Integrated Development Environment "IDE" For Arduino

Authors: Mohamed FEZARI and Ali Al Dahoud
Al Zaytoona University, Amman, Jordan

Introduction to Arduino IDE

IDE stands for "Integrated Development Environment" :it is an official software introduced by Arduino.cc, that is mainly used for editing, compiling and uploading the code in the Arduino Device. Almost all Arduino modules are compatible with this software that is an open source and is readily available to install and start compiling the code on the go.
In this article, we will introduce the Software, how we can install it, and make it ready for developing applications using Arduino modules.

Arduino IDE Definition

1. Arduino IDE is an open source software that is mainly used for writing and compiling the code into the Arduino Module.
2. It is an official Arduino software, making code compilation too easy that even a common person with no prior technical knowledge can get their feet wet with the learning process.
3. It is easily available for operating systems like MAC, Windows, Linux and runs on the Java Platform that comes with inbuilt functions and commands that play a vital role for debugging, editing and compiling the code in the environment.
4. A range of Arduino modules available including Arduino Uno, Arduino Mega, Arduino Leonardo, Arduino Micro and many more.
5. Each of them contains a microcontroller on the board that is actually programmed and accepts the information in the form of code.
6. The main code, also known as a sketch, created on the IDE platform will ultimately generate a Hex File which is then transferred and uploaded in the controller on the board.
7. The IDE environment mainly contains two basic parts: Editor and Compiler where former is used for writing the required code and later is used for compiling and uploading the code into the given Arduino Module.
8. This environment supports both C and C++ languages.

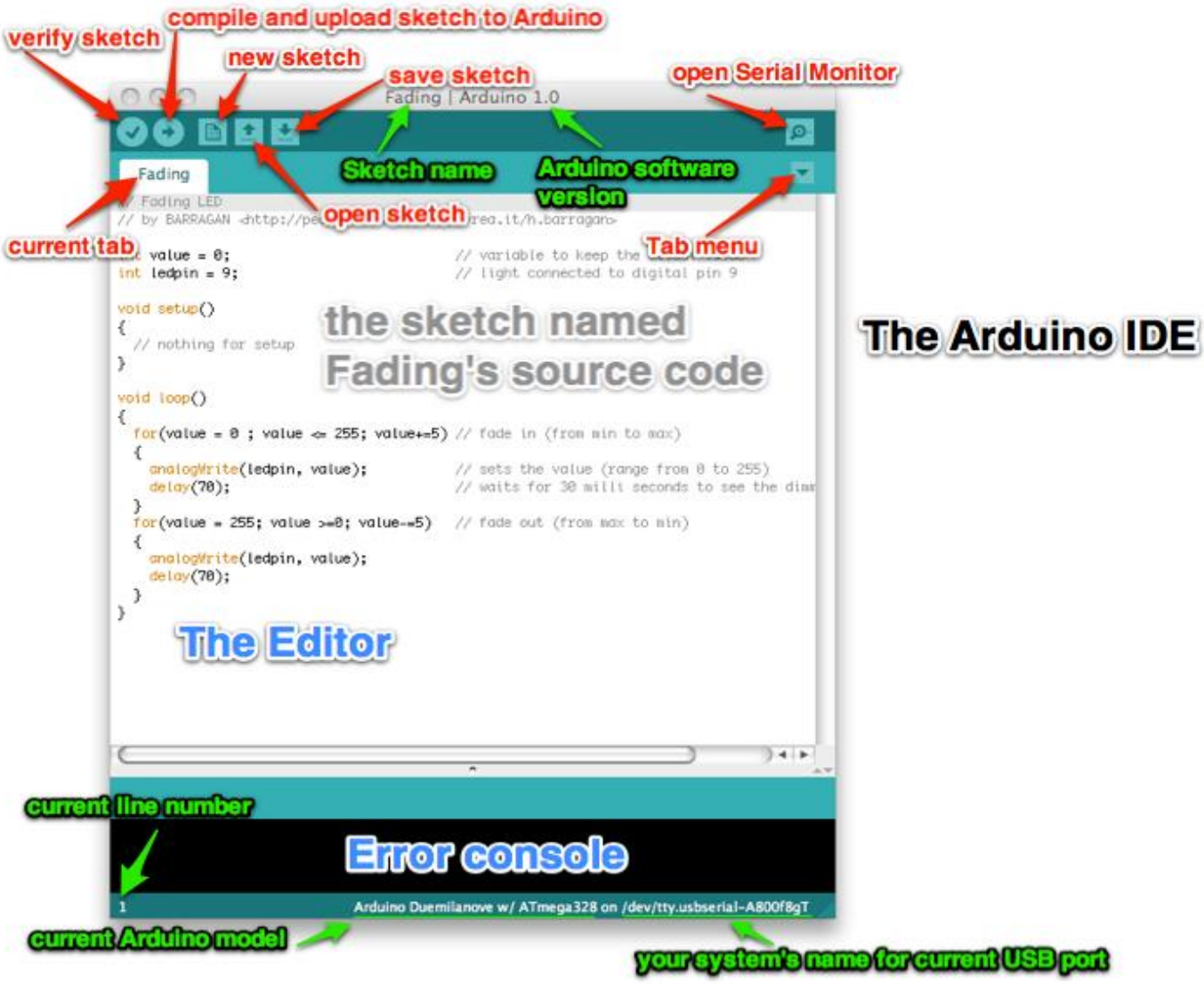
How to get Arduino IDE

we can download the Software from Arduino main website. As I said earlier, the software is available for common operating systems like Linux, Windows, and MACos,we select to download the correct software version that is easily compatible with our operating system.

