FAB LAB Enschede

Arduino workshop

Programming microcontrollers:

Arduino Uno with C

And Seeed Studio Grove Starter kit

CC BY NC SA

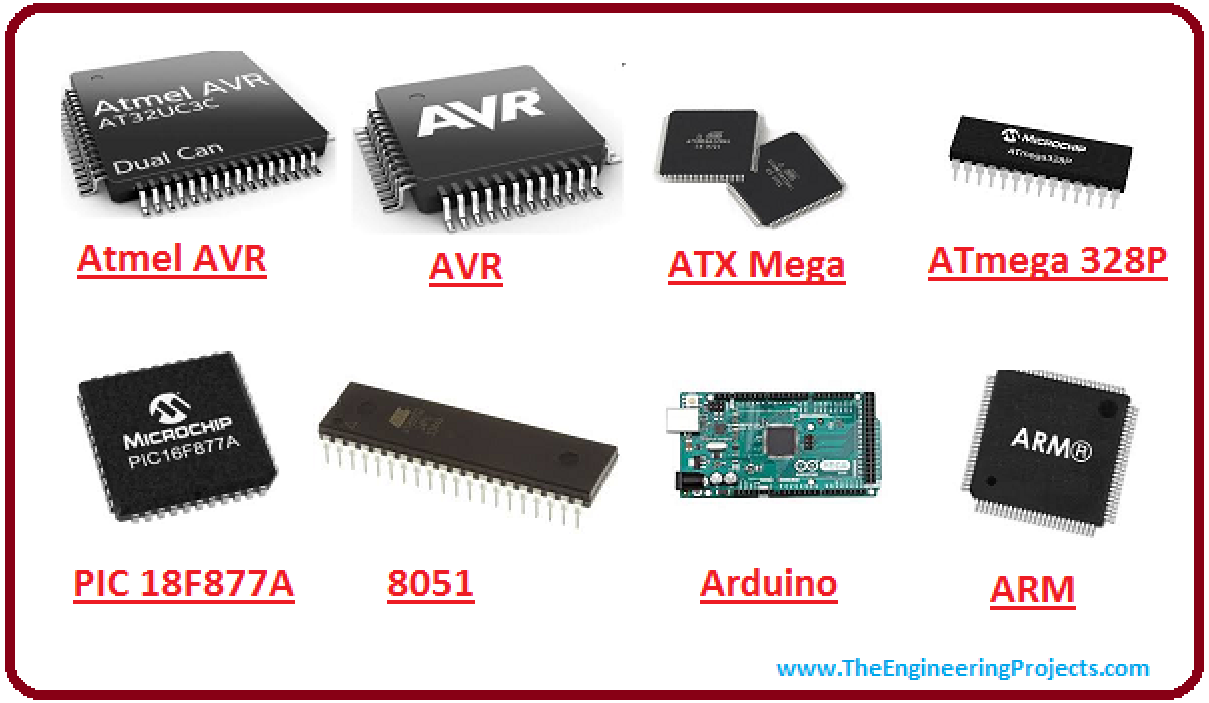
Leo Kuipers – Saxion Fablab Enschede - L.A.Kuipers@saxion.nl

Agenda

- What is Arduino?
- Designing with Grove
- Programming software
- Blink it!
- Programming in C

What is Arduino

Microcontrollers



A bit of history

- 1971: invention of the first microcontroller, the Intel 4004.
- 1975: Introduction of the first PIC microcontroller by General Instrument.
- 1980: Introduction of 8051 microcontroller by Intel. Its instruction set is still used nowadays.
- 1985: Introduction of the ARM1 by Acorn. Its abbreviation stands for Acorn RISC Machine, where RISC stands for Reduced Instruction Set Computer.
- 1993: introduction of EEPROM and Flash memory, allowing rapid prototyping and in-system programming.
- 1997: introduction of AVR microcontrollers by Atmel (later acquired by Microchip). One of the first to use on-chip flash memory for program storage.
- 2005: introduction of the Arduino platform, using ATmega8 AVR microcontrollers.

Microcontrollers require instructions

- All microcontrollers require a set of instructions: a program
  - Arithmetic (add, subtract, increment, decrement, multiply)
  - Logical (AND, OR, XOR)
  - Boolean (Clear a bit, Set a bit, Move a bit, Jump if specified bit is set)
  - Etc.

- All microcontrollers have a program counter
  - It remembers which instruction is being executed at the current time
  - After execution of this instruction, the program counter increases by 1
  - After a restart or reset, the program counter reverts to 0

Programming languages

- Binary code
- Assembly (1947)
- C (1972)

Programming languages

- Binary code

  `10110 000 01100001` (which is B0 61 in hex representation)

- Assembly (1947) → Low level programming: Machine code instructions (opcodes)

  `MOV AL, 61h` (Move command = opcode B0, register AL is identified by 000)

- C (1972) → Procedural computer programming language. Requires a compiler to translate the code into machine code.

```
int main(void) { printf("hello, world\n"); }
```

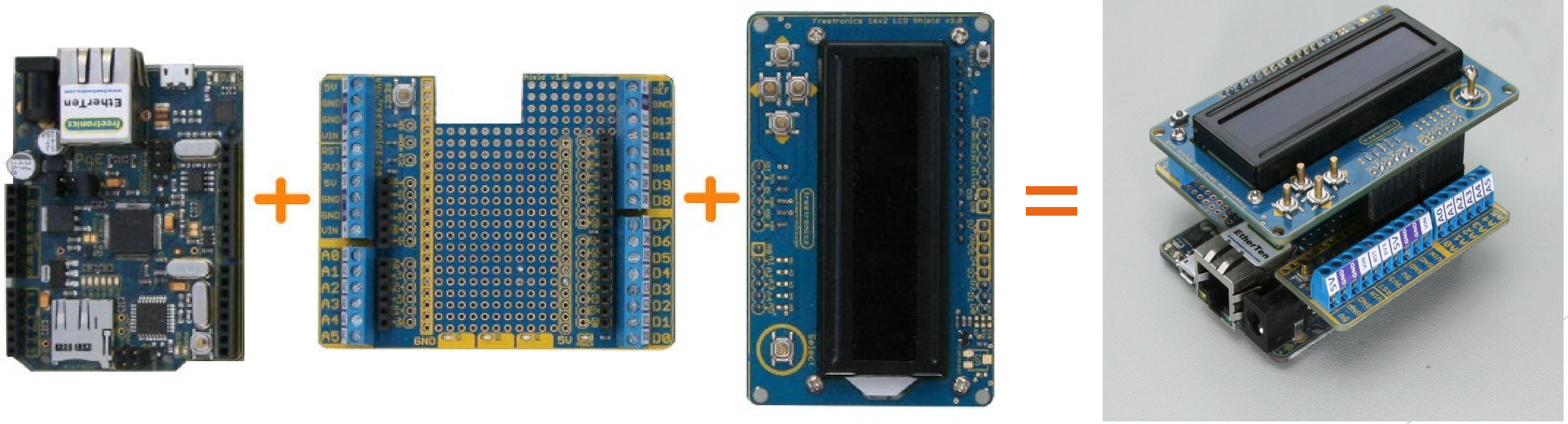
Arduino

- Open-source hardware
  - Easy to use
  - Various models
    - We’re going to use the Arduino Uno
  - Huge community
    - Arduino.cc
    - Stackexchange.com
- Open-source development software
  - Programming in C (or better: C++)
  - Convert to microcontroller instructions (“compile”)
  - Upload to Arduino

Arduino

- Shields
  - Hardware that can be plugged on top of the Arduino → Piggyback
  - Expand the functionality

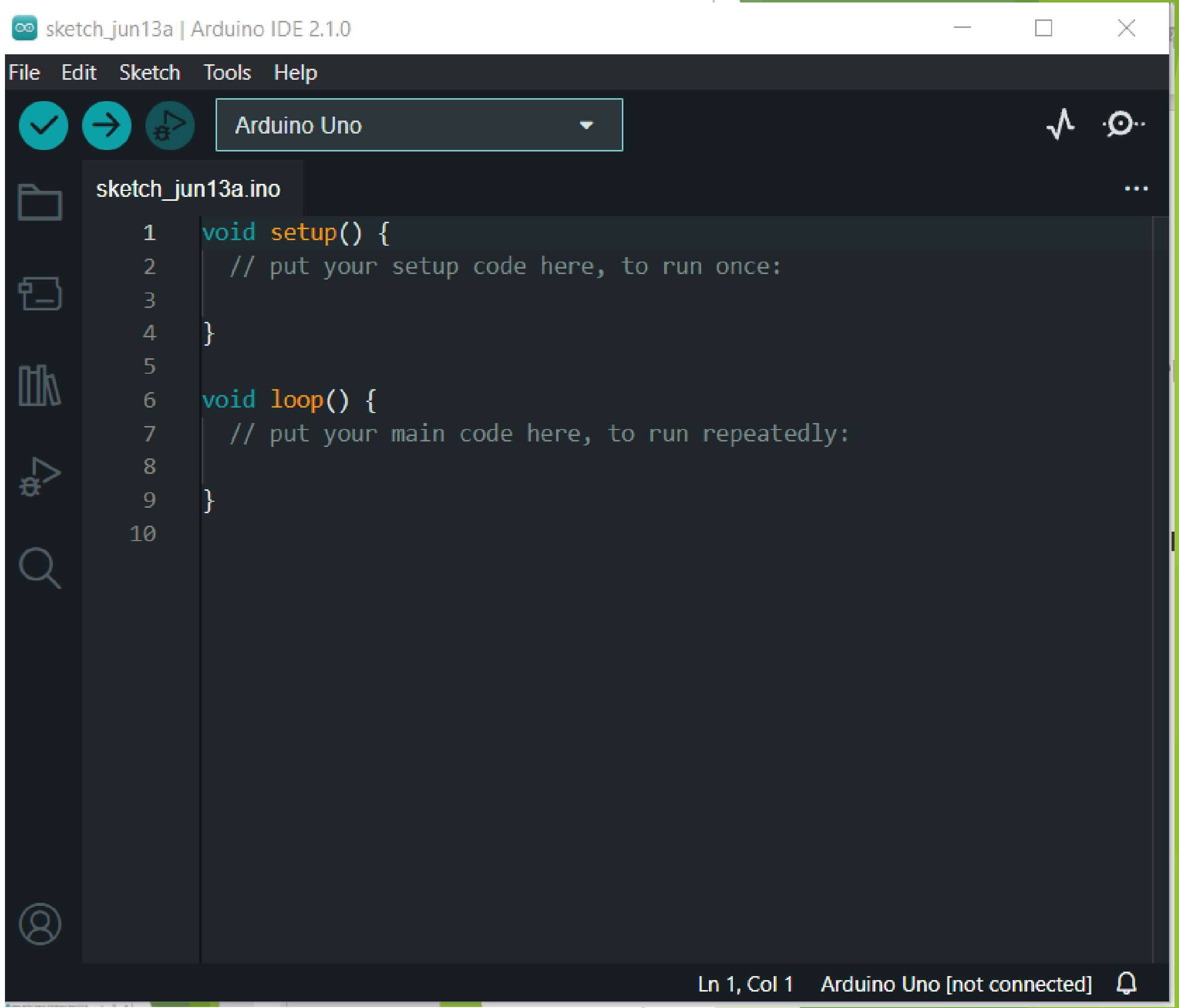
RANDOMNERDTUTORIALS.COM



Programming software

Arduino IDE

- IDE: Integrated Development Environment
  - Programming in C
  - Convert to microcontroller instructions ("compile")
  - Upload to Arduino



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

}

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

}
```

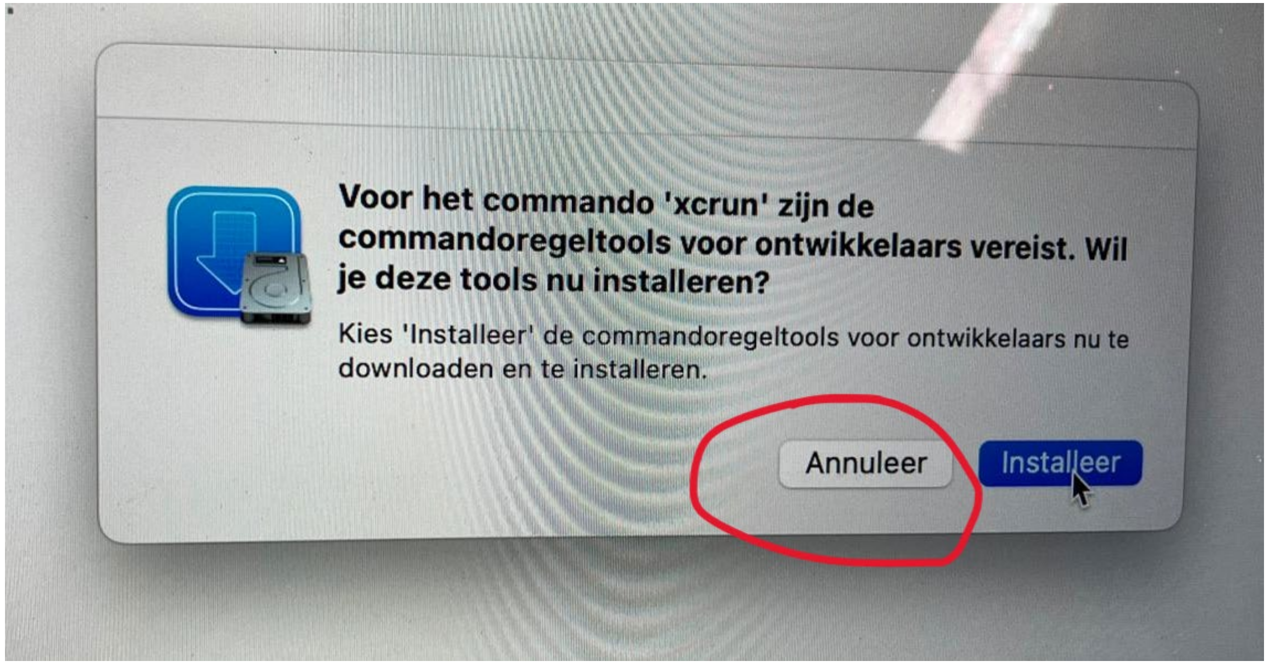
Arduino IDE

- Download the latest version at:

https://www.arduino.cc/en/Main/Software

Only for mac

Ignore this popup:

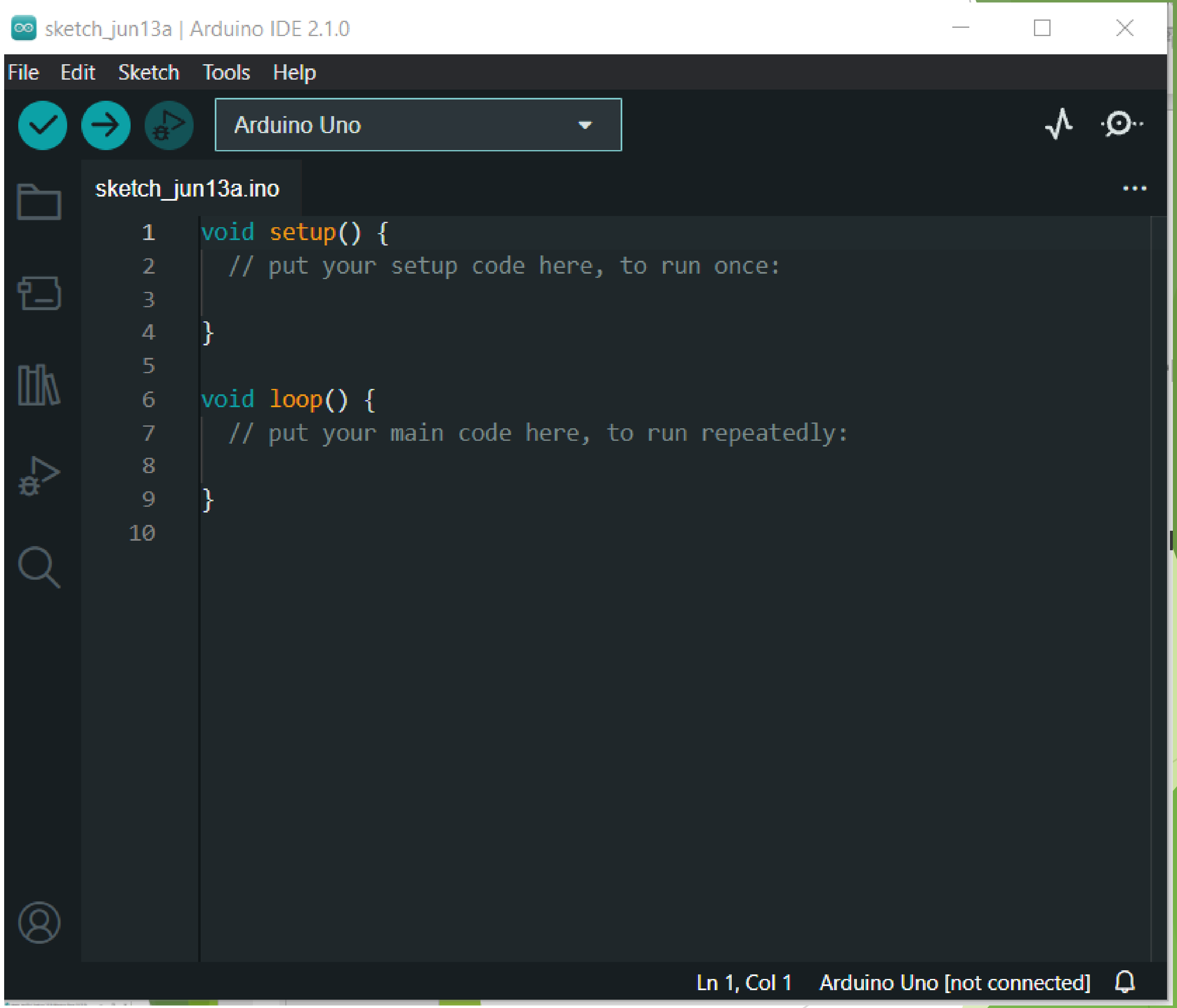


Voor het commando 'xcrun' zijn de commandoregeltools voor ontwikkelaars vereist. Wil je deze tools nu installeren?

Kies 'Installeer' de commandoregeltools voor ontwikkelaars nu te downloaden en te installeren.

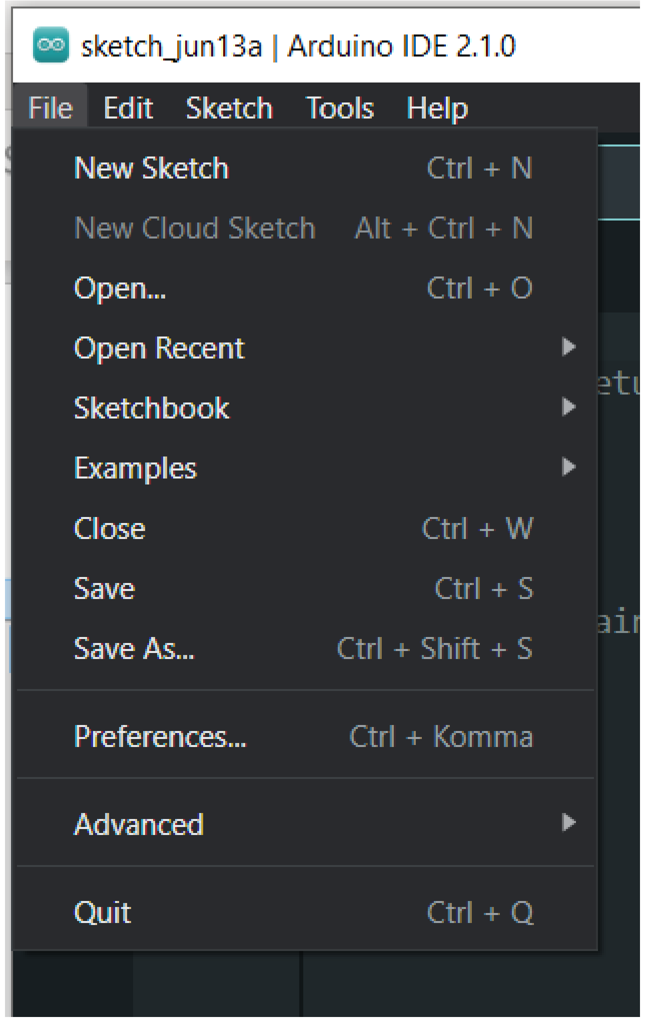
Arduino IDE

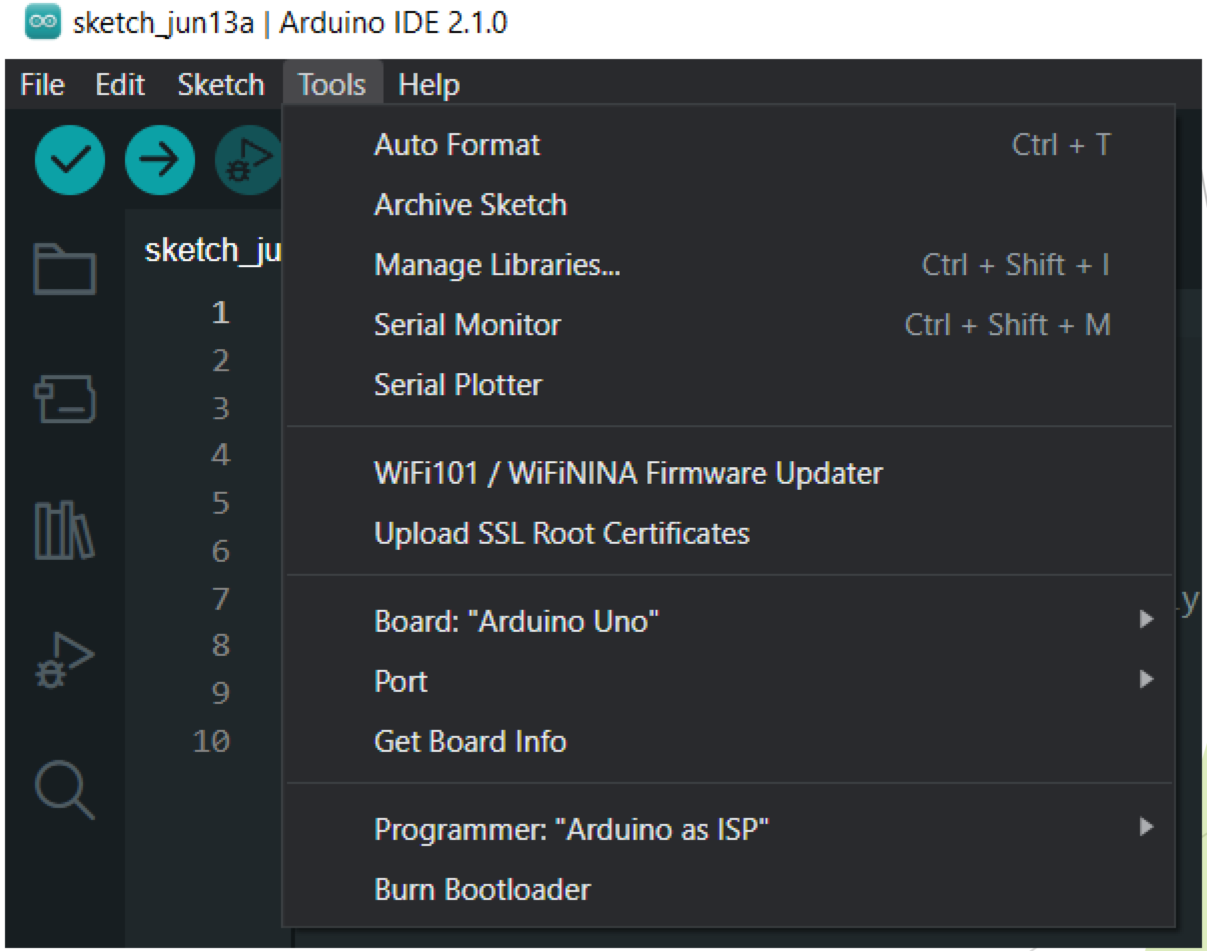
- Verify ✓
- Upload →



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

}

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

}
```

