4.6 Button control LED

1. Learning goal:

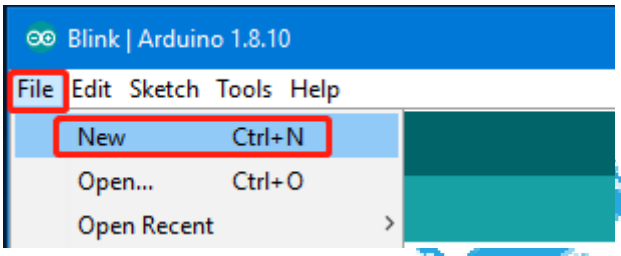
Buttons control LED lights.

2. Experimental phenomena:

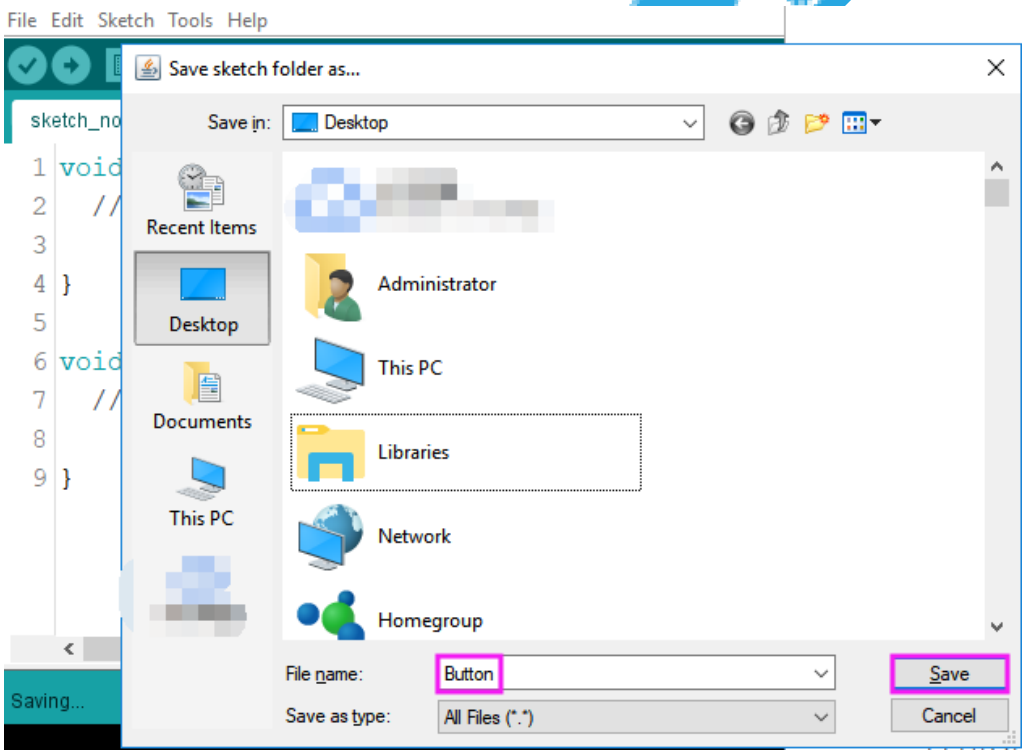
When button is pressed, the LED9 indicator status is reversed. When the power is turned on, the D9 indicator is off. Press the button to light up the LED9. Press the function button again to turn off the LED9.

3. Create new project

3.1 Click【File】-->【New】.



3.2 Press **Ctrl+S** to save and rename Button. As shown below.



3.3 We can see that there is a Serial folder with **Button.ino** on the computer desktop.

3.2 We will **Button.ino** as shown below.

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

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

The setup() function only runs once when the car is turned on or when the reset button is pressed, and the program for initializing the relevant content can be written;

The loop() function is the main loop function of the car and most of the data processing and logic processing are done in this function.

4. **Programming**

4.1 From the hardware manual, we csn know that the button K1 connect to Pin8 of UNO.

```
#define KEY_PIN 8    //Define key pin
#define LED_PIN 5    //Define LED9 pin
```

4.2 Define global variables to save button status.

```
//Button status
bool button_press = false;
```

4.3 Create a new keyscan function to detect whether the button is pressed.
Because the jitter occurs when the contacts of the mechanical buttons are closed and opened, if the MCU detects the on/off state of the button during the contact jitter, it may cause a judgment error.
That is to say, pressing or releasing the button once is mistakenly considered as multiple operations, causing mishandling.
Therefore, in order to ensure that the MCU responds only once to a keystroke, we must eliminate the effects of button jitter.
After eliminating jitter, judge the state of the button again, confirm that the button is pressed and enter the while loop until the button is released and the “button_press = !button_press” is exited.

```
void keyscan()
{
  int val;
  val = digitalRead(KEY_PIN); //Read the digital 8-port level value assi
  if (val == LOW)             //When the button is pressed
  {
    delay(10);                  //Delayed debounce
    val = digitalRead(KEY_PIN); //Read button status again
    while (val == LOW)
    {
      val = digitalRead(KEY_PIN); //Third read button status
      if (val == HIGH)            //Determine if the button is released,
      {
        button_press = !button_press;
        return;
      }
    }
  }
}
```

4.4 Initialize the pin mode in the setup() function, the button is in pull-up input mode, and the LED is in output mode.
Because from the hardware manual, we can see that the function button is pressed low, the release is high, that is, the low level is valid, so the pin voltage is pulled high by default to

work properly.