Details on IDE: The IDE environment is mainly distributed into three sections

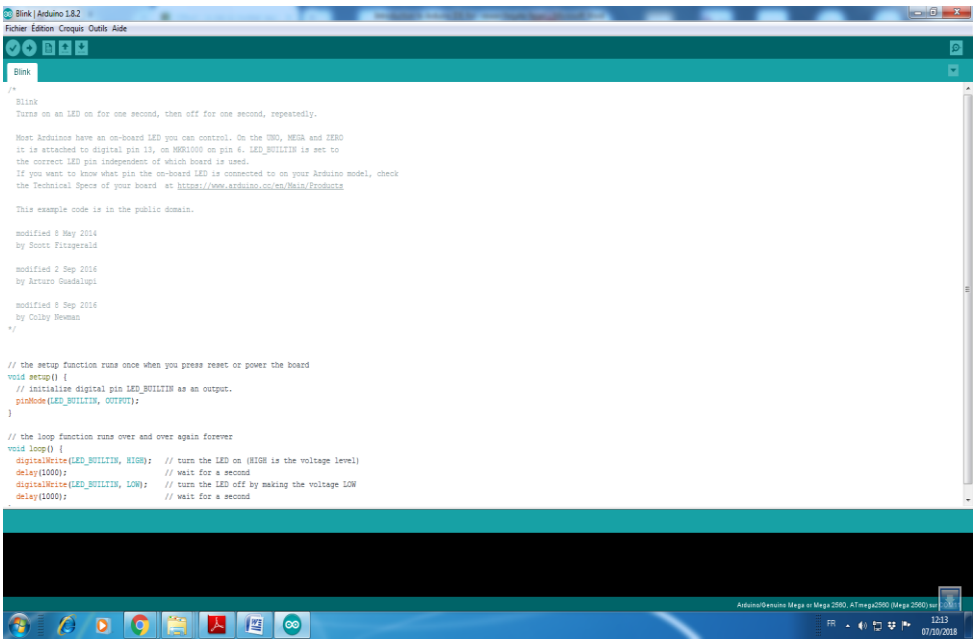
- **1. Menu Bar**
- **2. Text Editor**
- **3. Output Pane**

As we download and open the IDE software, it will appear like an image below.



The bar appearing on the top is called **Menu Bar** that comes with five different options as follow

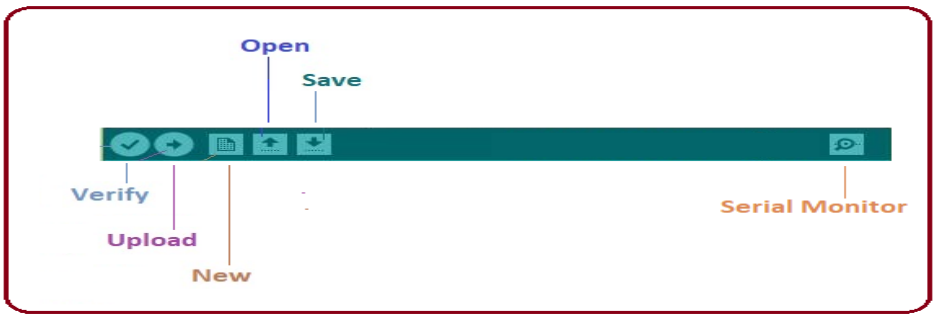
- **File** – You can open a new window for writing the code or open an existing one. Following table shows the number of further subdivisions the file option is categorized into.



And at the end of compilation, it will show you the hex file it has generated for the recent sketch that will send to the Arduino Board for the specific task you aim to achieve.

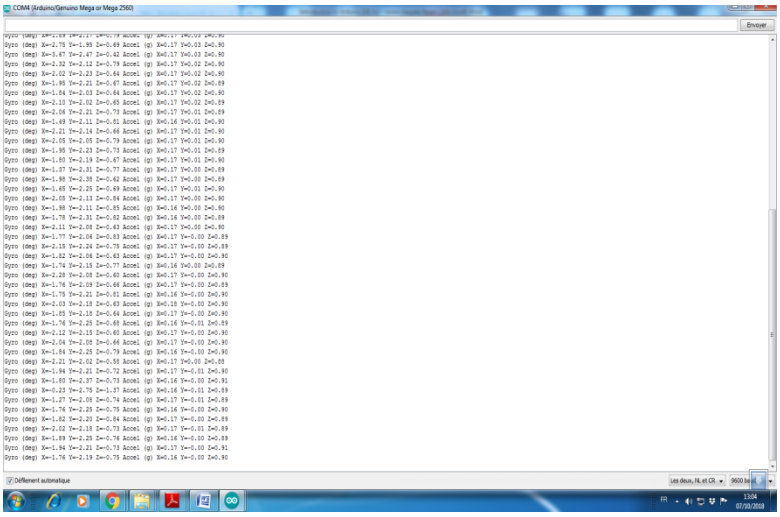
- **Edit** – Used for copying and pasting the code with further modification for font
- **Sketch** – For compiling and programming
- **Tools** – Mainly used for testing projects. The Programmer section in this panel is used for burning a bootloader to the new microcontroller.
- **Help** – In case you are feeling skeptical about software, complete help is available from getting started to troubleshooting.