Arduino IDE



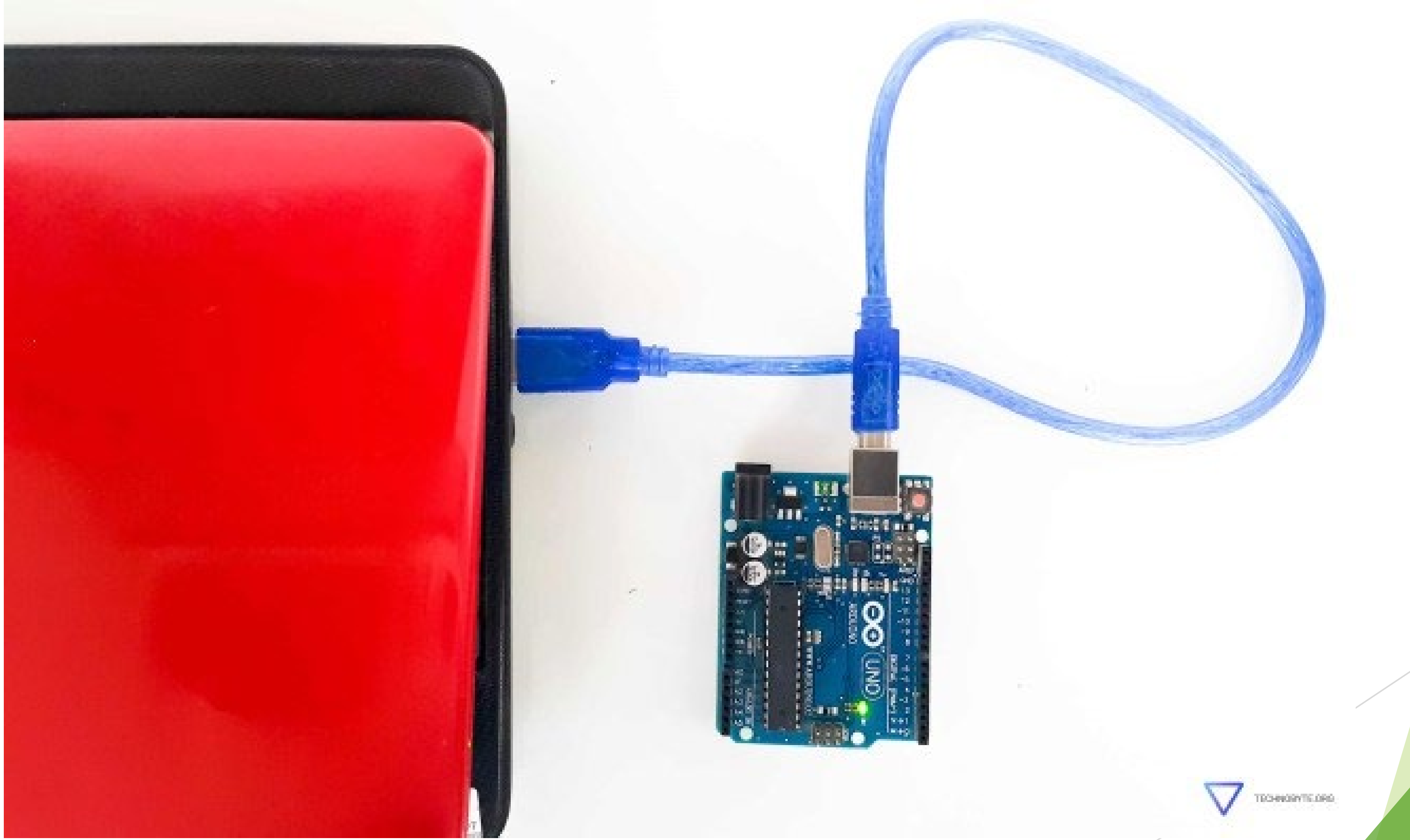

sketch_jun13a | Arduino IDE 2.1.0
File
Edit
Sketch
Tools
Help
sketch_ju
Auto Format Ctrl + T
Archive Sketch
Manage Libraries... Ctrl + Shift + I
Serial Monitor Ctrl + Shift + M
Serial Plotter
WiFi101 / WiFiNINA Firmware Updater
Upload SSL Root Certificates
Port
Get Board Info
Burn Bootloader

Board: "Arduino Uno"

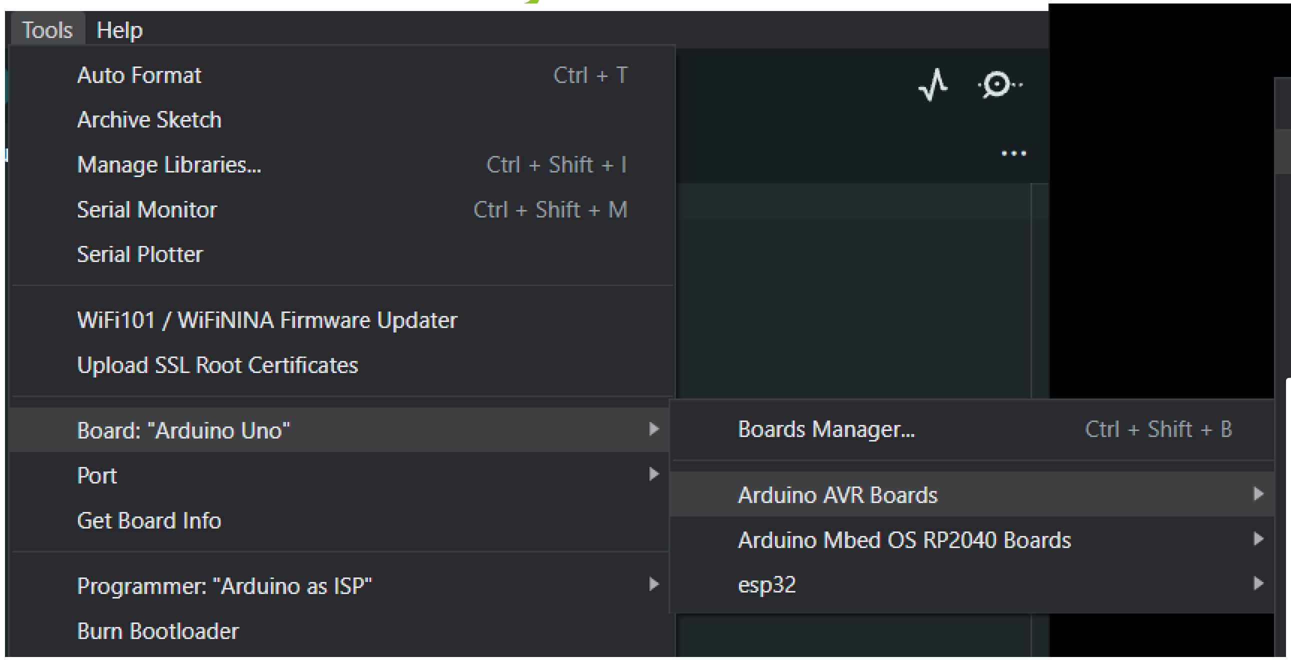
Programmer: "Arduino as ISP"

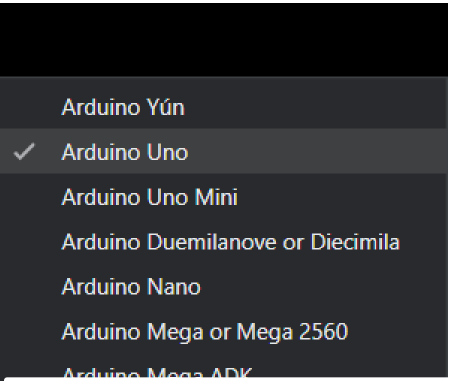
Blink it!

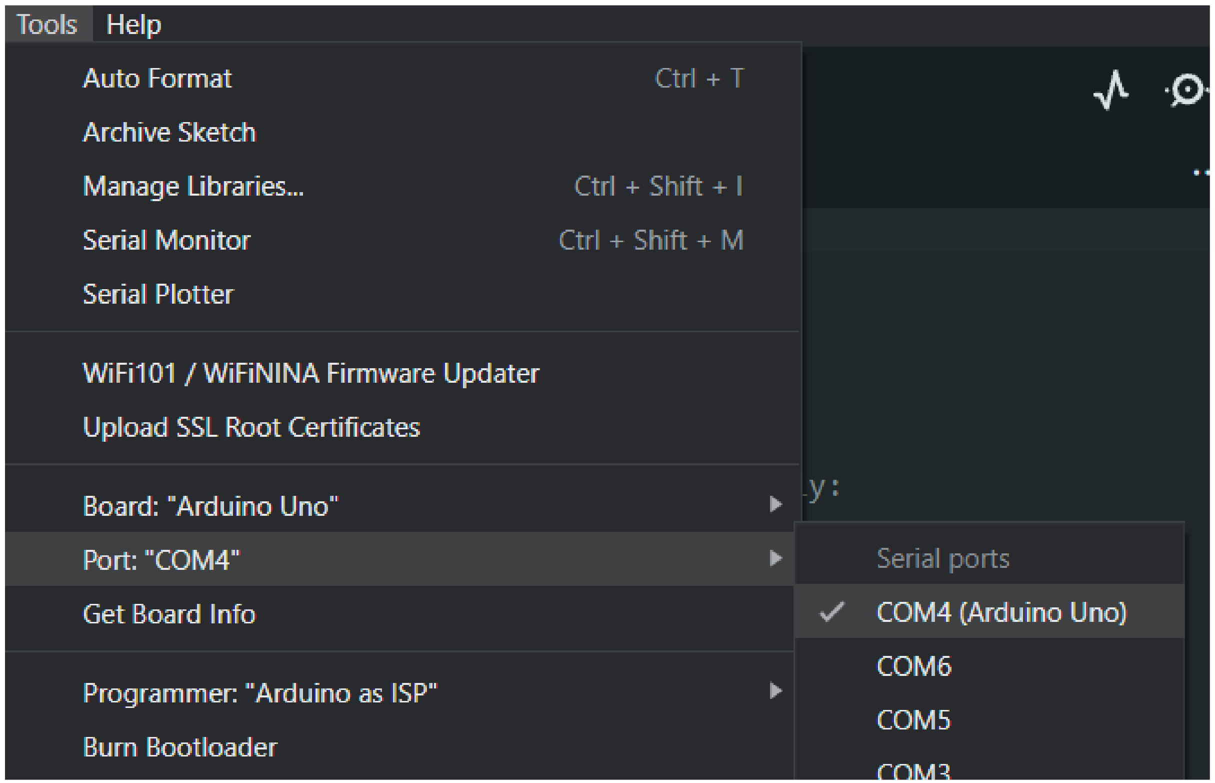
Connect your Arduino



Connect your Arduino

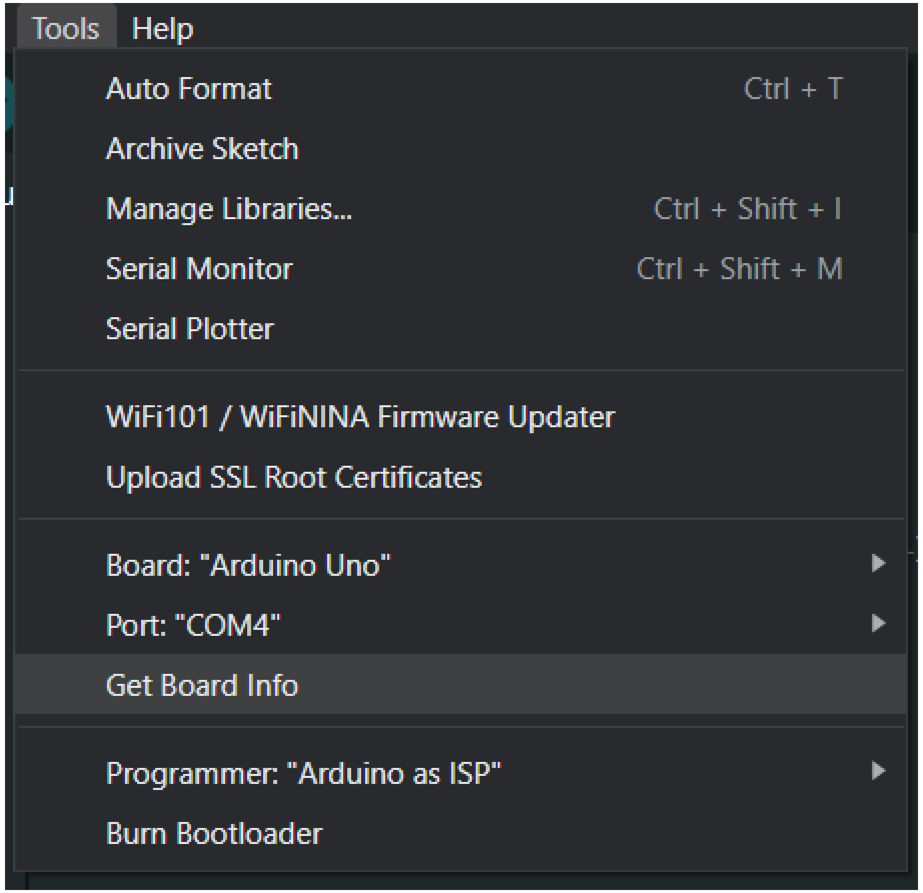

Tools
Help
Auto Format Ctrl + T
Archive Sketch
Manage Libraries... Ctrl + Shift + I
Serial Monitor Ctrl + Shift + M
Serial Plotter
WiFi101 / WiFiNINA Firmware Updater
Upload SSL Root Certificates
Board: "Arduino Uno"
Port
Get Board Info
Programmer: "Arduino as ISP"
Burn Bootloader
Boards Manager... Ctrl + Shift + B
Arduino AVR Boards
Arduino Mbed OS RP2040 Boards
esp32



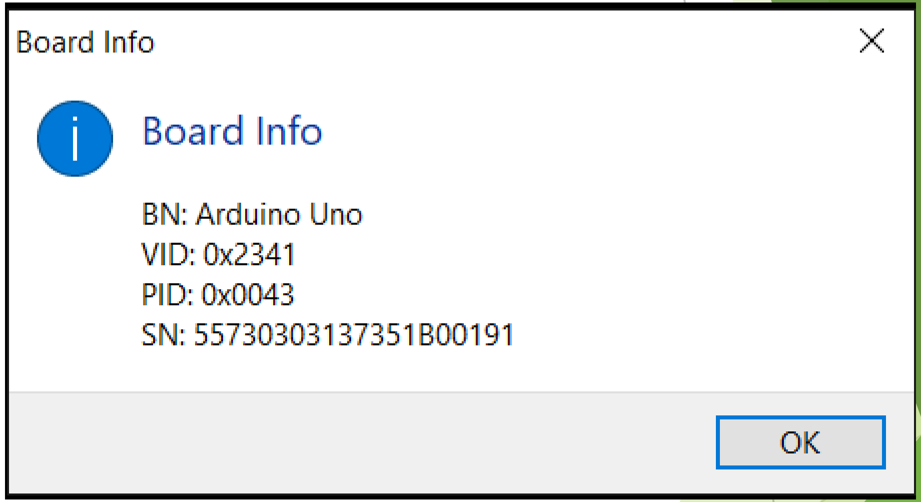


- After connecting your Arduino for the first time: additional software libraries will be automatically downloaded & installed.
- Due to a firewall issue this will fail when connected to Eduroam WIFI! Please use a personal hotspot.

