen CHINA