The **Six Buttons** appearing under the Menu tab are connected with the running program as follow.



- The check mark appearing in the circular button is used to verify the code. Click this once you have writtenyour code.
- The arrow key will upload and transfer the required code to the Arduino board.
- The dotted paper is used for creating a new file.

- The upward arrow is reserved for opening an existing Arduino project.
- The downward arrow is used to save the current running code.
- The button appearing on the top right corner is a **Serial Monitor** – A separate pop-up window that acts as an independent terminal and plays a vital role for sending and receiving the Serial Data. You can also go to the Tools panel and select Serial Monitor pressing Ctrl+Shift+M all at once will open the Serial Monitor. The Serial Monitor will actually help to debug the written Sketches where you can get a hold of how your program is operating. Your Arduino Module should be connected to your computer by USB cable in order to activate the Serial Monitor.
- You need to select the baud rate of the Arduino Board you are using right now. For my Arduino Uno Baud Rate is 9600, as you right the following code and click the Serial Monitor, the output will show as the image below.



The main screen below show how to select a program from examples ie: Bink.cc

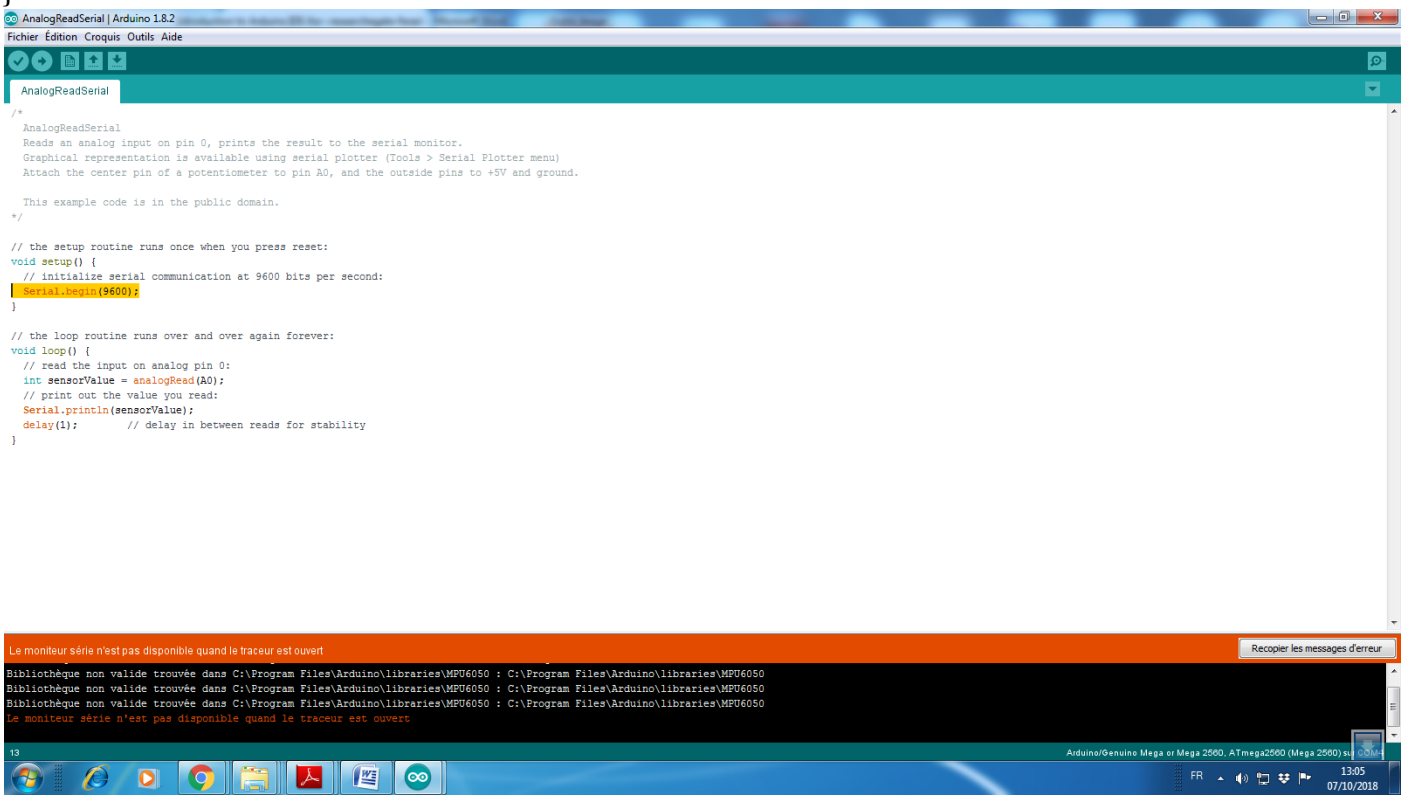
Example of Code in IDE Editor: blinking a LED on pin D13

```
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
```

```
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Example 2: read analog value from pin A0