Connect your Arduino


Tools
Help
Auto Format Ctrl + T
Archive Sketch
Manage Libraries... Ctrl + Shift + I
Serial Monitor Ctrl + Shift + M
Serial Plotter
WiFi101 / WiFiNINA Firmware Updater
Upload SSL Root Certificates
Get Board Info
Burn Bootloader

Board: "Arduino Uno"
Port: "COM4"
Programmer: "Arduino as ISP"

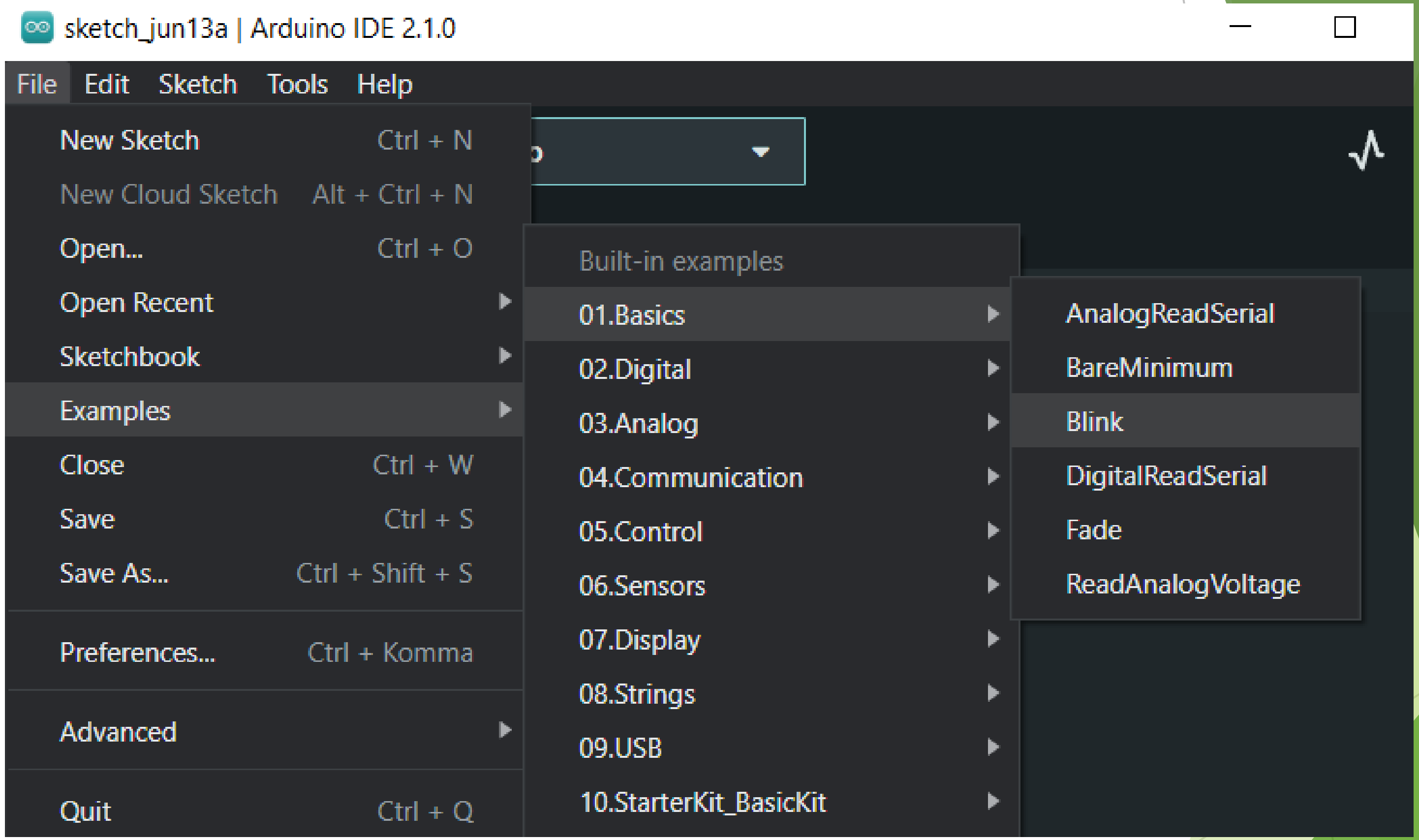


Board Info

BN: Arduino Uno
VID: 0x2341
PID: 0x0043
SN: 55730303137351B00191

Program your Arduino

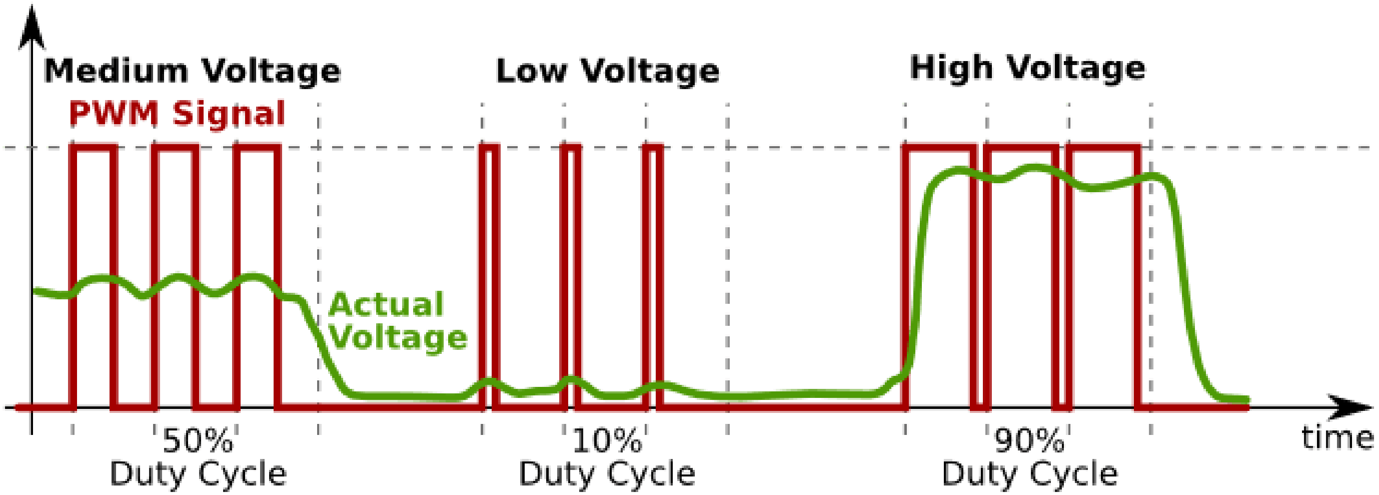
- Open the Blink example



- Upload the code to your Arduino

Blink!

- Can you make it blink faster?
- How fast?
- What happens if you vary the on-time and the off-time?

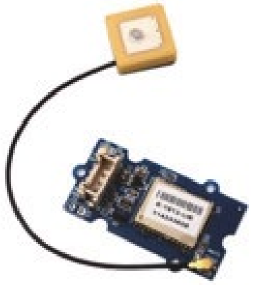


Designing with Grove

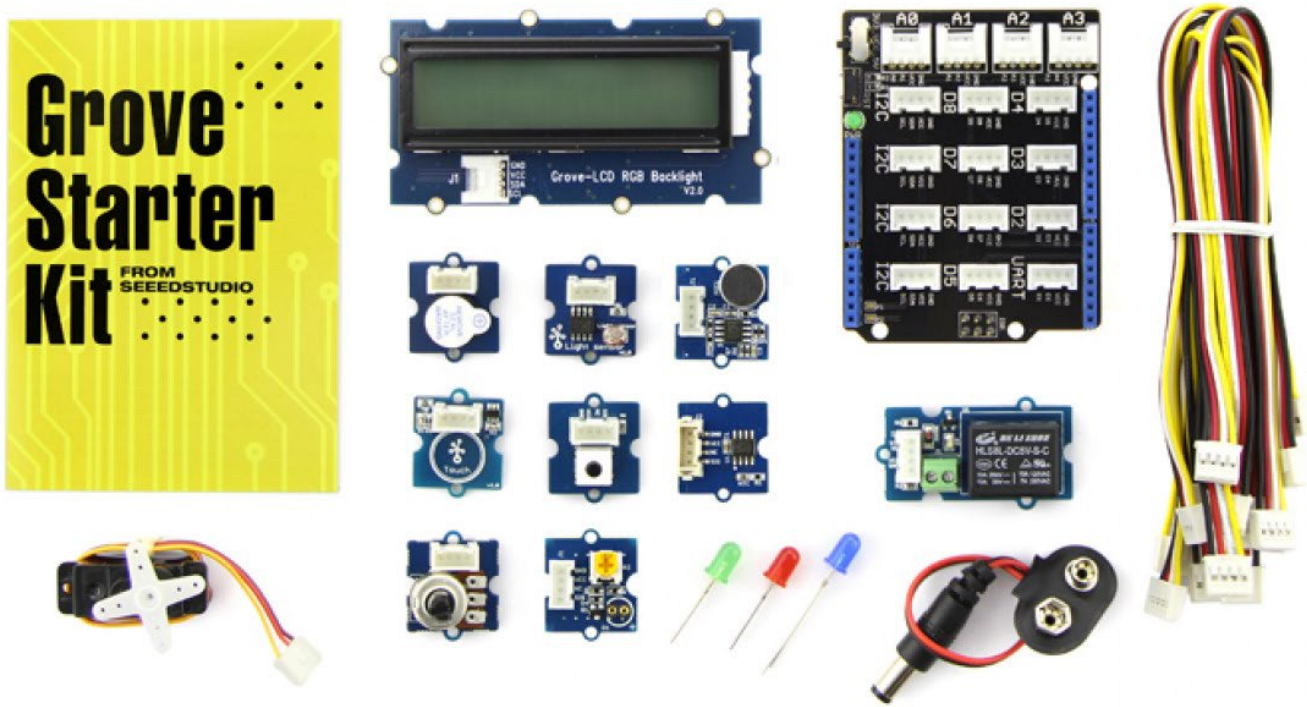
By Seeedstudio

Seeed studio Grove

- Modular prototyping system
- One base unit (Arduino Shield with various connectors)
- Various modules, each with a single function, e.g.
  - Button
  - Buzzer
  - Servo
  - Temperature sensor
  - Light Sensor
  - Rotary Angle sensor
  - At least 70 more…

