Microcontroller Programming with the Arduino

Contents

[Intro 2](#_Toc453331332)

[Audience 2](#_Toc453331333)

[Your kit 2](#_Toc453331334)

[Development Environment 4](#_Toc453331335)

[Connecting the Arduino 4](#_Toc453331336)

[Setting up your programming environment 4](#_Toc453331337)

[Ubuntu 4](#_Toc453331338)

[Windows 7/8/10, OS X, Linux (other) 4](#_Toc453331339)

[Interface basics 5](#_Toc453331340)

# Intro

## Audience

This tutorial pdf is intended for those students currently studying the first year programming course in C. It will teach you the basics of programming the arduino microcontroller without any assumptions that the reader knows anything about electronics, all basic concepts are learnt along the way.

## Your kit

As part of your course you should have access to the arduino and the ICs (integrated circuits allow you to extend the functionality of your arduino), ask your teacher about these if you haven’t got a hold of them yet (you may have to pay a small fee for the kit).

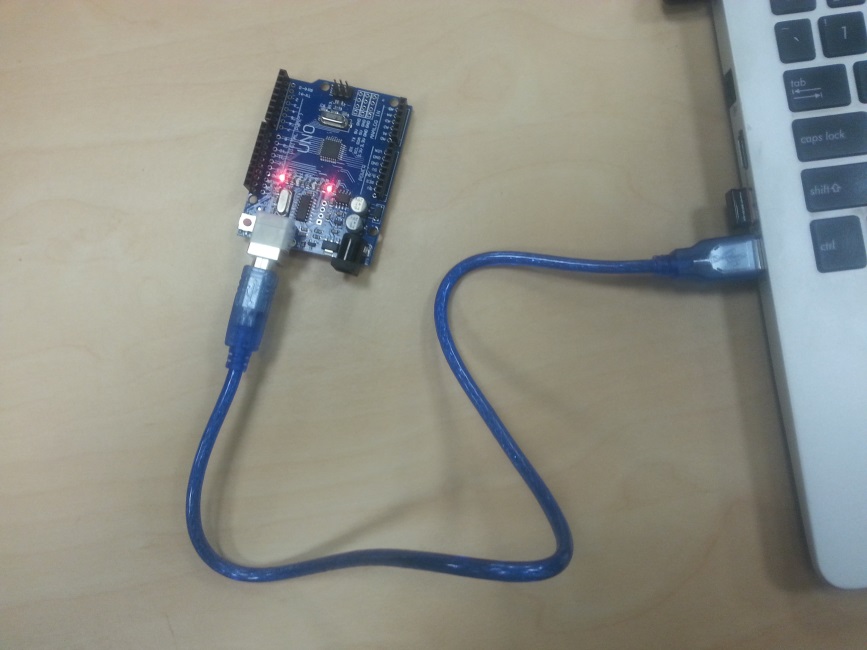
Here is a small decription of each element (don’t worry if you don’t have a particular item, it may have been taken out for repairs or been removed from the course).

|  |
| --- |
| C:\Users\s2807774\Documents\GitHub\ArduinoTutes\ims\arduino.png |
| The Arduino microcontroller   * The USB connection is where you will plug your Arduino into your desktop or laptop. * The reset button resets the program currently running on the Arduino. * The digital pins can read and write output/input 1s and 0s (bits) as high (5 volt) and low (less than 3 volts) voltages. They can also simulate analogue output in the form of something called pulse width modulation, which is a fancy name for flipping between high and low voltage at different speeds so that the output is high for some percentage of time to simulate analogue output. * The pin13 light is an LED (light emitting diode), if you write a 1 to pin13, the light will turn on. * The micro-controller processes your program to control the pins on the Arduino. * The 3.3 volt and 5 volt pin supply voltage to devices connected to the Arduino. * The ground pins complete the circuit from devices connected to the digital/analogue output or 5v/3.3v pins. * The analogue pins can read analogue input. They can also do general I/O too. |

|  |  |  |
| --- | --- | --- |
| digital multi-function shield   * Is connected to the top of your Arduino * Has a reset button too * Has 3 buttons which you can read input * Has 4 LED lights * Has a 7 segment (digital alarm clock) display | LCD shield   * Can write text to lCD screen * Has several buttons * Turn the screw on the blue box to adjust screen brightness | C:\Users\s2807774\Documents\GitHub\ArduinoTutes\ims\buzzer.png  Buzzer   * Can make annoying sounds at different pitches |
| Sound Sensor   * Can detect sound * Turn the screw to adjust sensitivity | C:\Users\s2807774\Documents\GitHub\ArduinoTutes\ims\obstacle sensor.png  Obstacle sensor   * Detects if an object is within a defined range * Turn the screw to change the range | Ultrasonic sensor   * Detects the distance between the object it faces and itself |
| Digital compass   * Detects the axis it is rotated at | IMU – Internal Measurement Unit   * Detects the axis it is rotated at * Detects accelerations it is put through | Temperature sensor   * Can detect the temperature of the current environment |

# Development Environment

## Connecting the Arduino



The first step is to get your cable, connect one end to the Arduino and the other to the USB port of your laptop of Desktop. Use the picture above as a reference, some lights on the Arduino should light up.

## Setting up your programming environment

We will be using the Arduino IDE (integrated development environment) to write and upload out programs to the Arduino device. The website for this IDE is: <https://www.arduino.cc/en/Main/Software> .