```
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(1);        // delay in between reads for stability
}
```

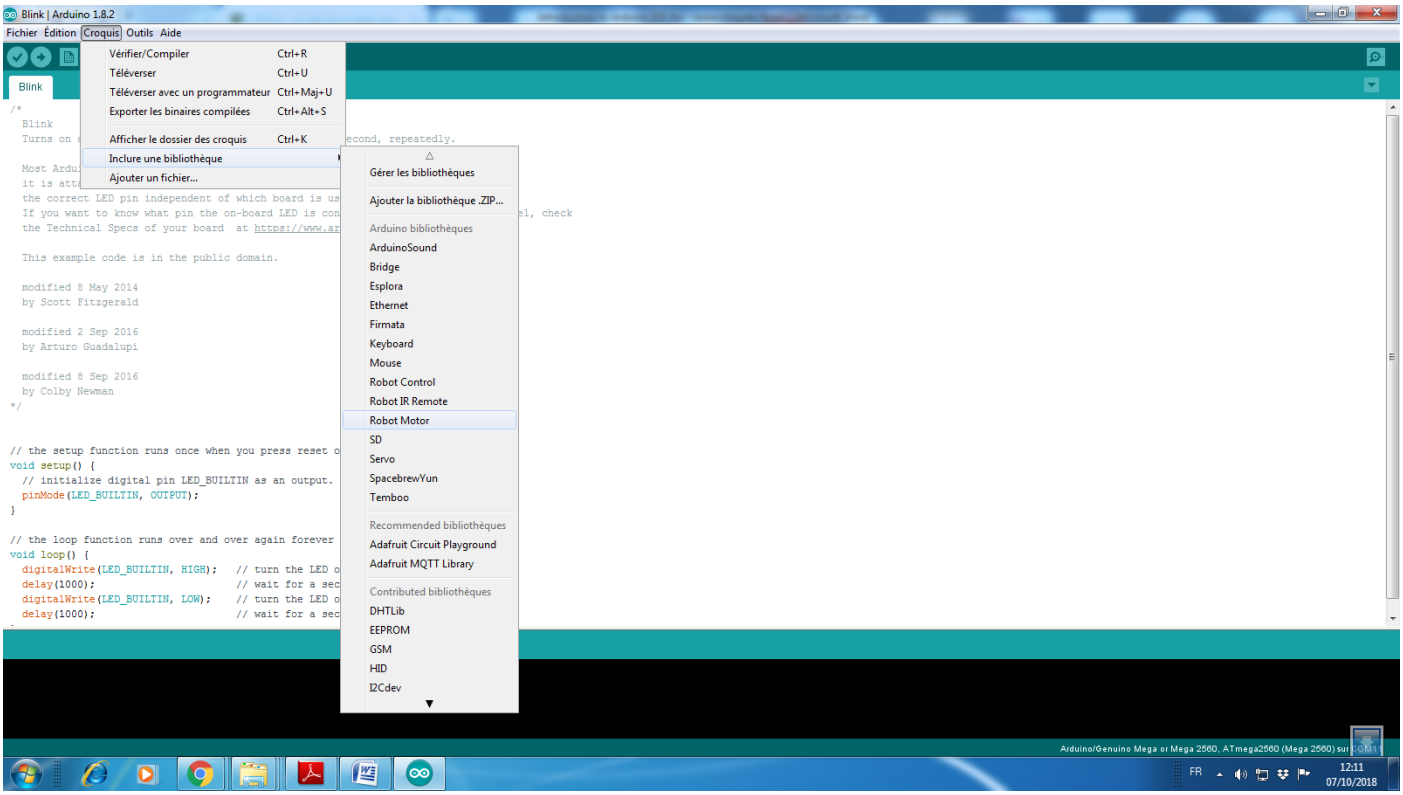


The bottom of the main screen is described as an Output Pane that mainly highlights the compilation status of the running code: the memory used by the code, and errors occurred in the program. You need to fix those errors before you intend to upload the hex file into your Arduino Module.

More or less, Arduino C language works similar to the regular C language used for any embedded system microcontroller, however, there are some dedicated libraries used for calling and executing specific functions on the board.

Libraries

Libraries are very useful for adding the extra functionality into the Arduino Module. There is a list of libraries you can add by clicking the Sketch button in the menu bar and going to Include Library.



As you click the Include Library and Add the respective library it will on the top of the sketch with a #include sign. Suppose, I Include the EEPROM library, Temperature sensors DHT11/22, LCD or I2C library it will appear on the text editor as

#include <EEPROM.h>.
#include <dht.h>
#include <I2Cdev.h>

Most of the libraries are preinstalled and come with the Arduino software. However, we can also download them from the external sources.

Making Pins As Input or Output

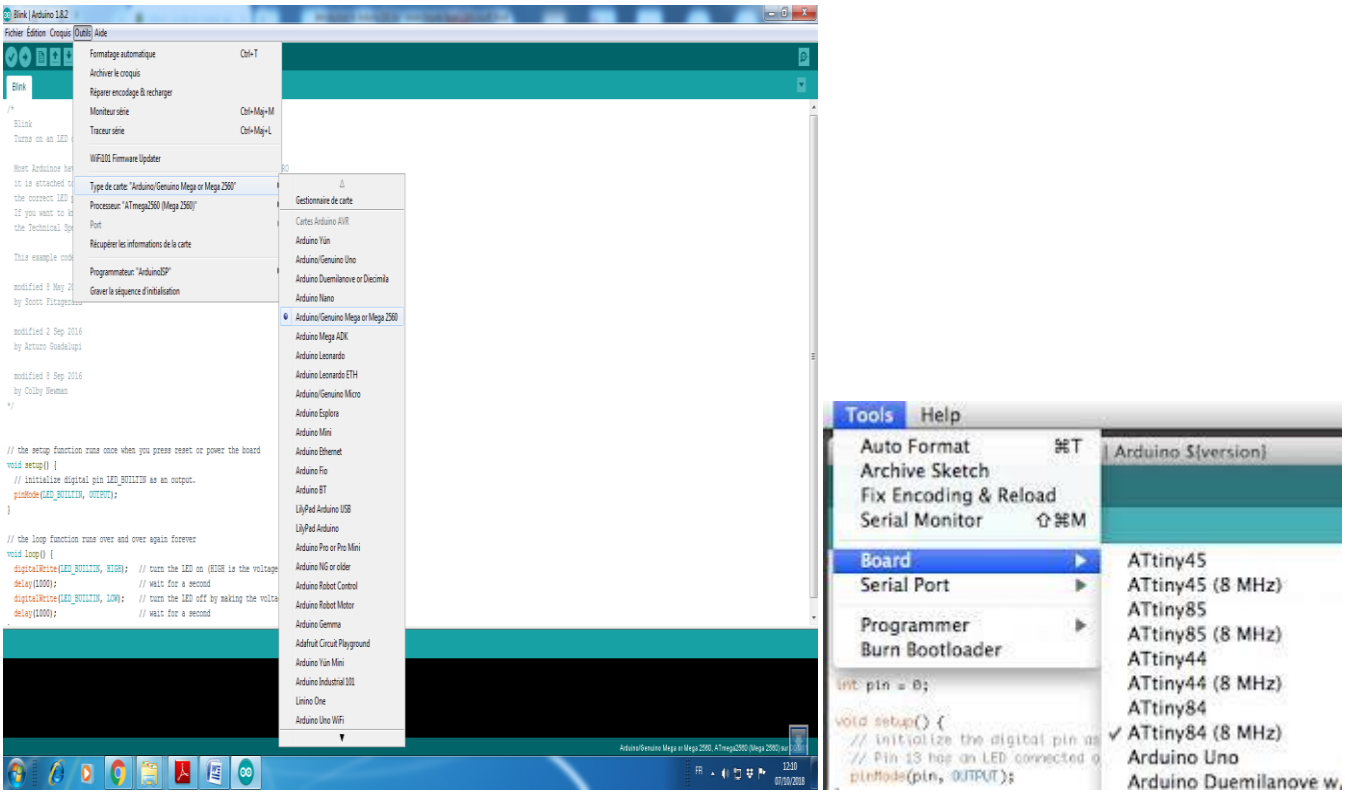