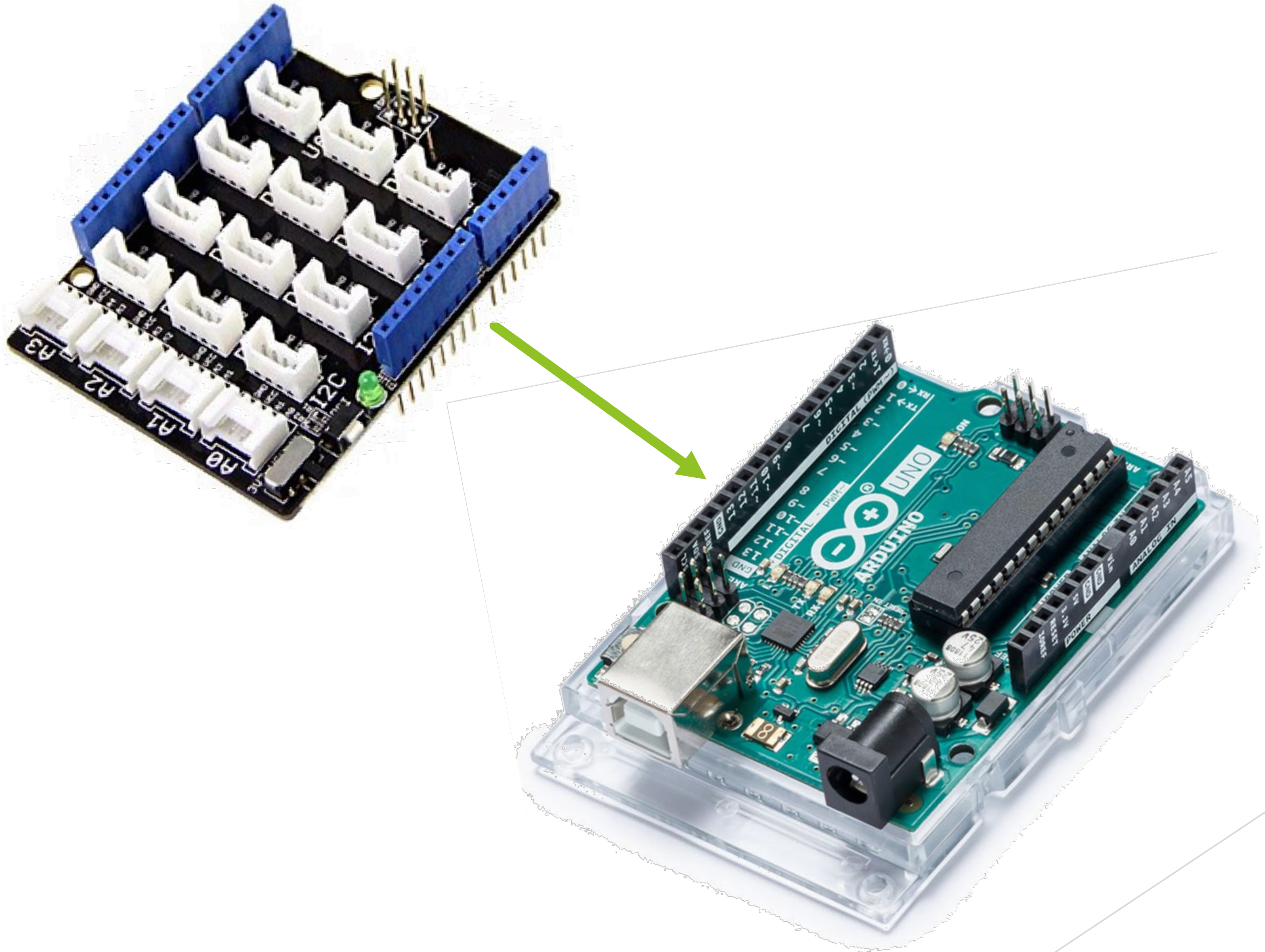


Grove - Starter kit



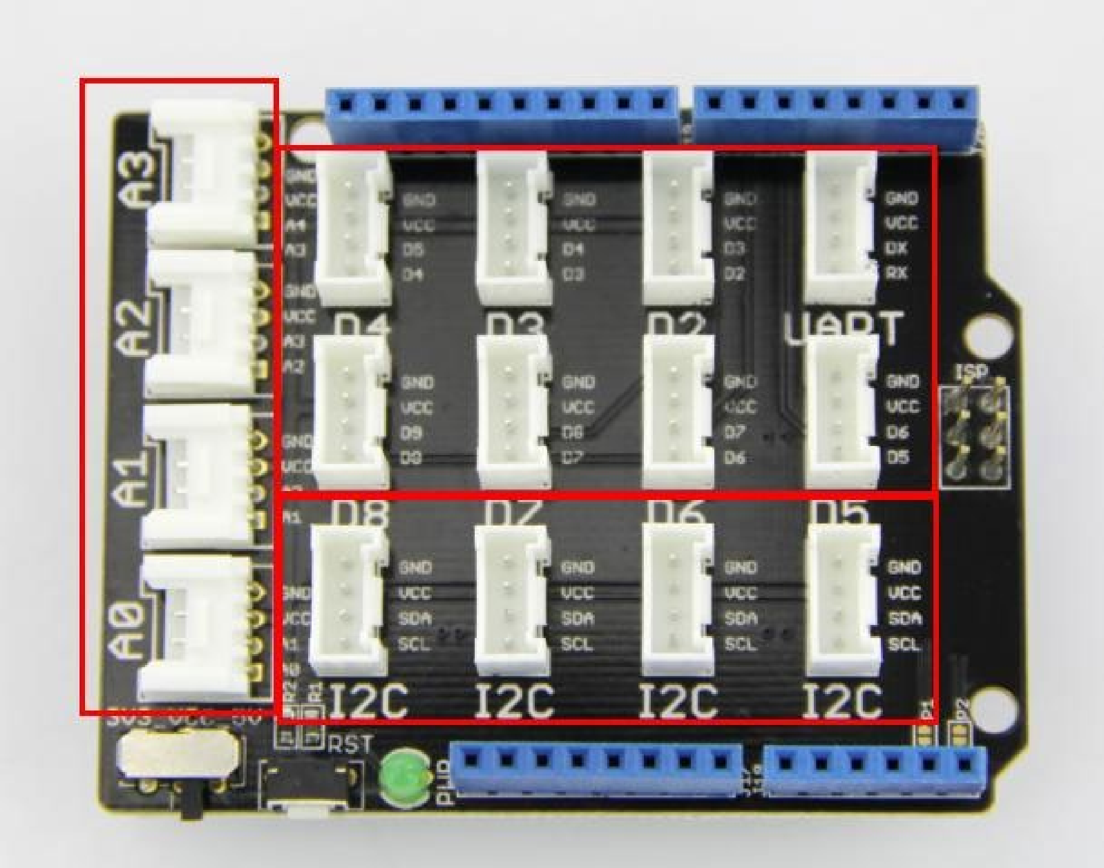
Blink with Grove shield

Connect your Arduino



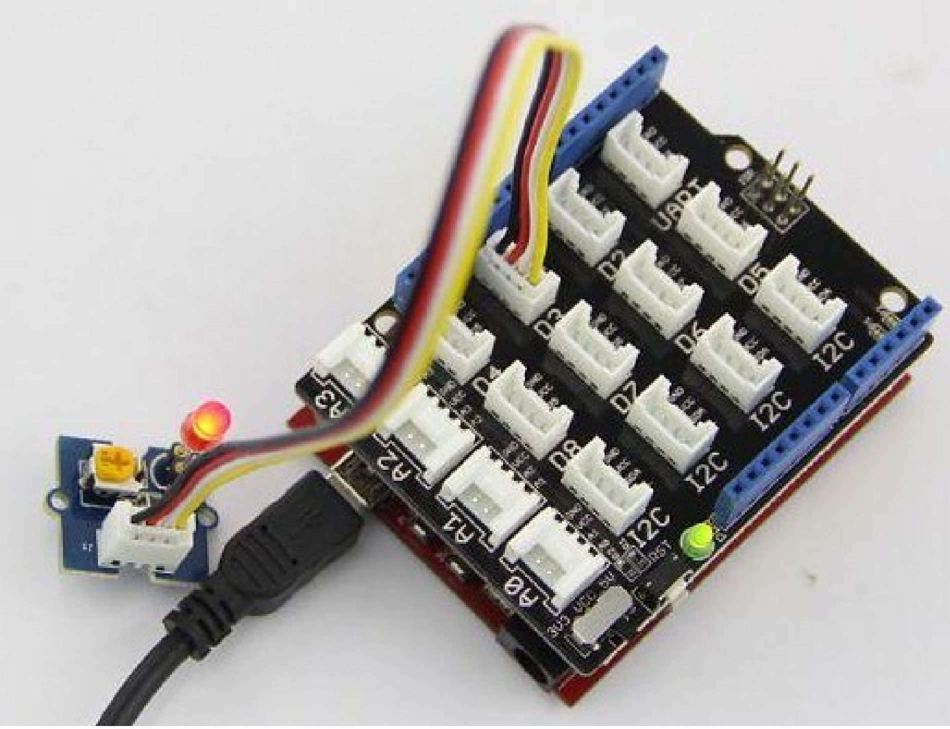
Grove Base shield

- 1. Digital ports (D1 to D8)
    - 2 digital pins per connector
- 2. Analog input ports (A0 to A3)
    - 2 analog pins per connector
- 3. I2C communication bus (4x)



Program your Arduino – Grove LED

- Disconnect your Arduino (always disconnect before you make hardware changes!)
- Connect the LED module to the Grove base shield (connector D3). Press firmly!



Program your Arduino – Grove LED

- Add the following code:
  - `const int pinLed = 3;`

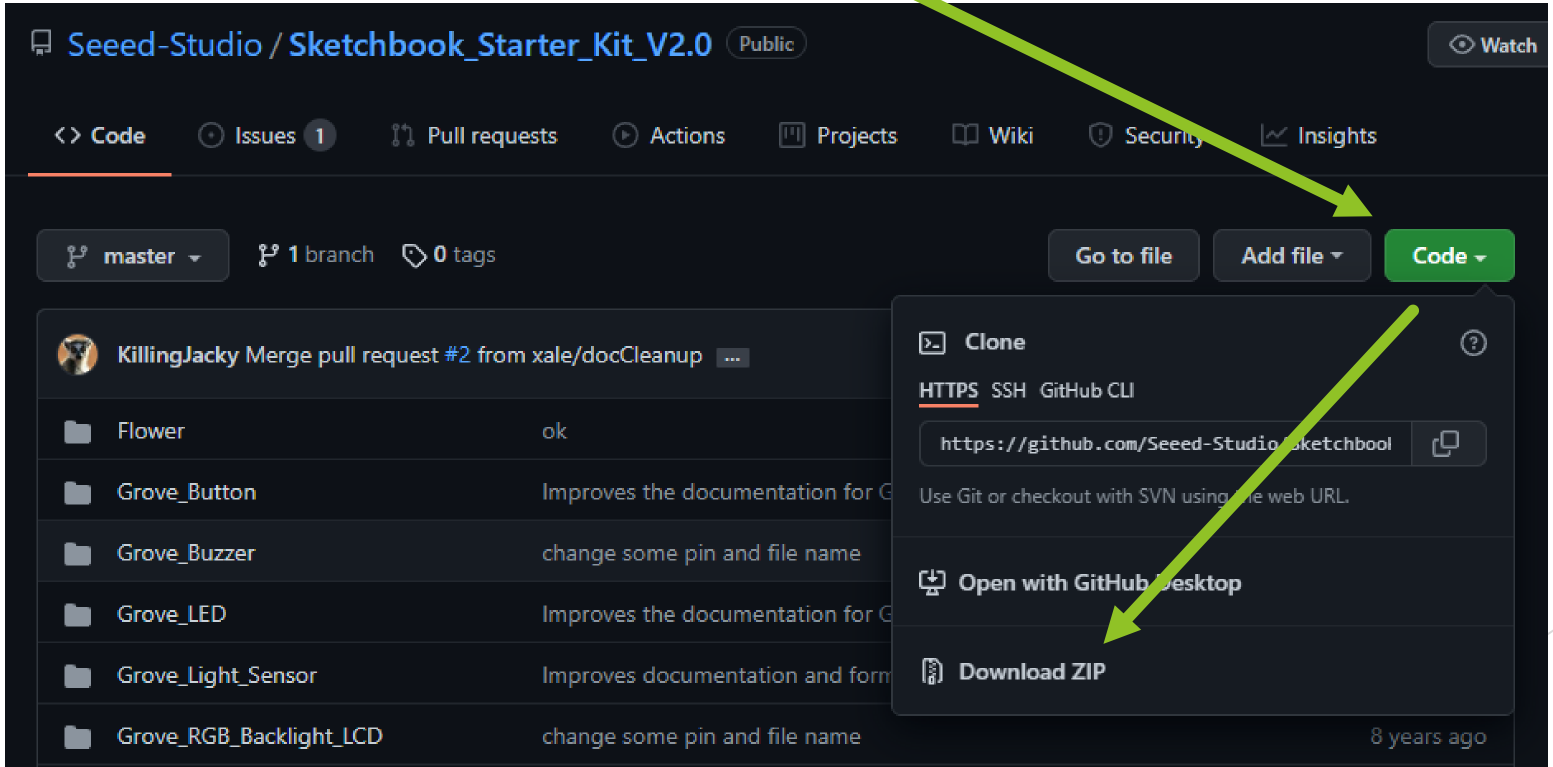
Watch out:

C is Case Sensitive!
&
All lines end with a ;

- Replace all occurrences of `LED_BUILTIN` with `pinLed`
- Connect your Arduino
- Upload the code to your Arduino ➡

Grove sketchbook with examples

- The Grove Starter kit Sketchbook:
- https://github.com/Seeed-Studio/Sketchbook_Starter_Kit_V2.0



https://github.com/Seeed-Studio/Sketchbool

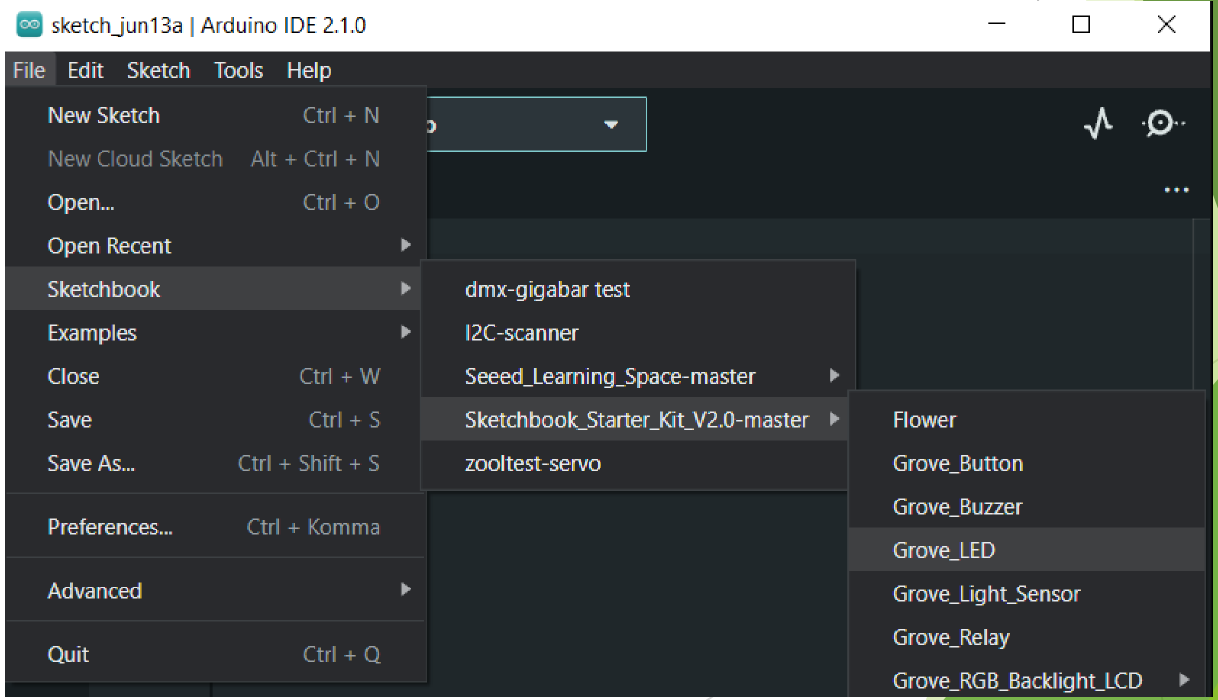
Use Git or checkout with SVN using the web URL.

KillingJacky Merge pull request #2 from xale/docCleanup ...

Flower	ok	
Grove_Button	Improves the documentation for G	
Grove_Buzzer	change some pin and file name	
Grove_LED	Improves the documentation for G	
Grove_Light_Sensor	Improves documentation and form	
Grove_RGB_Backlight_LCD	change some pin and file name	8 years ago

Grove sketchbook with examples

- Unzip and save in documents/Arduino folder
- Check Sketchbook location in Arduino IDE: File → Preferences
- Close & reopen Arduino IDE
- Grove examples are here

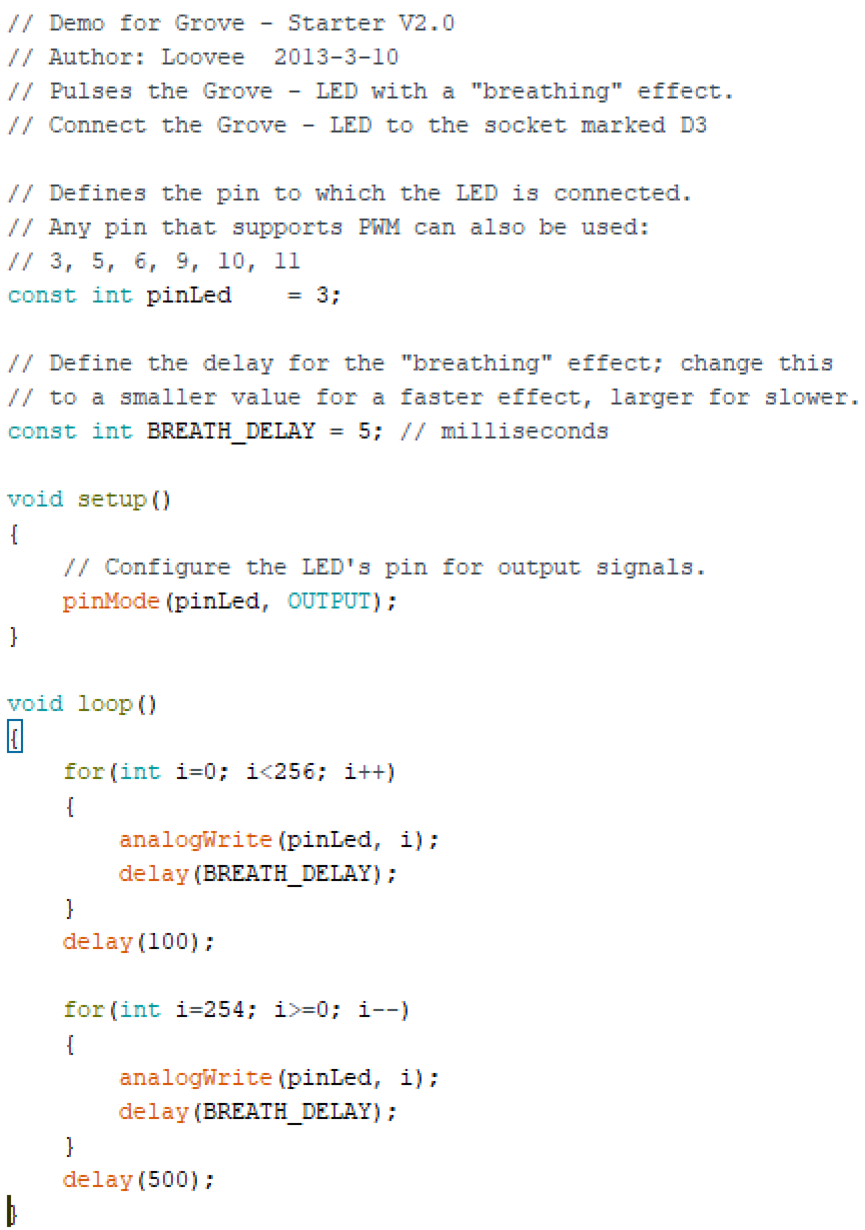


Open and upload the following example: `Grove_LED`

Programming in C

Programming in C

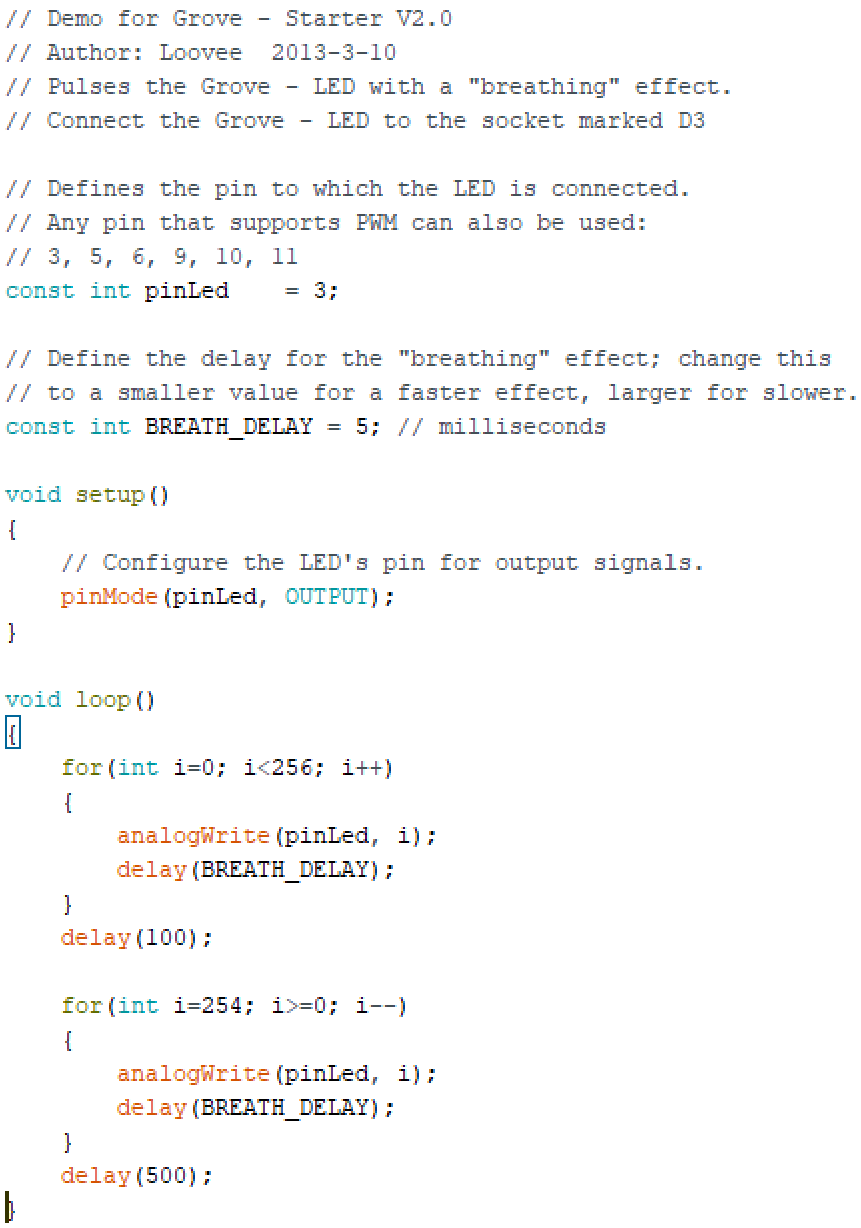
- Comment
- Definitions & Inclusions
- Standard function Setup()
- Standard function Loop()



```
// Demo for Grove - Starter V2.0
// Author: Loovee  2013-3-10
// Pulses the Grove - LED with a "breathing" effect.
// Connect the Grove - LED to the socket marked D3

// Defines the pin to which the LED is connected.
// Any pin that supports PWM can also be used:
// 3, 5, 6, 9, 10, 11
const int pinLed    = 3;

// Define the delay for the "breathing" effect; change this
// to a smaller value for a faster effect, larger for slower.
const int BREATH_DELAY = 5; // milliseconds

void setup()
{
    // Configure the LED's pin for output signals.
    pinMode(pinLed, OUTPUT);
}

void loop()
{
    for(int i=0; i<256; i++)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(100);

    for(int i=254; i>=0; i--)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(500);
}
```