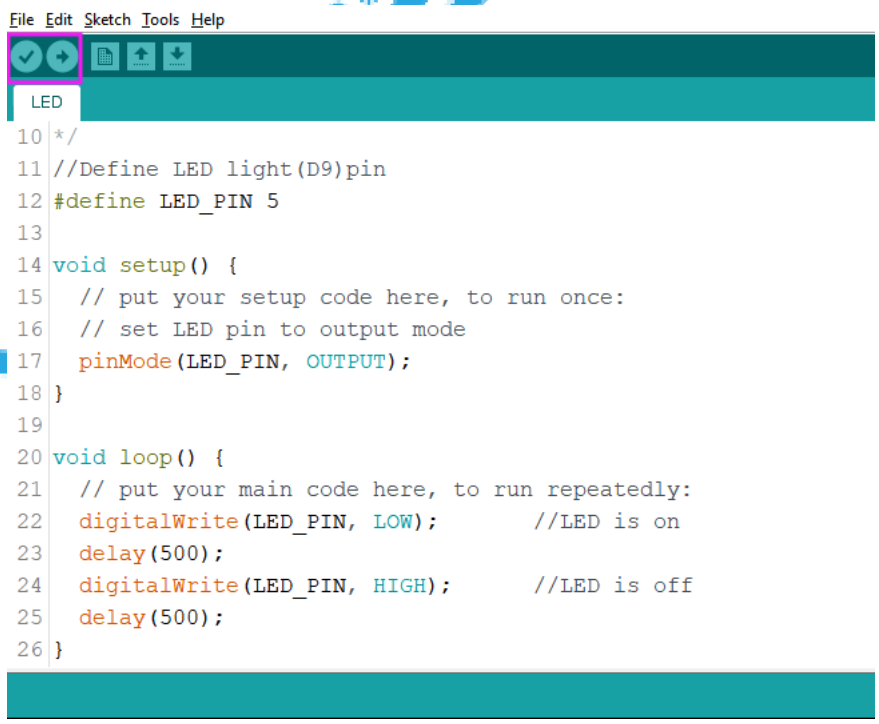
```
void setup() {
  // put your setup code here, to run once:
  pinMode(KEY_PIN, INPUT_PULLUP);//Set the button pin to pull-up input mode
  pinMode(LED_PIN, OUTPUT);      //Set the LED pin to output mode
}
```

4.5 In the loop () main loop function to detect whether the button is pressed, if button is pressed, the LED D9 is lit. When pressed again, the LED D9 is off.

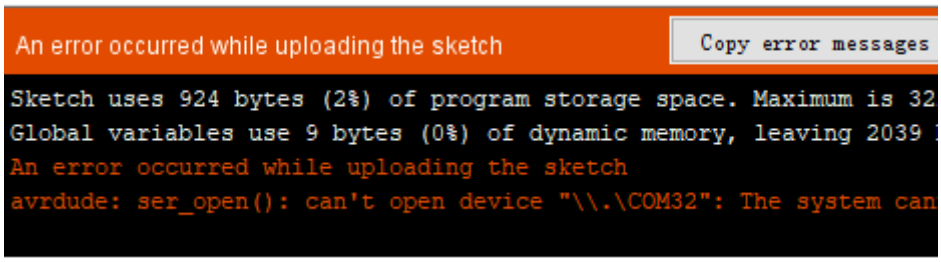
```
void loop() {
  // put your main code here, to run repeatedly:
  keyscan();
  if (button_press)
  {
    digitalWrite(LED_PIN, LOW);       //Turn on the LED light
  }
  else
  {
    digitalWrite(LED_PIN, HIGH);      //Turn off the LED light
  }
}
```

5. Compiling and downloading code

5.1 After the code is written, press Ctrl+S to save, then click the "√" button to compile. If there is no problem, click "→" to upload (the car must be connected to the computer via the USB cable).



5.2 If the compilation passes normally, but the following error occurs during uploading, the reason may be that the wrong serial port or the serial port is occupied.



Solution: Open the device manager to see if there is a serial port with CH340 tag. If not, please restart the Omniduino car, then, re-plug the USB cable or replace a USB cable;
If there is a serial port number, we need to close the other serial port or assistant software, avoid serial port occupation, and then re-select the serial port to ArduinoIDE【Tool】-->【Port】.