The digitalRead and digitalWrite commands are used for addressing and making the Arduino pins as an input and output respectively.
These commands are text sensitive i.e. you need to write them down the exact way they are given like digitalWrite starting with small “d” and write with capital “W”. Writing it down with Digitalwrite or digitalwritewon’t be calling or addressing any function.

Example : if we want to use Pin D13 as output , the code will be; pinMode(13, OUTPUT); followed by digitalWrite(13,HIGH);

If we want to use Pin D13 as input, the code will be : pinMode(13, INPUT);followed by x=digitalRead(13);

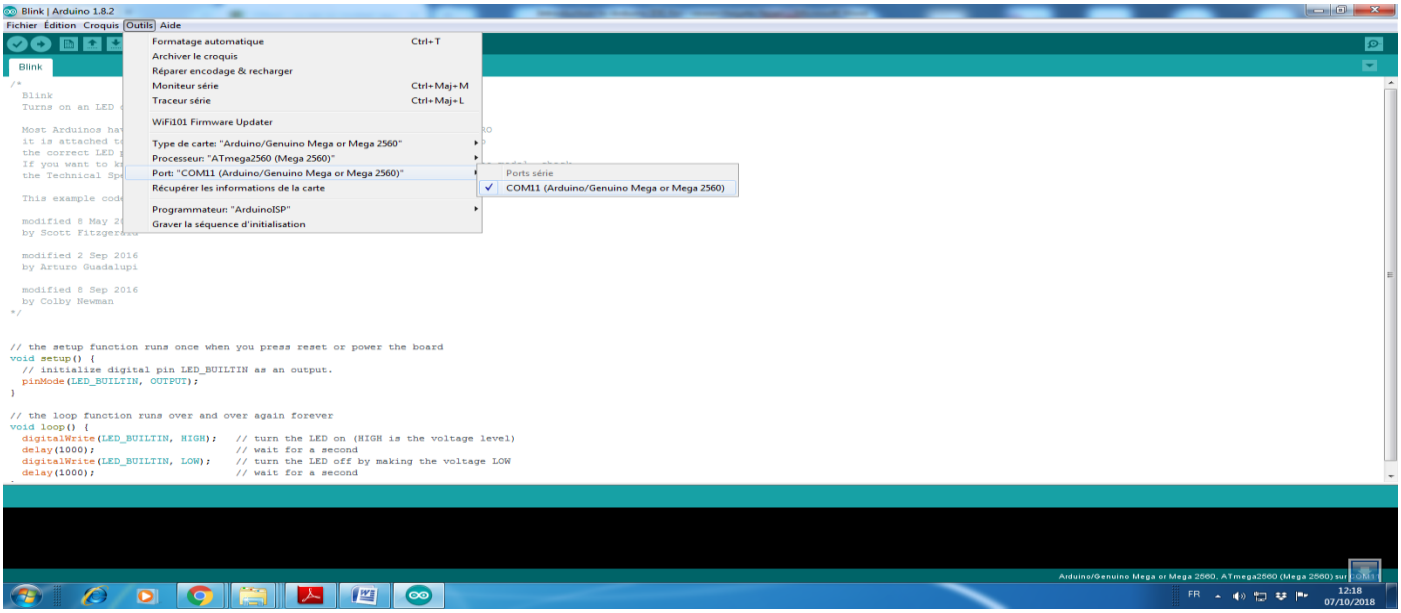
Selecting Board of Arduino

In order to upload the sketch, we need to select the relevant board we are using and the ports for that operating system. As we click the Tools on the Menu, it will open like the figure below.



Just we go to the “Board” section and select the board wewould like to work on. Similarly, COM1, COM2, COM4, COM5, COM7 or higher are reserved for the serial and USB board. we can look for the USB serial device in the ports section of the Windows Device Manager.

Folloinwg figure shows the COM4 that wehave used for my project, indicating the Arduino Uno with COM4 port at the right bottom corner of the screen.