Programming in C

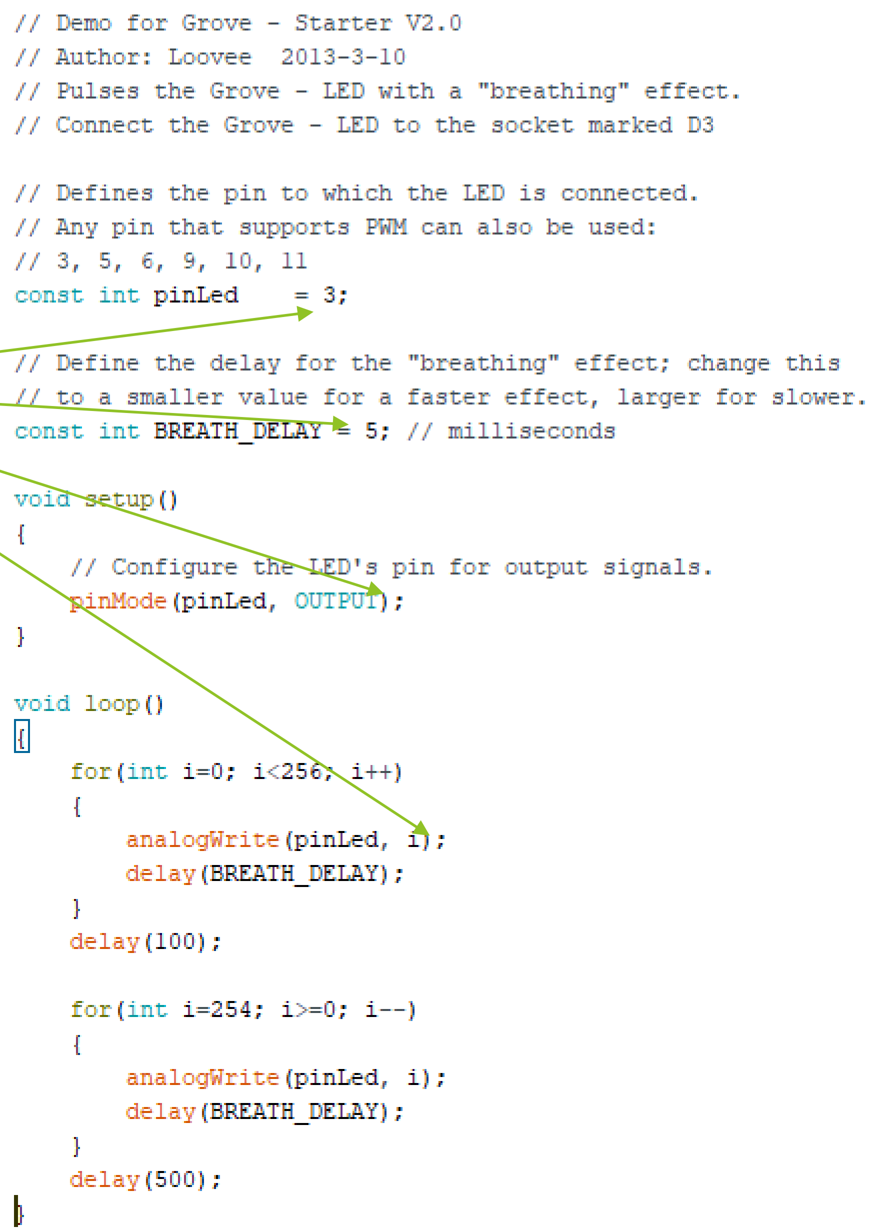
- Comment
  - start a line with //
  - This line will be ignored
  - Use it to take notes, to explain or while testing/debugging your code



```
// Demo for Grove - Starter V2.0
// Author: Loovee  2013-3-10
// Pulses the Grove - LED with a "breathing" effect.
// Connect the Grove - LED to the socket marked D3

// Defines the pin to which the LED is connected.
// Any pin that supports PWM can also be used:
// 3, 5, 6, 9, 10, 11
const int pinLed    = 3;

// Define the delay for the "breathing" effect; change this
// to a smaller value for a faster effect, larger for slower.
const int BREATH_DELAY = 5; // milliseconds

void setup()
{
    // Configure the LED's pin for output signals.
    pinMode(pinLed, OUTPUT);
}

void loop()
{
    for(int i=0; i<256; i++)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(100);

    for(int i=254; i>=0; i--)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(500);
}
```

Programming in C

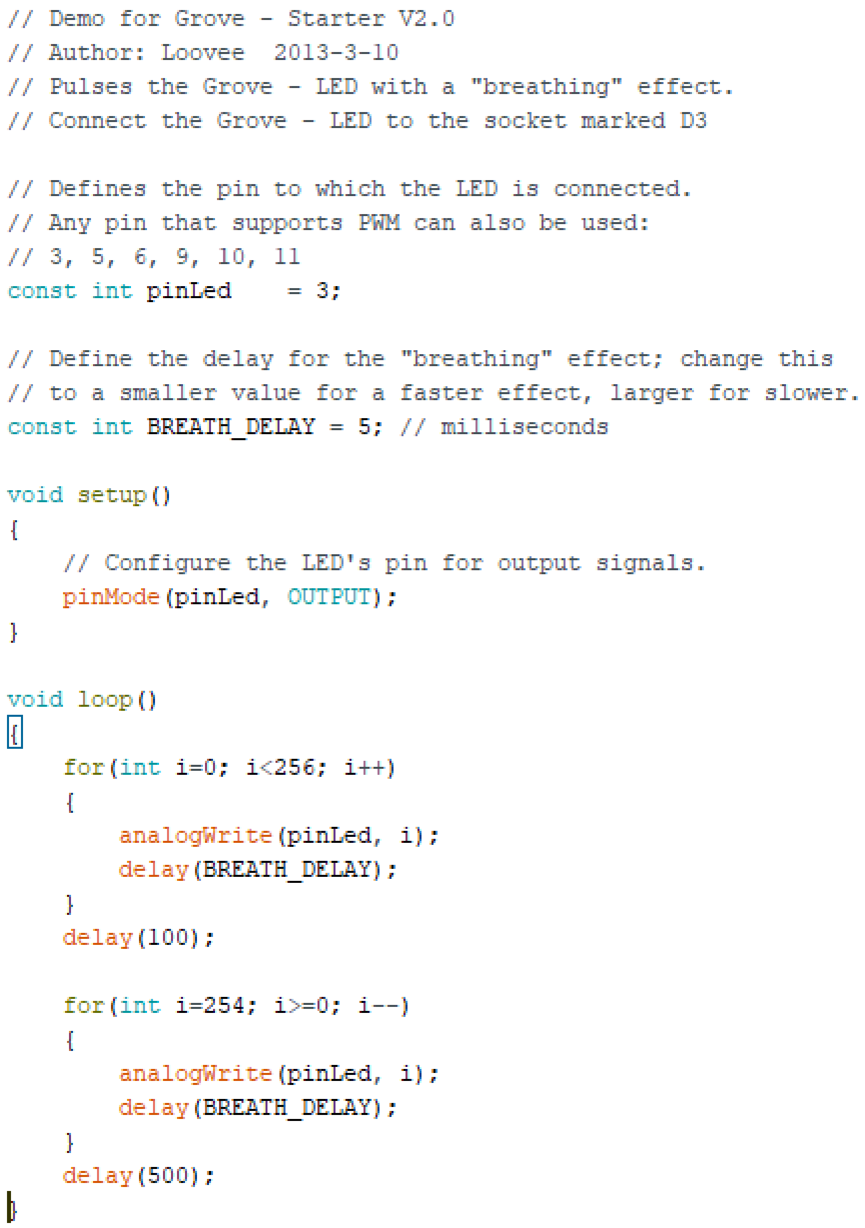
- Every statement ends with a ;



```
// Demo for Grove - Starter V2.0
// Author: Loovee  2013-3-10
// Pulses the Grove - LED with a "breathing" effect.
// Connect the Grove - LED to the socket marked D3

// Defines the pin to which the LED is connected.
// Any pin that supports PWM can also be used:
// 3, 5, 6, 9, 10, 11
const int pinLed    = 3;

// Define the delay for the "breathing" effect; change this
// to a smaller value for a faster effect, larger for slower.
const int BREATH_DELAY = 5; // milliseconds

void setup()
{
    // Configure the LED's pin for output signals.
    pinMode(pinLed, OUTPUT);
}

void loop()
{
    for(int i=0; i<256; i++)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(100);

    for(int i=254; i>=0; i--)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(500);
}
```

Programming in C

- Definitions & Inclusions
  - Declare what constants and variables you’re going to use
  - Constants: Declare once, don’t touch it later
  - Variable: Declare once and change its value throughout your code
    - `int my_variable = 10;`
    - Counting starts at 0!
  - You need to define the datatypes, so enough memory can be allocated
    - `int, long, float, char`



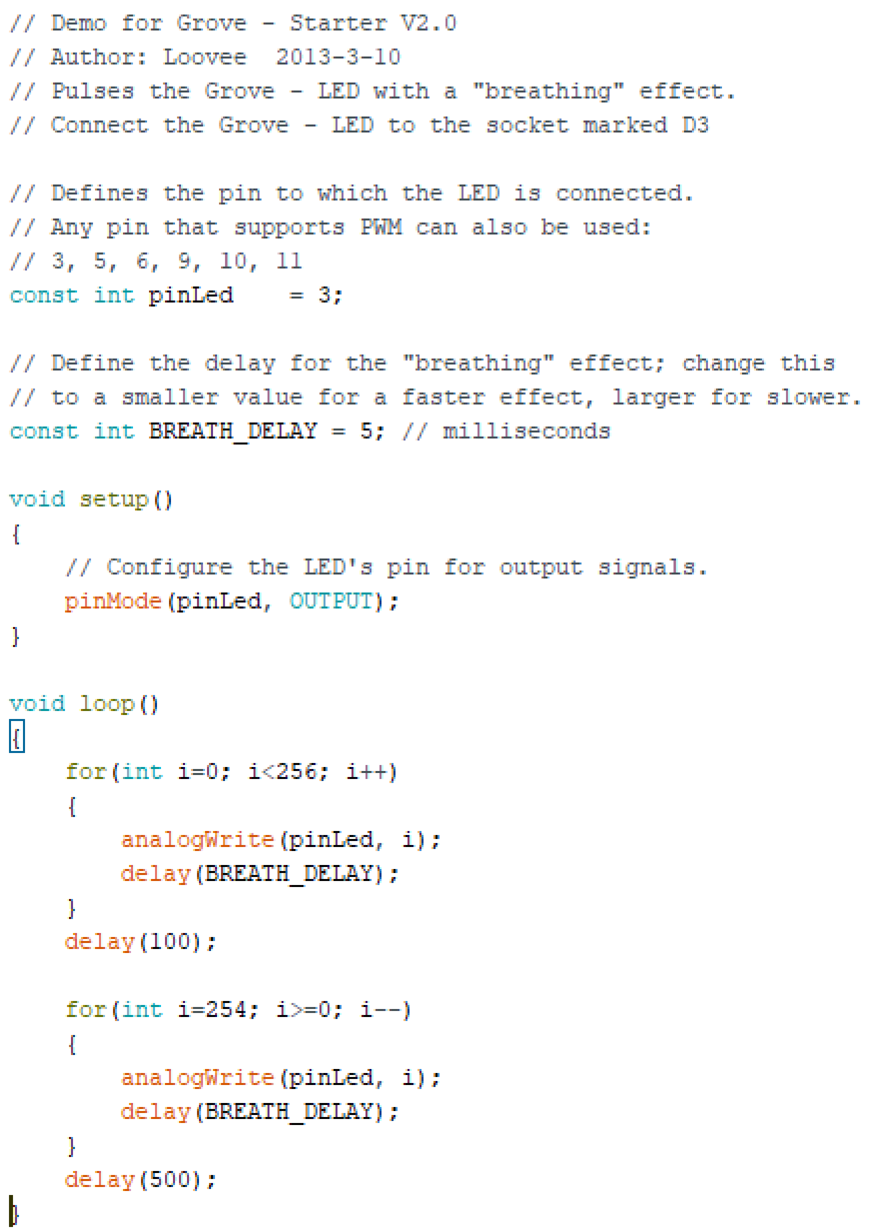
```
// Demo for Grove - Starter V2.0
// Author: Loovee  2013-3-10
// Pulses the Grove - LED with a "breathing" effect.
// Connect the Grove - LED to the socket marked D3

// Defines the pin to which the LED is connected.
// Any pin that supports PWM can also be used:
// 3, 5, 6, 9, 10, 11
const int pinLed    = 3;

// Define the delay for the "breathing" effect; change this
// to a smaller value for a faster effect, larger for slower.
const int BREATH_DELAY = 5; // milliseconds

void setup()
{
    // Configure the LED's pin for output signals.
    pinMode(pinLed, OUTPUT);
}

void loop()
{
    for(int i=0; i<256; i++)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(100);

    for(int i=254; i>=0; i--)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(500);
}
```

Datatypes

Type	Values		remarks
Void	No value		
Boolean	0 to 1	2^1	FALSE or TRUE
Byte	0 to 255	2^8	Only integer values
Int	-32.768 to 32.767	2^{16}	Only integer values
Unsigned Int	0 to 65.535		Only integer values
Long	-2.147.483.648 to 2.147.483.647	2^{32}	Only integer values
Unsigned Long	0 to 4.294.967.295		Only integer values
Float	~-3*10^38 tot ~3*10^38	2^{64}	Decimals allowed
Double Float	...		Decimals allowed
Char	-128 to 127		Usually only characters
Unsigned Char	0 to 255		
Array			Set of values
String			Array of char's

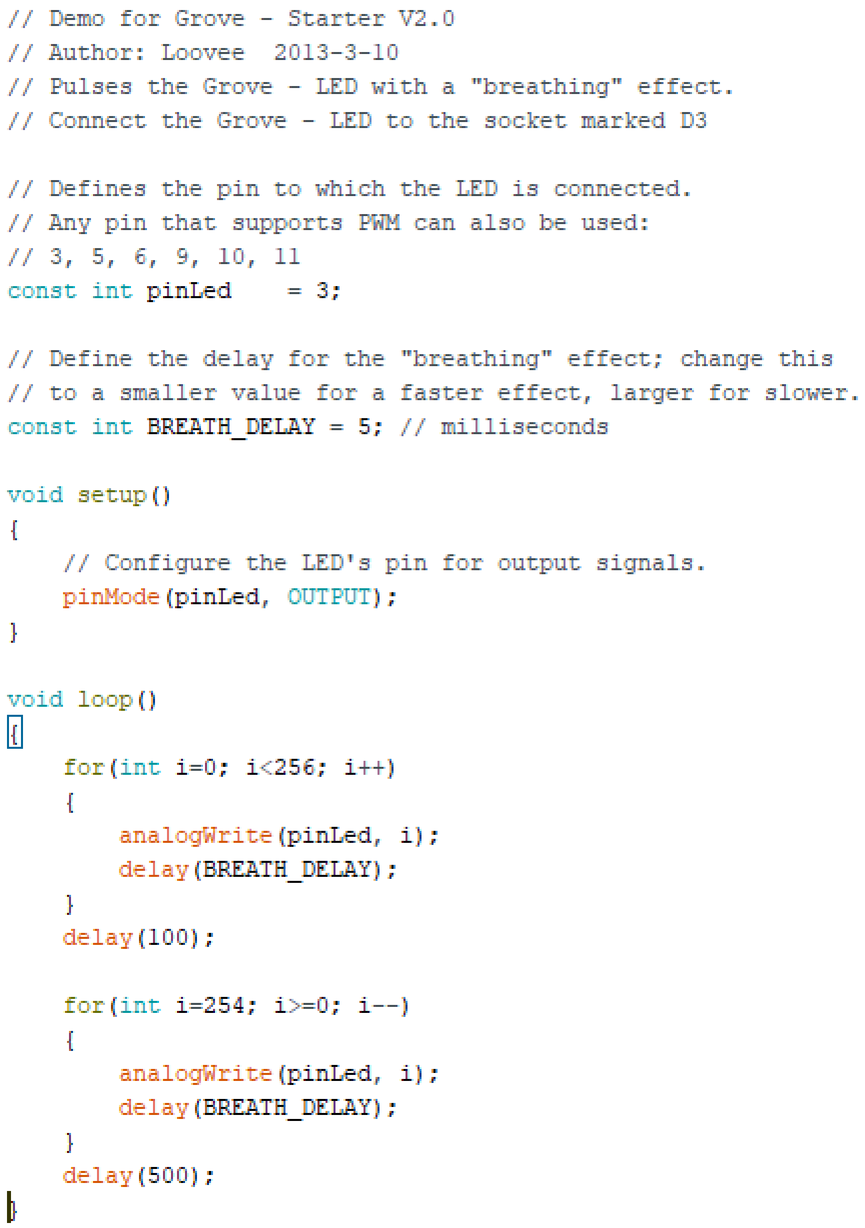
Programming in C

- Standard function Setup()
  - This function runs only once
  - Everything between the { } curly brackets is part of the setup function



```
// Demo for Grove - Starter V2.0
// Author: Loovee  2013-3-10
// Pulses the Grove - LED with a "breathing" effect.
// Connect the Grove - LED to the socket marked D3

// Defines the pin to which the LED is connected.
// Any pin that supports PWM can also be used:
// 3, 5, 6, 9, 10, 11
const int pinLed    = 3;

// Define the delay for the "breathing" effect; change this
// to a smaller value for a faster effect, larger for slower.
const int BREATH_DELAY = 5; // milliseconds

void setup()
{
    // Configure the LED's pin for output signals.
    pinMode(pinLed, OUTPUT);
}

void loop()
{
    for(int i=0; i<256; i++)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(100);

    for(int i=254; i>=0; i--)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(500);
}
```

Programming in C



```
// Demo for Grove - Starter V2.0
// Author: Loovee  2013-3-10
// Pulses the Grove - LED with a "breathing" effect.
// Connect the Grove - LED to the socket marked D3

// Defines the pin to which the LED is connected.
// Any pin that supports PWM can also be used:
// 3, 5, 6, 9, 10, 11
const int pinLed    = 3;

// Define the delay for the "breathing" effect; change this
// to a smaller value for a faster effect, larger for slower.
const int BREATH_DELAY = 5; // milliseconds

void setup()
{
    // Configure the LED's pin for output signals.
    pinMode(pinLed, OUTPUT);
}

void loop()
{
    for(int i=0; i<256; i++)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(100);

    for(int i=254; i>=0; i--)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(500);
}
```