Installation instructions are as follows:

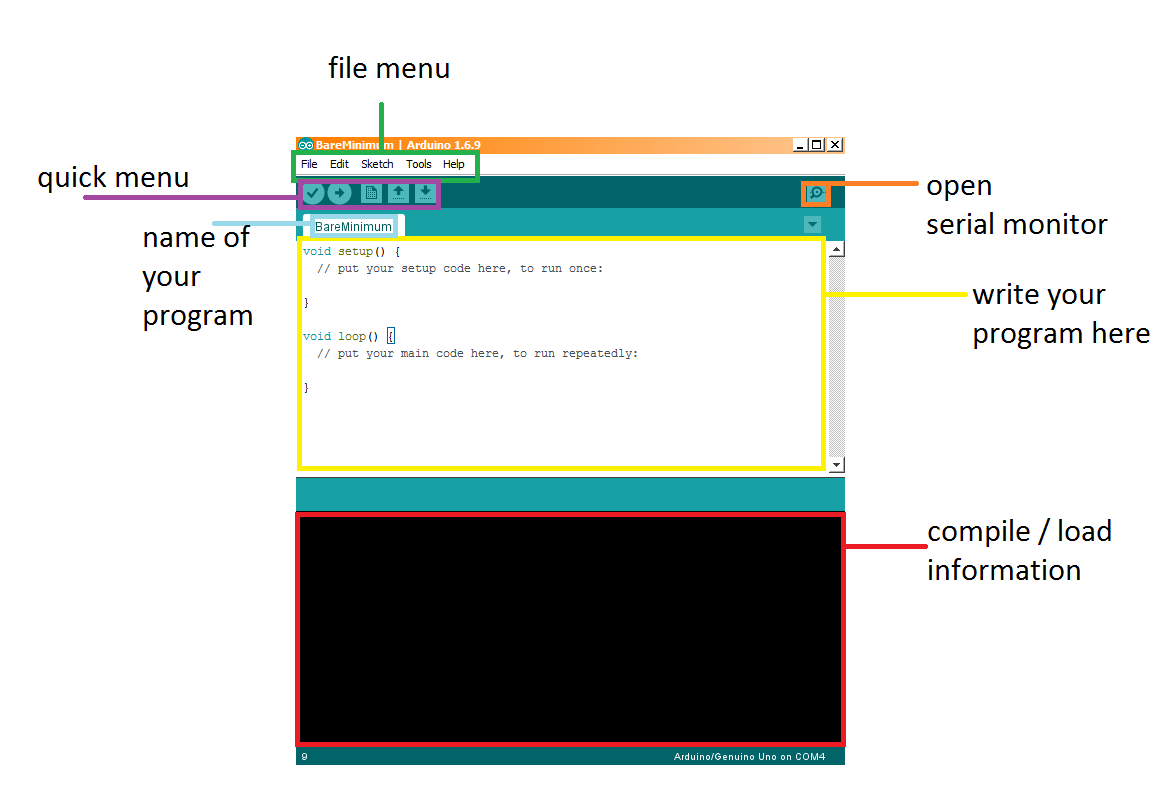
### Ubuntu

1. Open the terminal (Ctrl + t)
2. Paste in: sudo-apt get install arduino arduino-core
3. Follow prompts

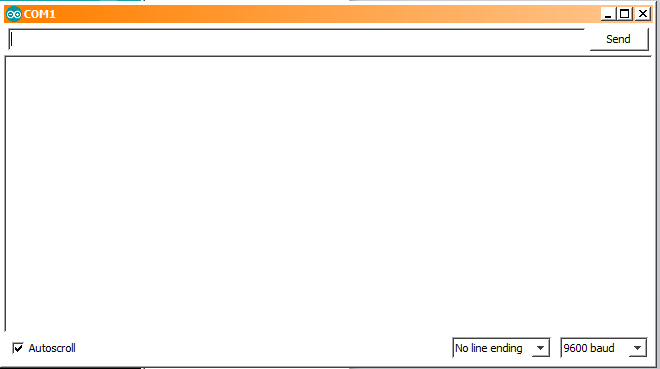
### Windows 7/8/10, OS X, Linux (other)

1. Go to <https://www.arduino.cc/en/Main/Software>
2. Download the correct installer
3. follow prompts

## Interface basics



* In the file menu (green) you can do various operations such as:
  + File -> open/save your program (called a sketch)
  + Edit -> find/replace/copy and paste
  + Sketch -> compile/upload program to Arduino
  + Tools -> set the port the Arduino is connected to, set the board type (we are using the UNO)
* The text-editor (yellow) is where you will write your programs
* The compile/load information section (red) displays compile time information
* The quick menu has 5 buttons
  + The tick button compiles your program
  + The arrow button uploads the program to the Arduino (only if it compiles first)
  + The rest of the buttons are: Open a new file, open a specific file, save
* Serial Monitor open button (orange)
  + Picture below
  + Allows the developer to send information to the Arduino
  + Allows the developer to read information sent from the Arduino
  + If Auto scroll is checked, the latest message from the Arduino is focused on
  + 9600 baud rate is the rate at which the Arduino is communicating with your IDE, 9600 baud is 9600 bits per second

.

# Programming the Arduino

## Most basic Program (Sketch)

The language we will be using to program the Arduino will be a variation of C/C++.

Take a look at the most basic program for Arduino, it does nothing. There are two functions setup and loop. The setup function is called when the program begins and the loop function is called continually whilst the Arduino is running.

void setup() {

// put your setup code here, to run once:

}

void loop() {

// put your main code here, to run repeatedly:

}

Figure 1 Most basic program

Essential the Arduino executes the following C program: “setup(); while(1) loop();”

### Running the program

1. Connect your Arduino to your laptop/desktop
2. Enter the most basic program into the editor (should appear by default)
3. Click the compile button (the tick) to verify the program is legitimate C
4. Click the upload button

Note: if the upload fails try:

1. going to tools->board and make sure the UNO is selected
2. going to the tools->port and select the right board (you may have a few options, try them until you find which USB port your Arduino is connected to.