- After correct selection of both Board and Serial Port, click the verify and then upload button appearing in the upper left corner of the six button section or you can go to the Sketch section and press verify/compile and then upload.
- The sketch is written in the text editor and is then saved with the file extension .ino.

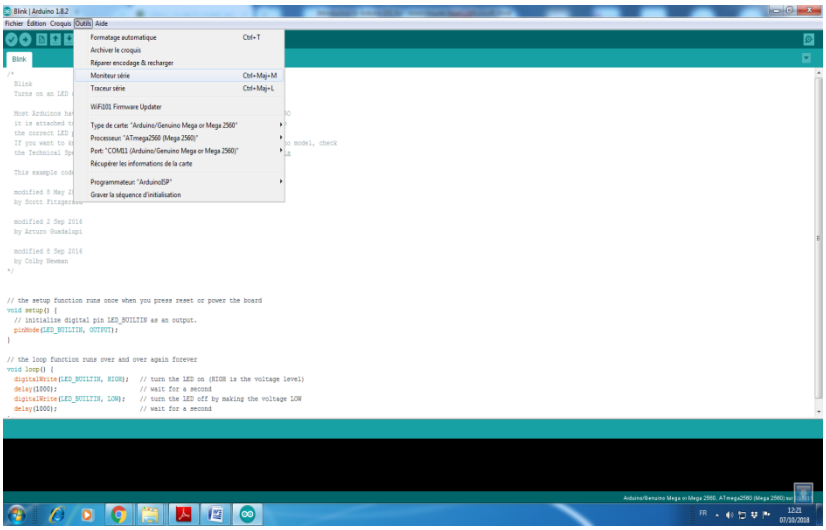
It is important to note that the recent Arduino Modules will reset automatically as you compile and press the upload button the IDE software, however, older version may require the physical reset on the board.

- As we upload the code, TX and RX LEDs will blink on the board, indicating the desired program is running successfully.

Note: The port selection criteria mentioned above is dedicated for Windows operating system only, you can check this Guide if you are using MAC or Linux.

- The amazing thing about this software is that no prior arrangement or bulk of mess is required to install this software, you will be writing your first program within 2 minutes after the installation of the IDE environment.

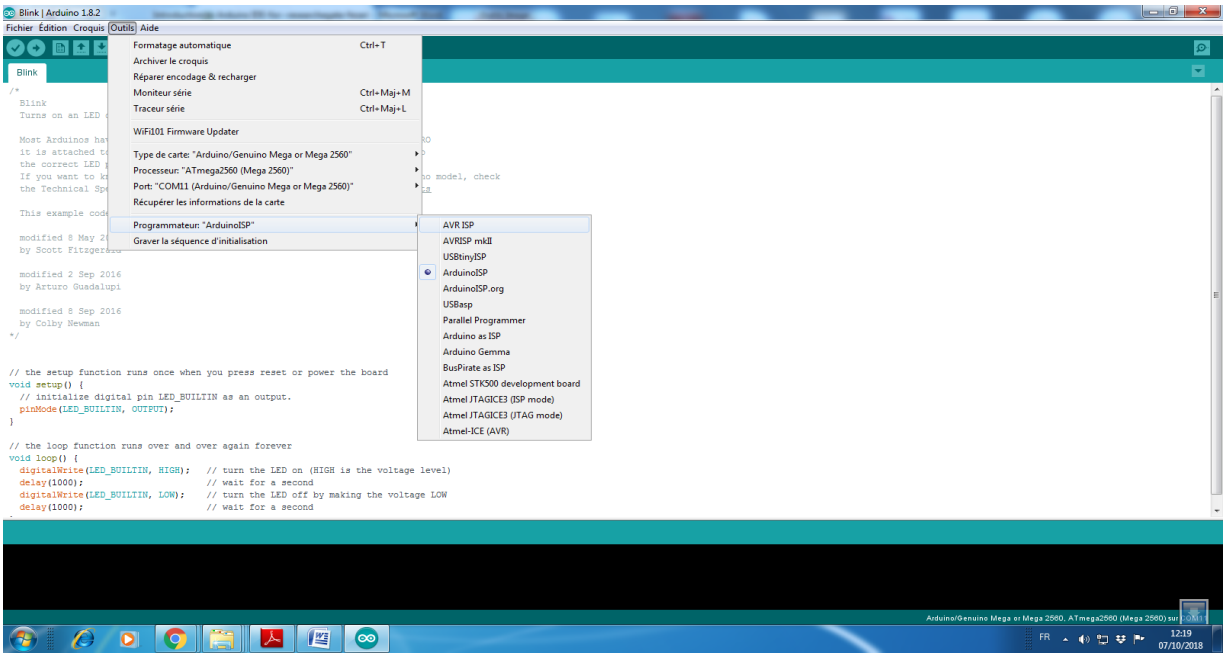
Using Serial Monitor



Example of test serial monitor on program test2.cc

Bootloader

As we go to the Tools section, we will find a bootloader at the end. It is very helpful to burn the code directly into the controller, setting us free from buying the external burner to burn the required code.