- Standard function Loop()
  - This function loops until Arduino is reset
  - Everything between the { } curly brackets is part of the loop function

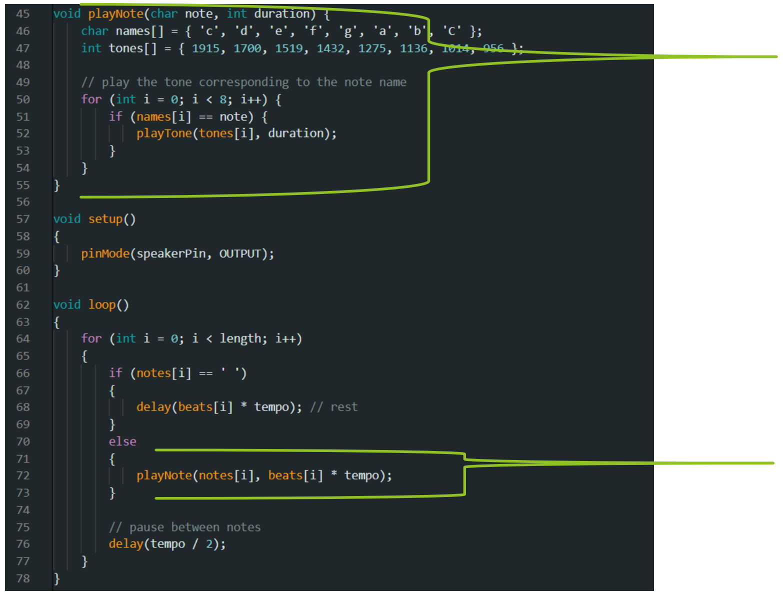
Custom function

Implement your own logic

Make some sound

- 1st exercise
  - Upload Grove_LED
  - Connect the buzzer to D3 instead of the LED

- 2nd exercise
  - Upload Grove_Buzzer



```
void playNote(char note, int duration) {
    char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };
    int tones[] = { 1915, 1700, 1519, 1432, 1275, 1136, 1014, 956 };

    // play the tone corresponding to the note name
    for (int i = 0; i < 8; i++) {
        if (names[i] == note) {
            playTone(tones[i], duration);
        }
    }
}

void setup()
{
    pinMode(speakerPin, OUTPUT);
}

void loop()
{
    for (int i = 0; i < length; i++)
    {
        if (notes[i] == ' ')
        {
            delay(beats[i] * tempo); // rest
        }
        else
        {
            playNote(notes[i], beats[i] * tempo);
        }

        // pause between notes
        delay(tempo / 2);
    }
}
```

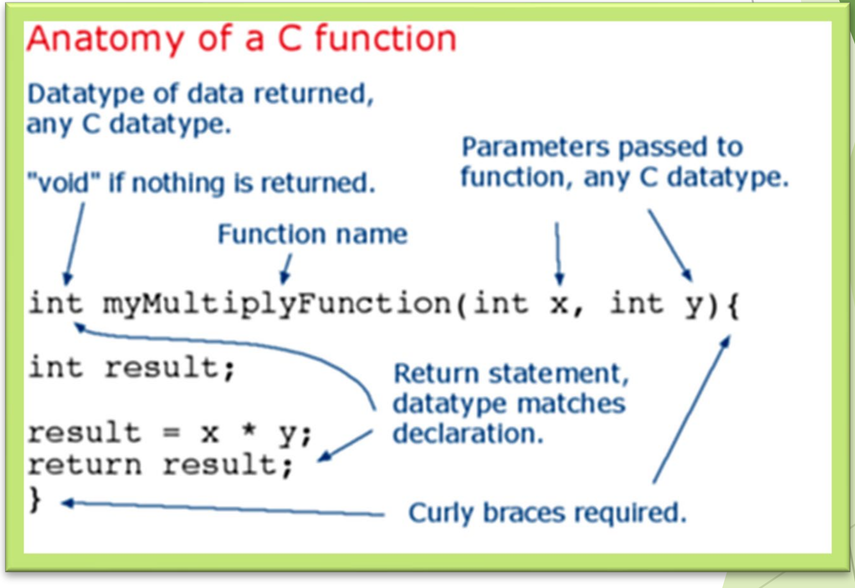
Custom function

Call the
custom function

Custom functions

- 2 required functions in an Arduino: setup() and loop()
- Other functions must be created outside the brackets of those two functions.

```
int myMultiplyFunction(int x, int y) {
    int result;
    result = x * y;
    return result;
}

void loop(){
    int i = 2;
    int j = 3;
    int k;
    k = myMultiplyFunction(i, j); // k now contains 6
}
```



Anatomy of a C function

Datatype of data returned, any C datatype.

"void" if nothing is returned.

Function name

Parameters passed to function, any C datatype.

```
int myMultiplyFunction(int x, int y){
int result;
result = x * y;
return result;
}
```

Return statement, datatype matches declaration.

Curly braces required.

Program flow

Structuring your code

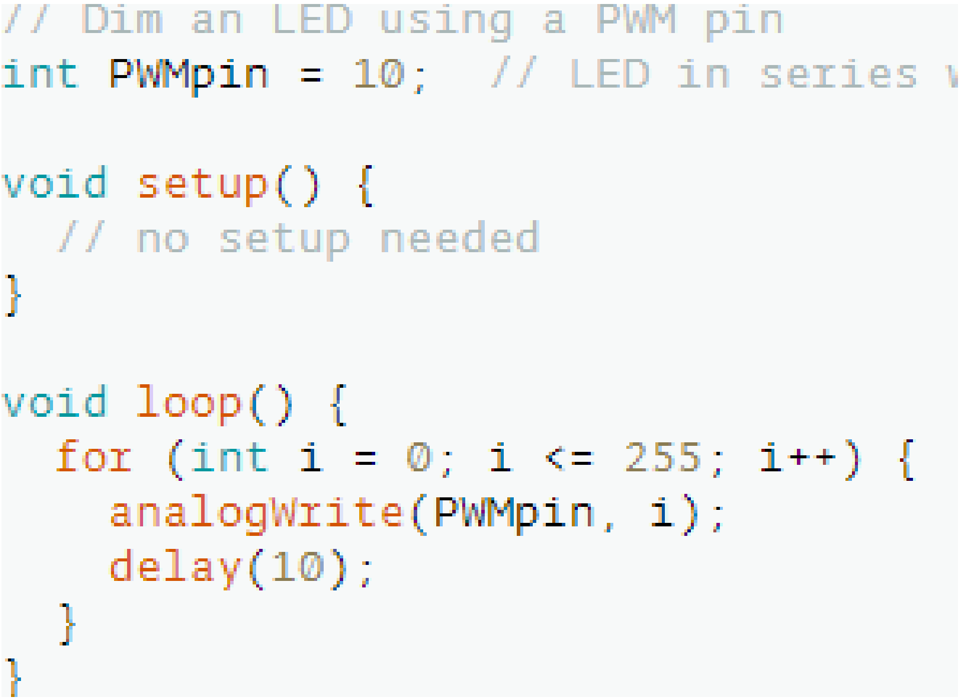
Control Structure

- For loop
- While loop
- If .. Else
- Custom functions

For Loop

- Repeat code x times
  - Initialize: Set i to 0
  - Condition: Each time through the loop, test if the condition is true
  - If true: Increment i with 1 and execute the code

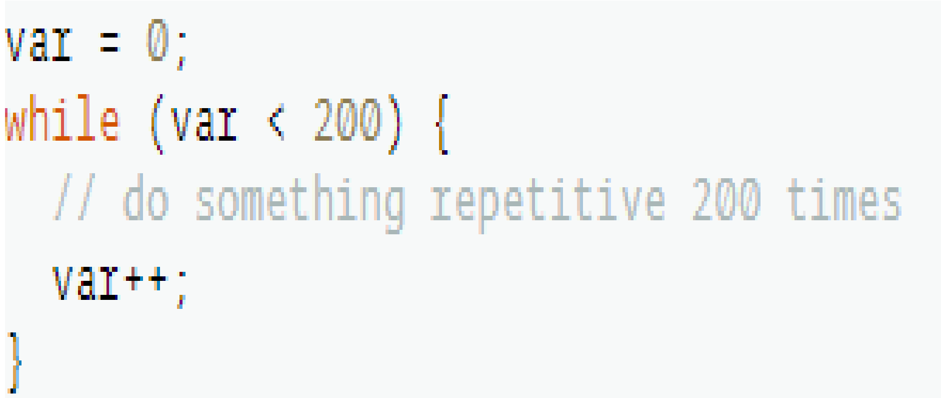
  - Pay attention to ( ) and { }



```
// Dim an LED using a PWM pin
int PWMpin = 10;   // LED in series

void setup() {
  // no setup needed
}

void loop() {
  for (int i = 0; i <= 255; i++) {
    analogWrite(PWMpin, i);
    delay(10);
  }
}
```

While loop

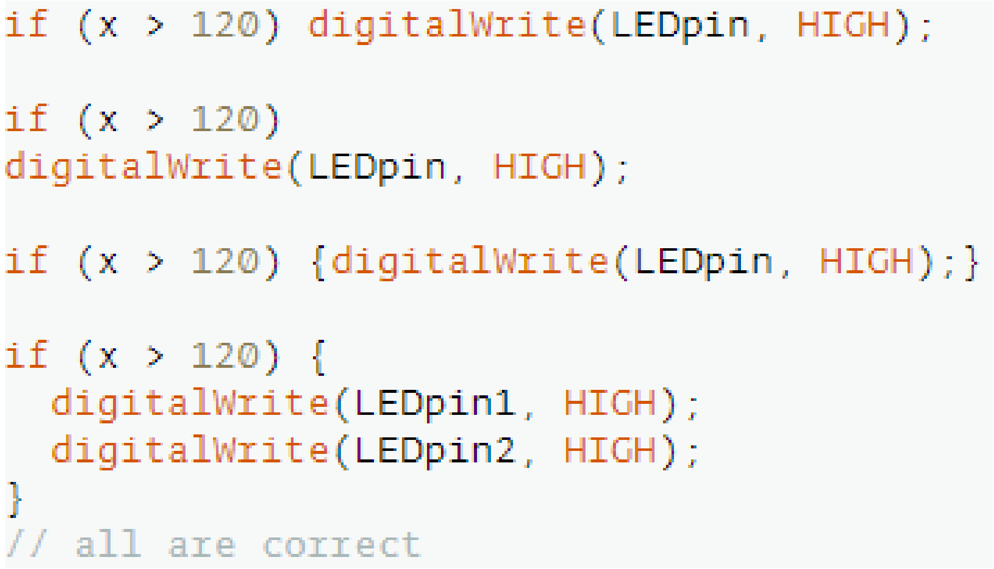
- Loop continuously and infinitely, until the tested variable becomes false
  - Keep looping as long as variable `var` is less than 200
  - `Var++` is short for `var = var + 1;`
- Something inside the loop must change the tested variable, or the while loop will never exit!
- The loop() function is a while loop



```
var = 0;
while (var < 200) {
  // do something repetitive 200 times
  var++;
}
```

IF-statement

- Checks for a condition and executes the following statement(s) if the condition is true
  - If x is greater than 120, turn LED on
- Several ways to use this statement
  - Keep your code clean
  - Use the bottom one!

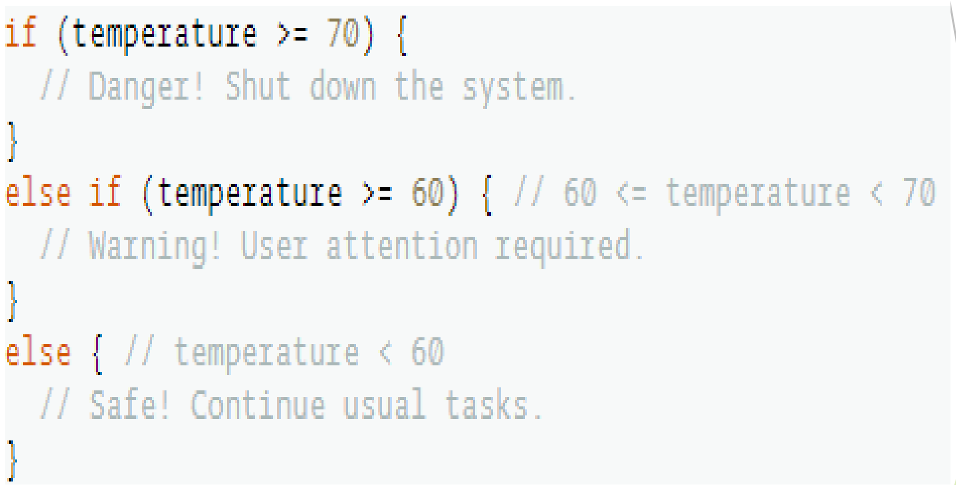


```
if (x > 120) digitalWrite(LEDpin, HIGH);

if (x > 120)
digitalWrite(LEDpin, HIGH);

if (x > 120) {digitalWrite(LEDpin, HIGH);}

if (x > 120) {
  digitalWrite(LEDpin1, HIGH);
  digitalWrite(LEDpin2, HIGH);
}
// all are correct
```

- Comparison statement is between ( )
- What needs to be done is between { }

IF-ELSE-statement

- If the IF condition is not true then test the next ELSE-IF condition.
- If all IF and ELSE-IF conditions are not true, then execute the ELSE code



```
if (temperature >= 70) {
  // Danger! Shut down the system.
}
else if (temperature >= 60) { // 60 <= temperature < 70
  // Warning! User attention required.
}
else { // temperature < 60
  // Safe! Continue usual tasks.
}
```

Comparison operators

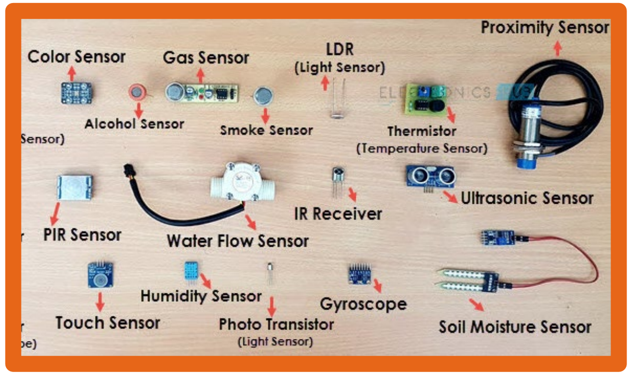
- != (not equal to)
- < (less than)
- <= (less than or equal to)
- == (equal to)

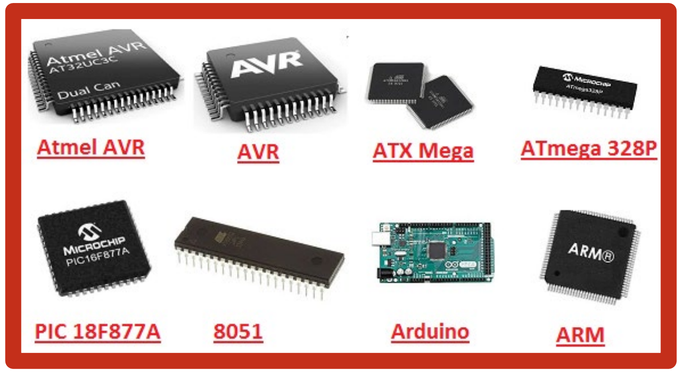
Note: there are 2 equal signs!

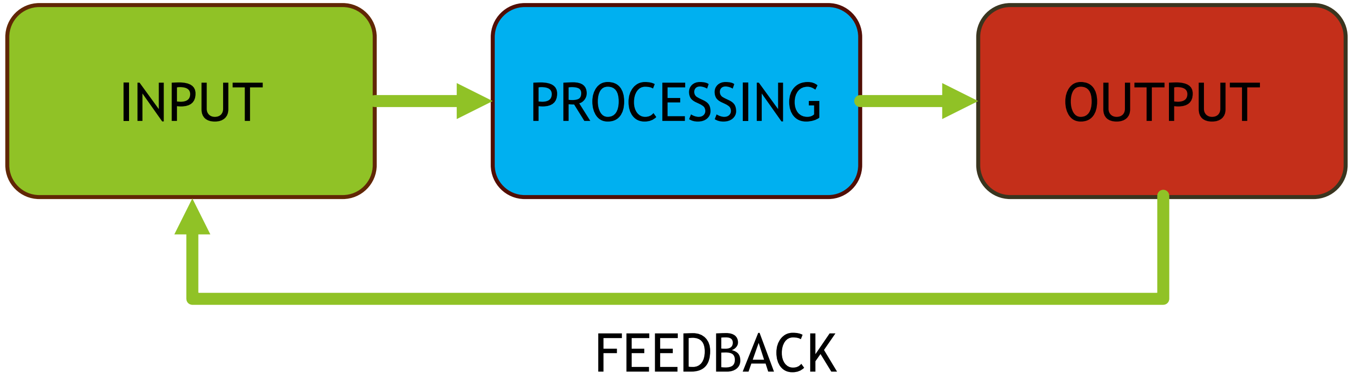
- > (greater than)
- >= (greater than or equal to)

- ! (logical not)
- && (logical and)
- || (logical or)

Use a sensor


Proximity Sensor
Color Sensor
Gas Sensor
LDR
(Light Sensor)
Alcohol Sensor
Smoke Sensor
Sensor)
Thermistor
(Temperature Sensor)
Ultrasonic Sensor
IR Receiver
PIR Sensor
Water Flow Sensor
Humidity Sensor
Gyroscope
Touch Sensor
Photo Transistor
(Light Sensor)
Soil Moisture Sensor