When we buy the new Arduino Module, the bootloader is already installed inside the controller. However, if we intend to buy a controller and put in the Arduino module, we need to burn the bootloader again inside the controller by going to the Tools section and selecting the burn bootloader.

the Porgram Structure

Declarations

Variables

Whenever you're using Arduino, you need to declare global variables and instances to be used later on. In a nutshell, a variable allows you to name and store a value to be used in the future. For example, you would store data acquired from a sensor in order to use it later. To declare a variable you simply define its type, name and initial value.

It's worth mentioning that declaring global variables isn't an absolute necessity. However, it's advisable that you declare your variables to make it easy to utilize your values further down the line.

Instances

In software programming, a **class** is a collection of functions and variables that are kept together in one place. Each class has a special function known as a **constructor**, which is used to create an **instance** of the class. In order to use the functions of the class, we need to declare an instance for it.

Setup()

Every Arduino sketch must have a setup function. This function defines the initial state of the Arduino upon boot and runs only once.

Here we'll define the following:

1. Pin functionality using the pinMode function
2. Initial state of pins
3. Initialize classes

4. Initialize variables
5. Code logic

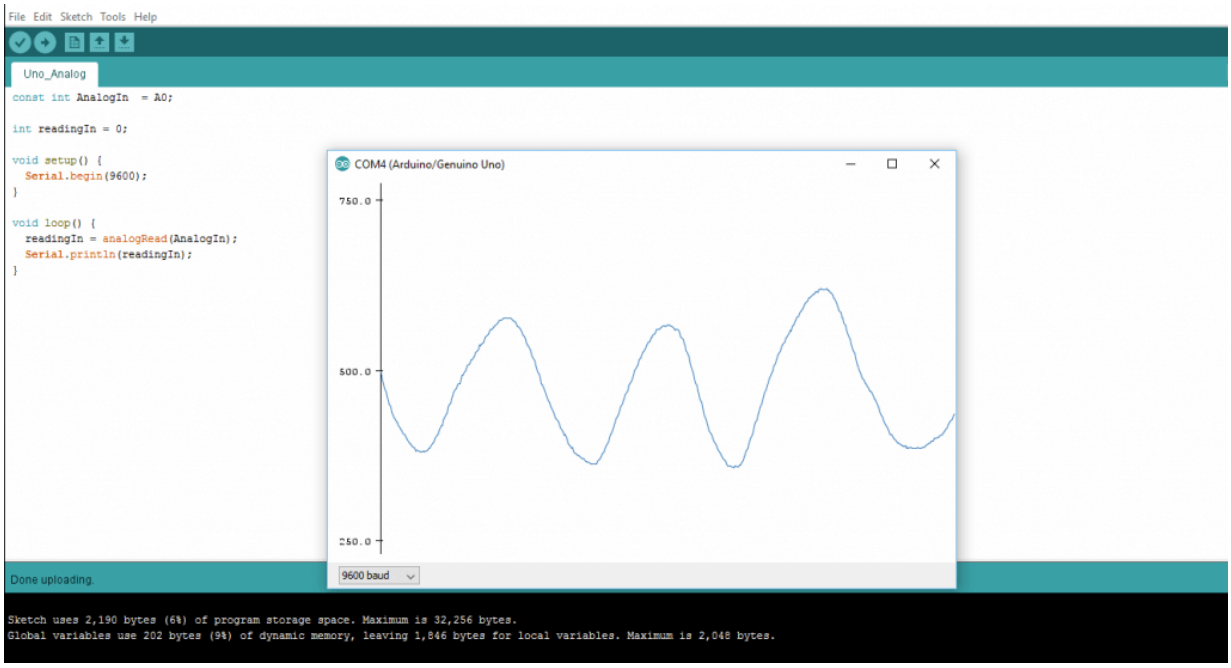
Loop()

The loop function is also a must for every Arduino sketch and executes once setup() is complete. It is the main function and as its name hints, it runs in a loop over and over again. The loop describes the main logic of your circuit.

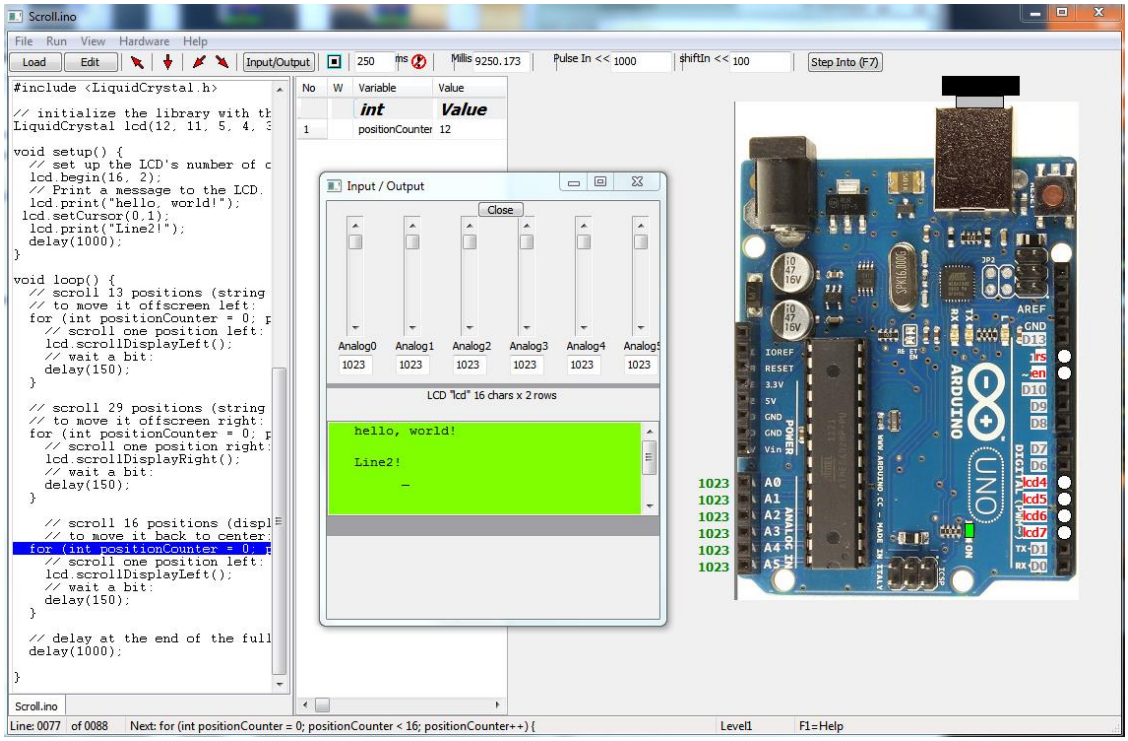
.

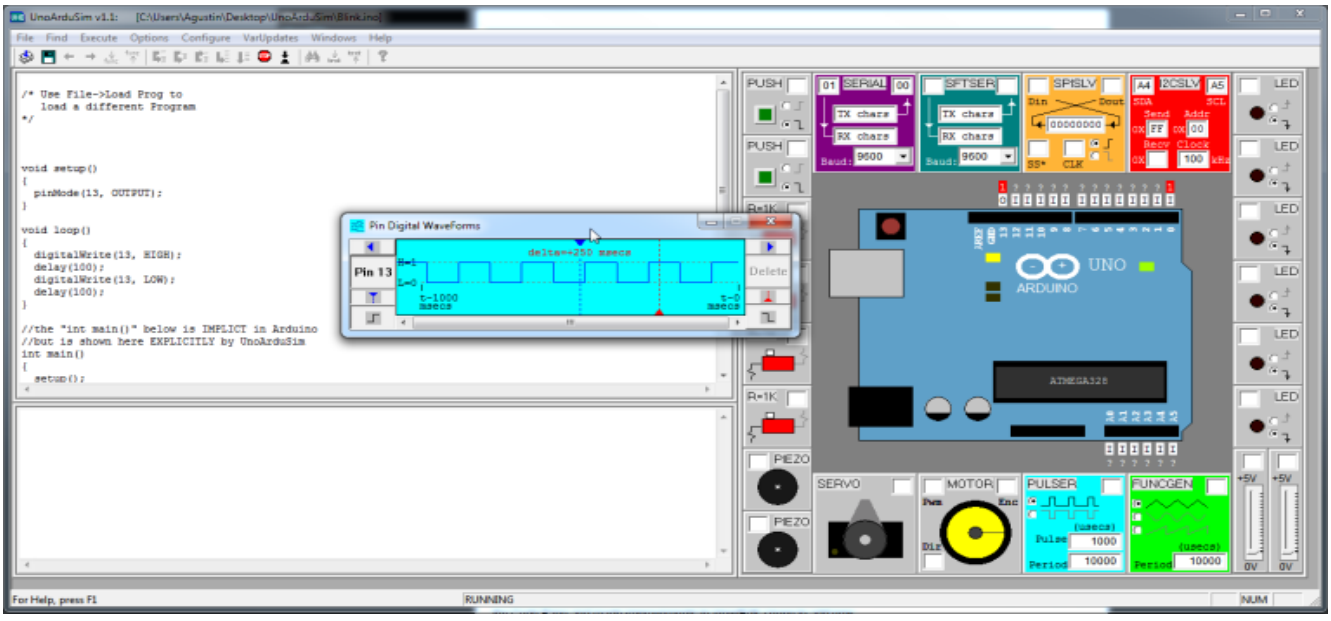
Serial Ploter

Arduino **serial plotter** is another component of the Arduino IDE, which allows you to generate a real-time graph of your serial data. The serial plotter makes it much easier for you to analyze your data through a visual display. You're able to create graphs, negative value graphs, and conduct waveform analysis



Simulators for ARDUINO: in the next article we will see some details on powerful simulators for Arduino, example: Autodesk Eagle (recommended), Proteus , Autodesk Circuits, Virtronics Simulator for Arduino , Electronify , Fritzing , VBB4Arduino – Virtual Breadboard for Arduino,





References:

https://www.arduino.cc/en/Main/Software
https://www.arduino.cc/en/Guide/HomePage
http://www.audon.co.uk/arduino.html
https://www.circuito.io/blog/arduino-code/
https://www.makerspaces.com/arduino-uno-tutorial-beginners/
https://windowsreport.com/arduino-simulators/
http://virtronics.com.au/Simulator-for-Arduino.html