Atmel AVR
AVR
ATX Mega
ATmega 328P
PIC 18F877A
8051
Arduino
ARM


INPUT
PROCESSING
OUTPUT
FEEDBACK

Use a sensor

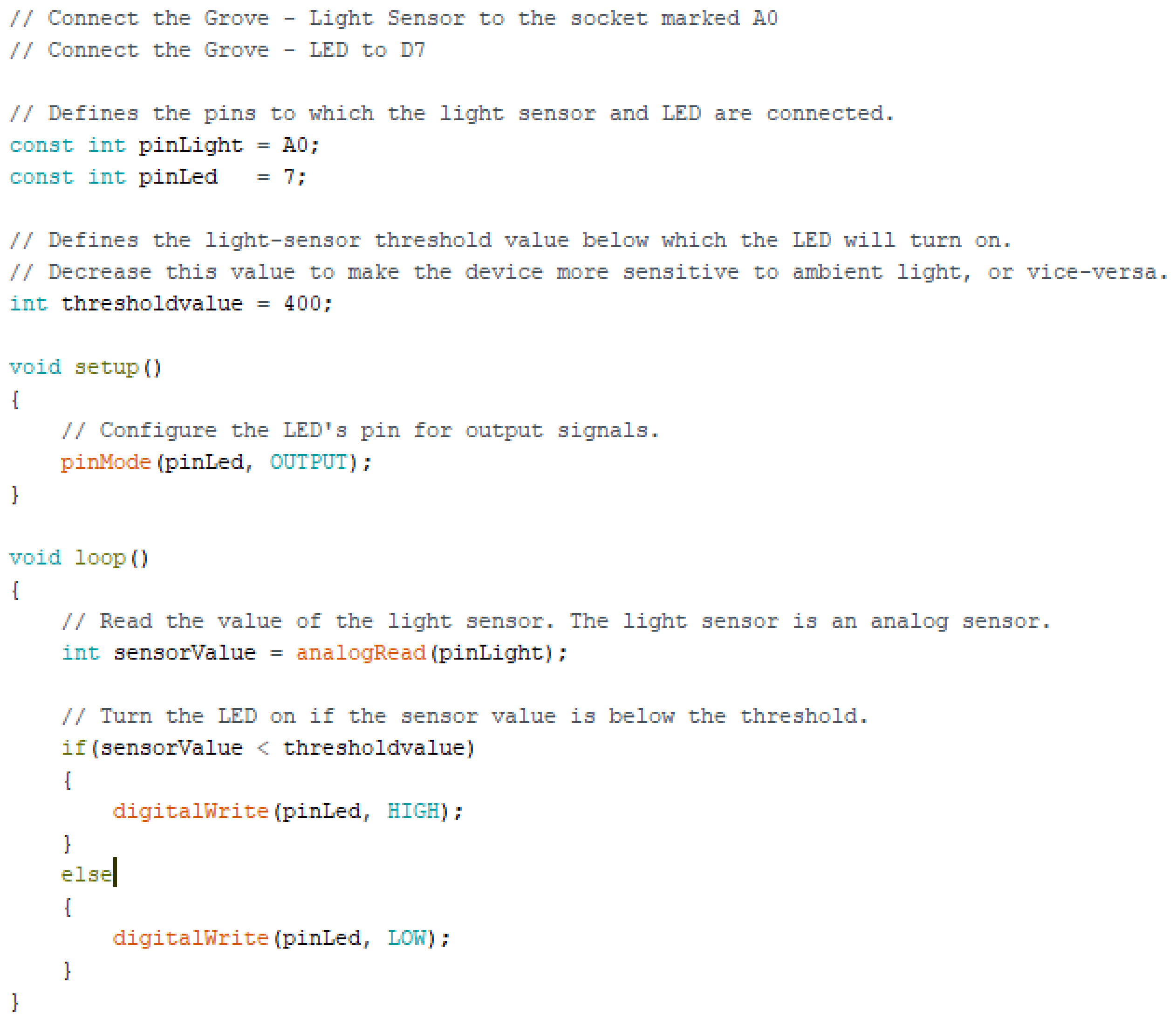
- Open Grove_Light_Sensor
- Connect the Grove light sensor and the LED (see comments in the code)
- Upload Grove_Light_Sensor

analogRead()

- Read an analog value
  - Potentiometers
  - Sensors
- analogRead*(pinname)*
  - 10-bit analog to digital converter
  - 1024 values
  - 5 volts / 1024 units: 4.9 mV per unit
  - 100 microseconds to read ADC input
  - so max rate is about 10,000/s

- *Try the following analogWrite example:*

  Grove_Light_Sensor



```
// Connect the Grove - Light Sensor to the socket marked A0
// Connect the Grove - LED to D7

// Defines the pins to which the light sensor and LED are connected.
const int pinLight = A0;
const int pinLed   = 7;

// Defines the light-sensor threshold value below which the LED will turn on.
// Decrease this value to make the device more sensitive to ambient light, or vice-versa.
int thresholdvalue = 400;

void setup()
{
    // Configure the LED's pin for output signals.
    pinMode(pinLed, OUTPUT);
}

void loop()
{
    // Read the value of the light sensor. The light sensor is an analog sensor.
    int sensorValue = analogRead(pinLight);

    // Turn the LED on if the sensor value is below the threshold.
    if(sensorValue < thresholdvalue)
    {
        digitalWrite(pinLed, HIGH);
    }
    else
    {
        digitalWrite(pinLed, LOW);
    }
}
```

Best practice

- Use meaningful names for variables

  buttonPressCounter

  - instead of bcnt
- Use constants instead of numbers in your code

  Const int delayTime = 100

  delay(delayTime)

  - instead of delay(100)
- Write comments to explain your code
- Use custom functions instead of copy/paste the same code
- Use proper indentation

```
void loop() {
_ _ for (position = 0; position <= 180; position += 1) {
- - - - if (currentPosition < endPosition) {
- - - - - - currentPosition = endPosition;
- - - - }
- - }
}
```

Standard Functions

- Digital output
  - (LED, PWM) → digitalWrite()
- Digital input → digitalRead()
  - (button, some sensors)
- Analog input → analogRead()
  - (sensors)
- Analog output → analogWrite()
- Delay (time in milliseconds) → delay()

- Reference → https://www.arduino.cc/reference/en

digitalRead() & digitalWrite()

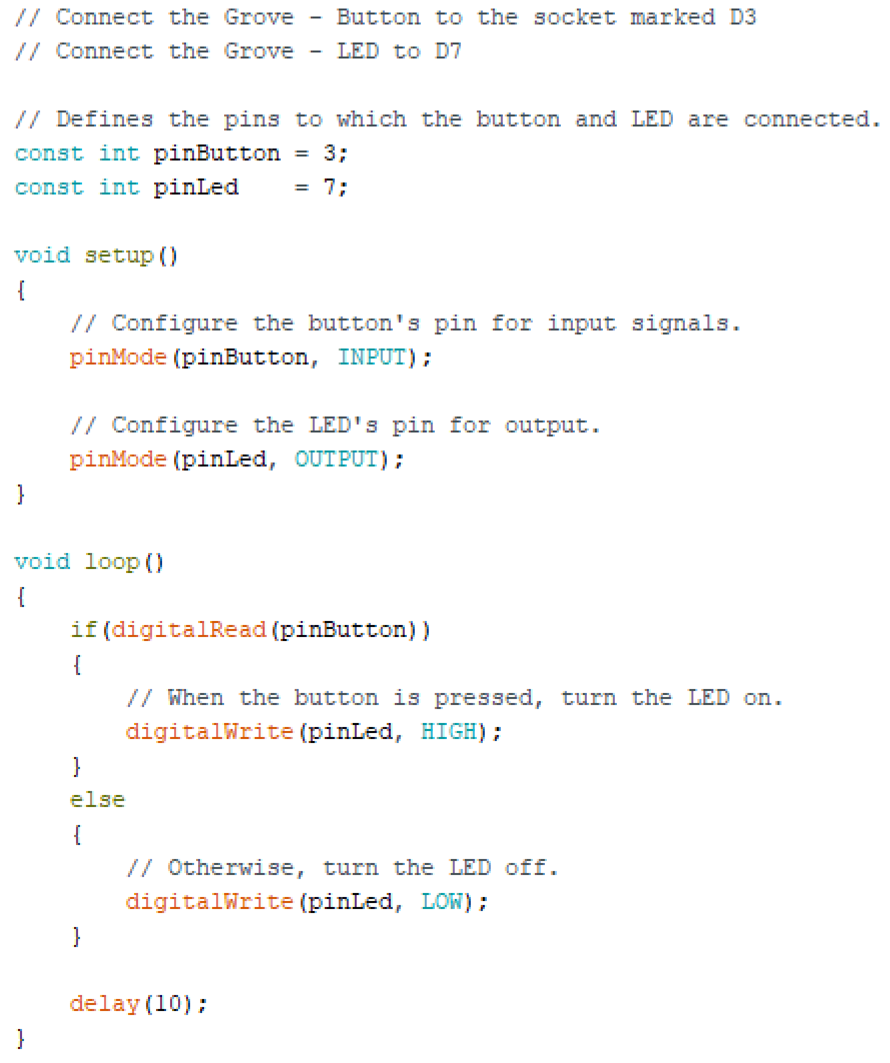
- Read value of a pin / write value to a pin
  - HIGH of LOW

- digitalRead(*pinname)*
- digitalWrite*(pinname, value)*

- *digitalWrite example: Blink*

- *Try the following digitalRead example:*

  Grove_Button



```
// Connect the Grove - Button to the socket marked D3
// Connect the Grove - LED to D7

// Defines the pins to which the button and LED are connected.
const int pinButton = 3;
const int pinLed    = 7;

void setup()
{
    // Configure the button's pin for input signals.
    pinMode(pinButton, INPUT);

    // Configure the LED's pin for output.
    pinMode(pinLed, OUTPUT);
}

void loop()
{
    if(digitalRead(pinButton))
    {
        // When the button is pressed, turn the LED on.
        digitalWrite(pinLed, HIGH);
    }
    else
    {
        // Otherwise, turn the LED off.
        digitalWrite(pinLed, LOW);
    }

    delay(10);
}
```

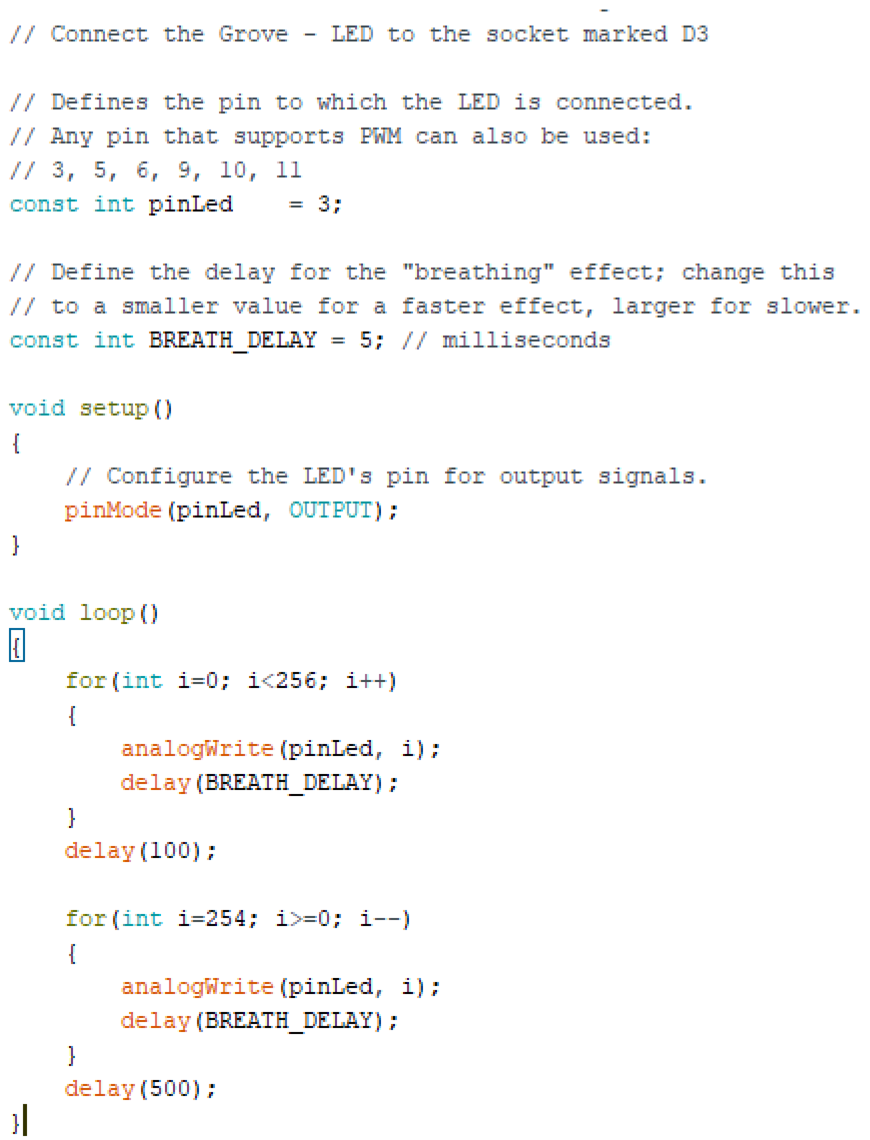
analogWrite()

- Output an analog value
  - Servo motors
  - LED dimming
- Only on analog ports (A0 ~A3)
- 256 values, 0 until 255
- analogWrite(*pinname*, *var*);

- *Try the following analogWrite example:*

  Grove_LED

- This example uses PWM:
  - Pulse Width Modulation



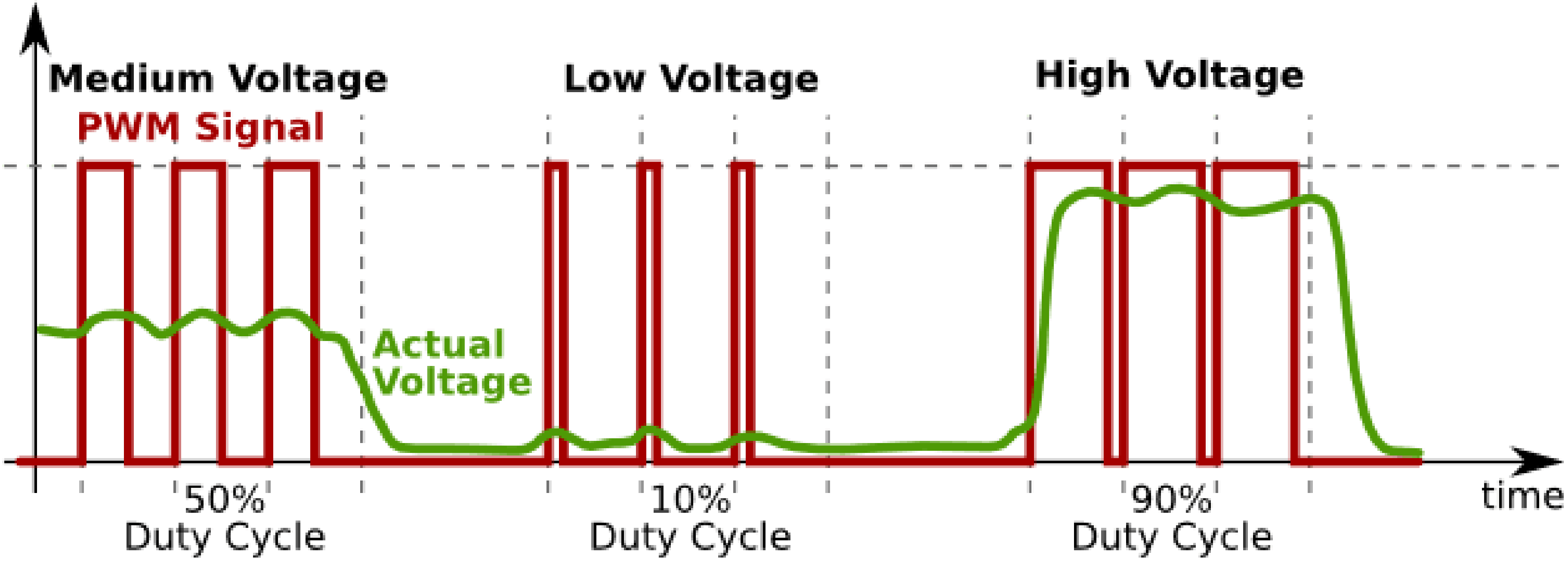
```
// Connect the Grove - LED to the socket marked D3

// Defines the pin to which the LED is connected.
// Any pin that supports PWM can also be used:
// 3, 5, 6, 9, 10, 11
const int pinLed    = 3;

// Define the delay for the "breathing" effect; change this
// to a smaller value for a faster effect, larger for slower.
const int BREATH_DELAY = 5; // milliseconds

void setup()
{
    // Configure the LED's pin for output signals.
    pinMode(pinLed, OUTPUT);
}

void loop()
{
    for(int i=0; i<256; i++)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(100);

    for(int i=254; i>=0; i--)
    {
        analogWrite(pinLed, i);
        delay(BREATH_DELAY);
    }
    delay(500);
}
```

analogWrite() uses PWM

- *Try the following analogWrite example:*

  Grove_LED

- This example uses PWM: Pulse Width Modulation
- PWM is not available on all Arduino pins. Check for the ~ sign on the PCB.



Servo motor

- Open Grove_servo
- Connect the Grove servo and positional sensor (see comments in the code)
- Upload Grove_servo

- Servo motors use pulse with modulation too!

- What happens if you swap the positional sensor with the light sensor?

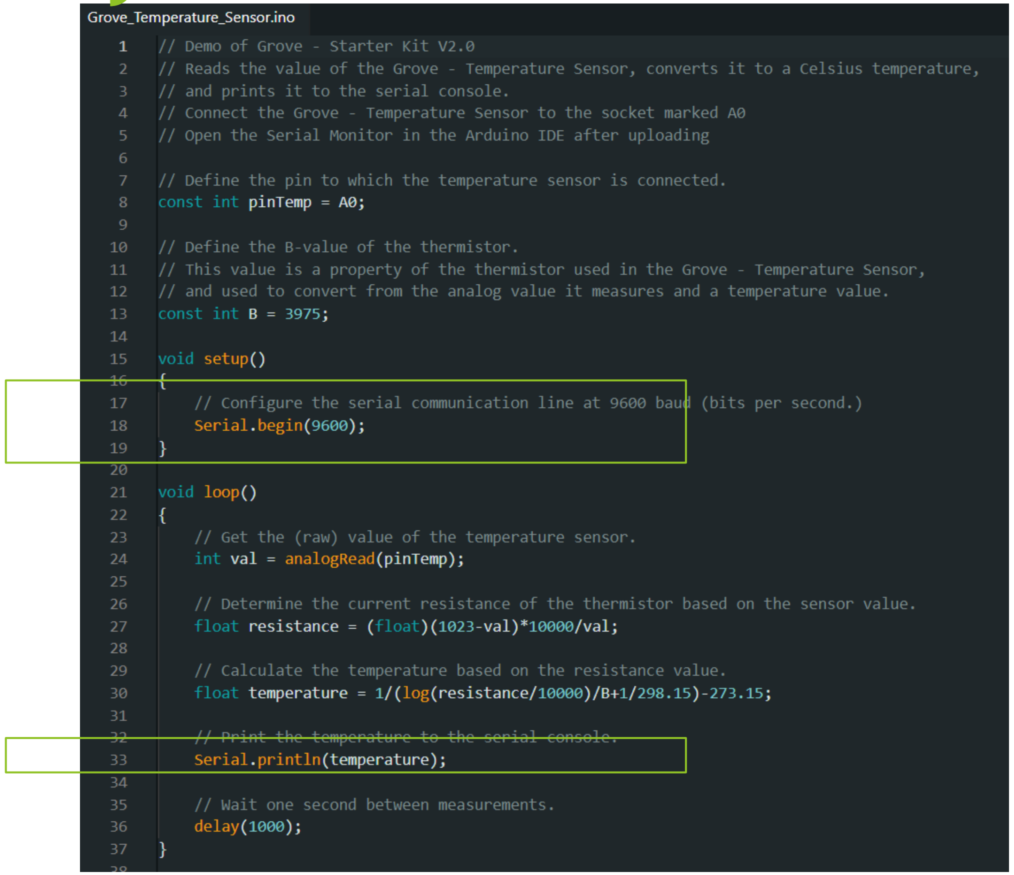
Where’s the bug

Temperature sensor

- Open Grove_Temperature_Sensor
- Connect the Grove temperature sensor (see comments in the code)
- Upload Grove_Temperature_Sensor
- Open the Serial Monitor and set it to 9600

Add communication with your PC

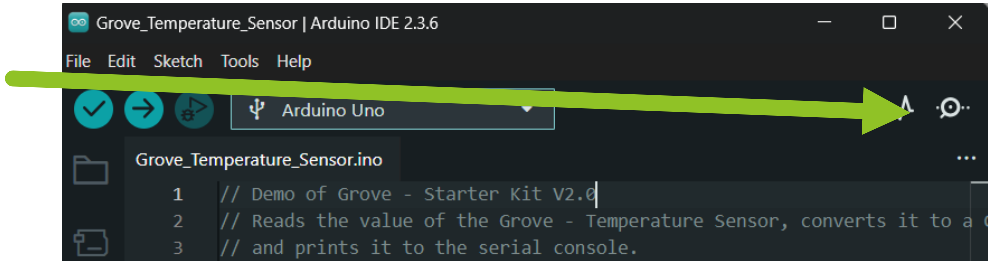
- Add to setup()
  - `Serial.begin(9600);`
- Add to loop()
  - `Serial.println(temperature);`



Grove_Temperature_Sensor.ino

```
// Demo of Grove - Starter Kit V2.0
// Reads the value of the Grove - Temperature Sensor, converts it to a Celsius temperature,
// and prints it to the serial console.
// Connect the Grove - Temperature Sensor to the socket marked A0
// Open the Serial Monitor in the Arduino IDE after uploading

// Define the pin to which the temperature sensor is connected.
const int pinTemp = A0;

// Define the B-value of the thermistor.
// This value is a property of the thermistor used in the Grove - Temperature Sensor,
// and used to convert from the analog value it measures and a temperature value.
const int B = 3975;

void setup()
{
    // Configure the serial communication line at 9600 baud (bits per second.)
    Serial.begin(9600);
}

void loop()
{
    // Get the (raw) value of the temperature sensor.
    int val = analogRead(pinTemp);

    // Determine the current resistance of the thermistor based on the sensor value.
    float resistance = (float)(1023-val)*10000/val;

    // Calculate the temperature based on the resistance value.
    float temperature = 1/(log(resistance/10000)/B+1/298.15)-273.15;

    // Print the temperature to the serial console.
    Serial.println(temperature);

    // Wait one second between measurements.
    delay(1000);
}
```

- Open Serial Monitor and see what happens
- Can you Serial.print other information?
- Open Serial Plotter and see what happens


Grove_Temperature_Sensor | Arduino IDE 2.3.6
File
Edit
Sketch
Tools
Help
Arduino Uno
Grove_Temperature_Sensor.ino

```
// Demo of Grove - Starter Kit V2.0
// Reads the value of the Grove - Temperature Sensor, converts it to a
// and prints it to the serial console.
```

Debugging

- once you run your program, you can’t see what’s happening inside
- Important: Plan you project!
  - Design → what is it, what should it do and how
  - Build → select the required hardware modules and connect them
  - Test → check that the modules are properly connected using test code
  - Code → program the required functionality in small chunks at a time and test
  - Debug → use serial monitor to print debug information to your computer

Debugging using serial monitor

- Print the following information:
  - Variables → print value of your variable just before you need it
  - Inputs → print sensor readings, button states
  - Outputs → print value you want them to be before writing them to the pin
  - Program flow → Print text to indicate the programs flow, i.e. inside an ‘if’ statement to see whether the condition was met
  - Anything else you find important to print to screen

Libraries

More functions for you

RGB LCD Display

- Open Grove RGB display -> HelloWorld
- Connect the Grove RGB Display (see comments in the code)
- Install the library lcd_rgb
- Upload the Hello World sketch

- Note: The RGB LCD Display code uses a library to do the hard work

Add library to Arduino IDE

- Search for the library in Arduino IDE
- OR download

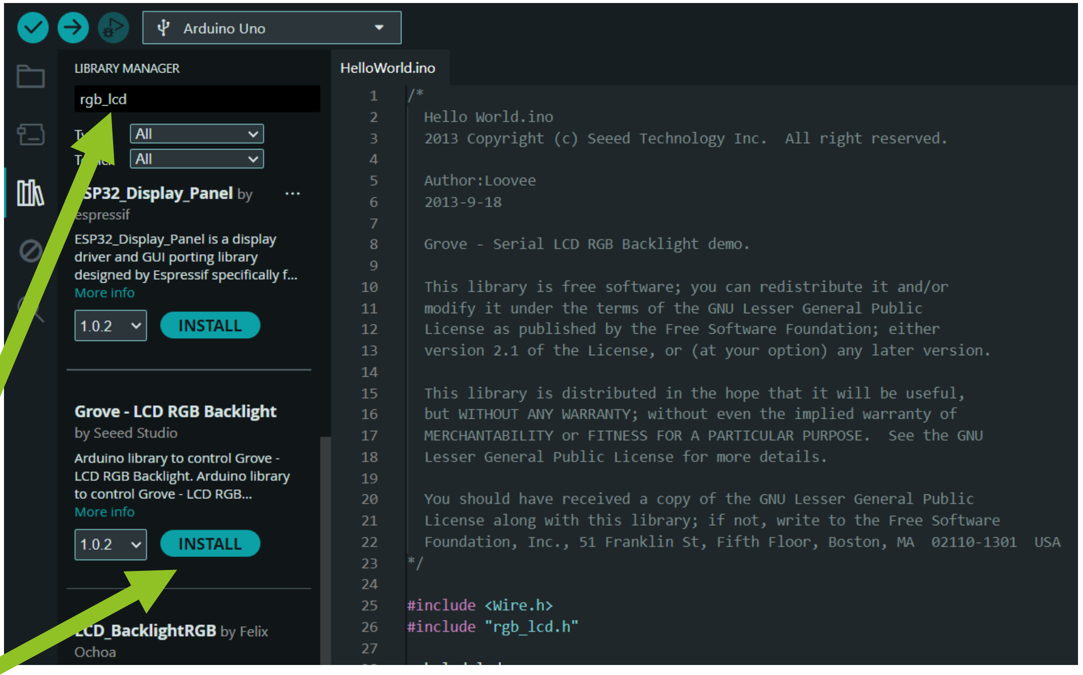
  e.g. https://github.com/Seeed-Studio/Grove_LCD_RGB_Backlight/archive/master.zip

  and save in folder ~/documents/Arduino/libraries

1 Open Library Manager

2 Search for “rgb_lcd”

3 click install



rgb_lcd

SP32_Display_Panel by espressif

ESP32_Display_Panel is a display driver and GUI porting library designed by Espressif specifically f...

Grove - LCD RGB Backlight by Seeed Studio

Arduino library to control Grove - LCD RGB Backlight. Arduino library to control Grove - LCD RGB...

CD_BacklightRGB by Felix Ochoa

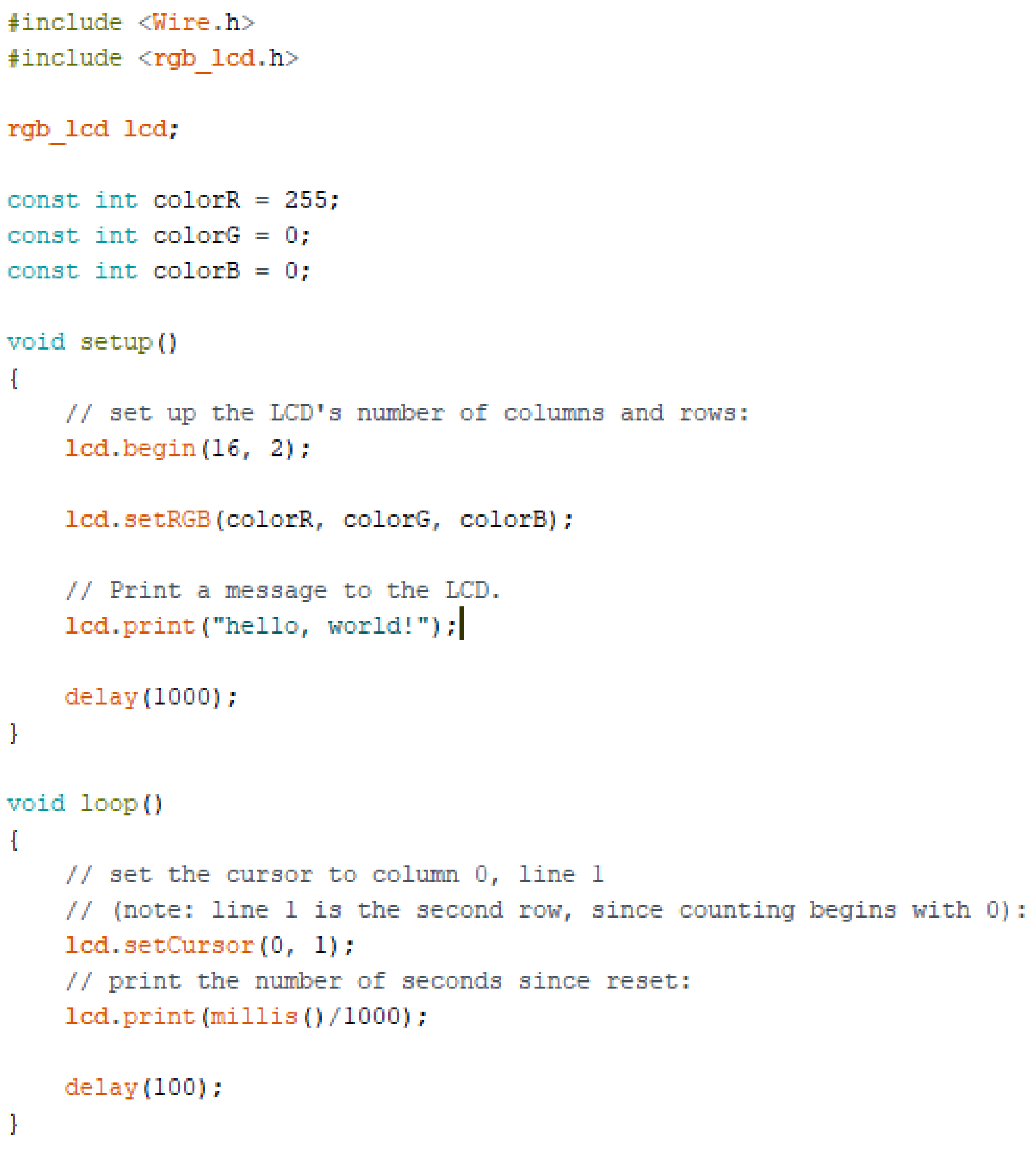
```
/*
  Hello World.ino
  2013 Copyright (c) Seeed Technology Inc.  All right reserved.

  Author:Loovee
  2013-9-18

  Grove - Serial LCD RGB Backlight demo.

  This library is free software; you can redistribute it and/or
  modify it under the terms of the GNU Lesser General Public
  License as published by the Free Software Foundation; either
  version 2.1 of the License, or (at your option) any later version.

  This library is distributed in the hope that it will be useful,
  but WITHOUT ANY WARRANTY; without even the implied warranty of
  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
  Lesser General Public License for more details.

  You should have received a copy of the GNU Lesser General Public
  License along with this library; if not, write to the Free Software
  Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301  USA
*/

#include <Wire.h>
#include "rgb_lcd.h"

```

Libraries

- Libraries full of functions:
  - Build in standard libraries
    - https://www.arduino.cc/reference/en/libraries/
  - Custom libraries
    - E.g. https://github.com/Seeed-Studio/Grove_LCD_RGB_Backlight/archive/master.zip
- Libraries are mostly used to interface with “advanced” modules that require communication protocols
  - UART (serial)
  - SPI
  - I2C
  - Onewire

Using libraries

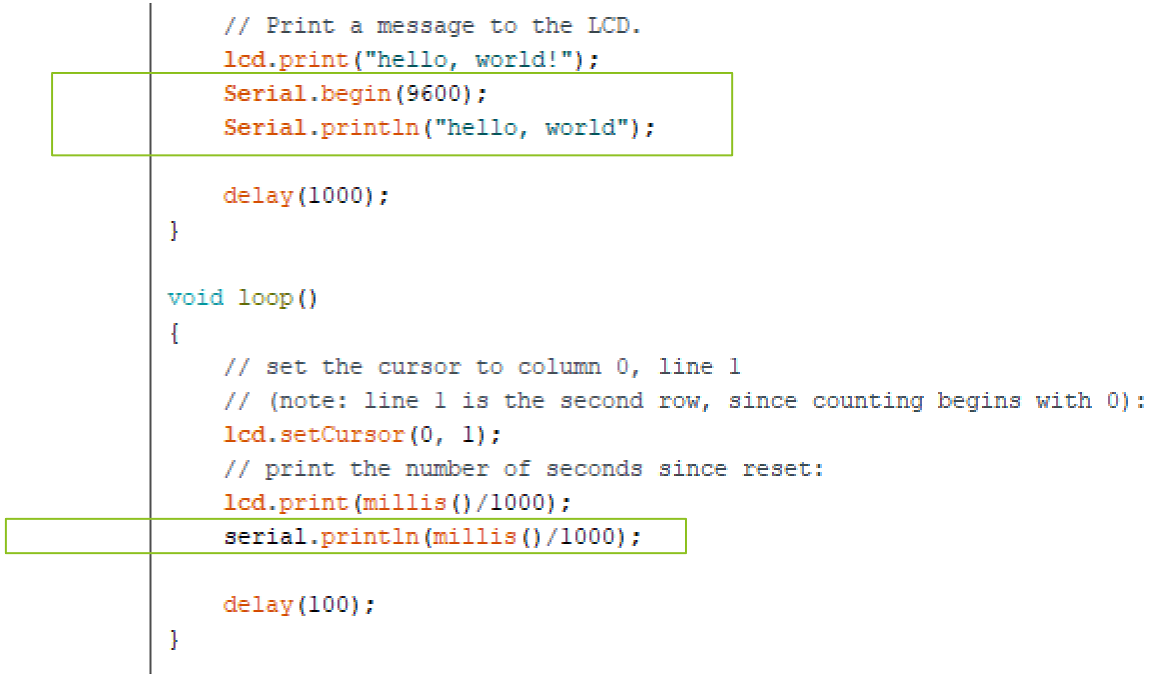
- Import the library
  - #include
- Instantiate a class
  - Required because you can re-use libraries for multiple objects (e.g. 2 LCDs)
  - Rgb_lcd lcd;
  - First library name
  - then the name of your new object
- Call function from the library
  - `Lcd.begin(16,2)`
  - First object name
  - Followed by a dot .
  - Finally the function



```
#include <Wire.h>
#include <rgb_lcd.h>

rgb_lcd lcd;

const int colorR = 255;
const int colorG = 0;
const int colorB = 0;

void setup()
{
    // set up the LCD's number of columns and rows:
    lcd.begin(16, 2);

    lcd.setRGB(colorR, colorG, colorB);

    // Print a message to the LCD.
    lcd.print("hello, world!");

    delay(1000);
}

void loop()
{
    // set the cursor to column 0, line 1
    // (note: line 1 is the second row, since counting begins with 0):
    lcd.setCursor(0, 1);
    // print the number of seconds since reset:
    lcd.print(millis()/1000);

    delay(100);
}
```

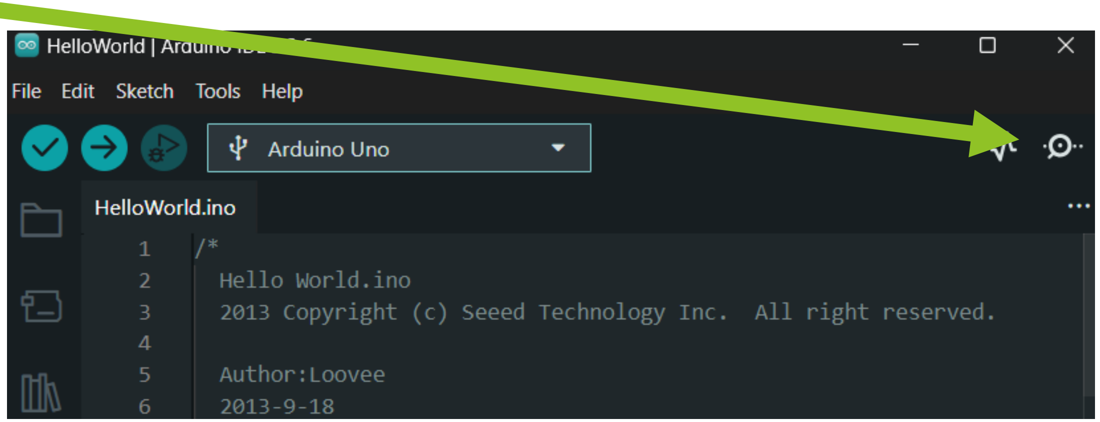
Add communication with your PC

- Add to setup()
  - `Serial.begin(9600);`
  - `Serial.println(“hello, world”);`
- Add to loop()
  - `Serial.println(millis()/1000);`



```
    // Print a message to the LCD.
    lcd.print("hello, world!");
    Serial.begin(9600);
    Serial.println("hello, world");

    delay(1000);
}

void loop()
{
    // set the cursor to column 0, line 1
    // (note: line 1 is the second row, since counting begins with 0):
    lcd.setCursor(0, 1);
    // print the number of seconds since reset:
    lcd.print(millis()/1000);
    serial.println(millis()/1000);

    delay(100);
}
```

- Open Serial Monitor and see what happens



```
/*
  Hello World.ino
  2013 Copyright (c) Seeed Technology Inc.  All right reserved.

  Author:Loovee
  2013-9-18
```

- Can you combine this code with the temperature sensor code so the temperature gets onto the screen?

End of this presentation