MAKERbuino coding #1 - first steps

Preparing your coding rig

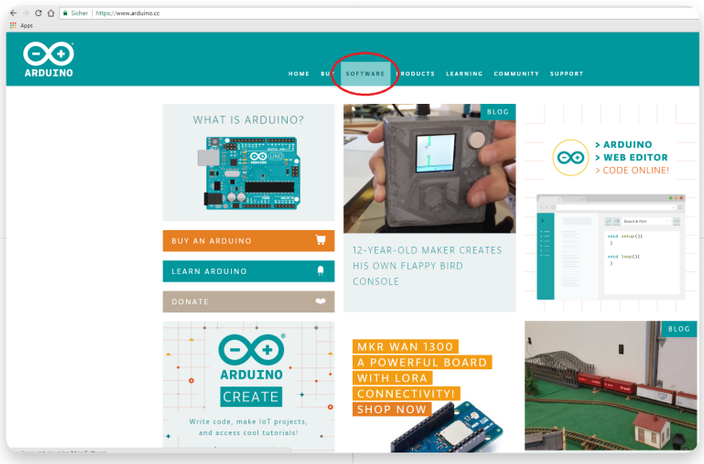
Downloading and installation of Arduino IDE

Made by Miguel Drager (@Bl4ckM4ch1n3)

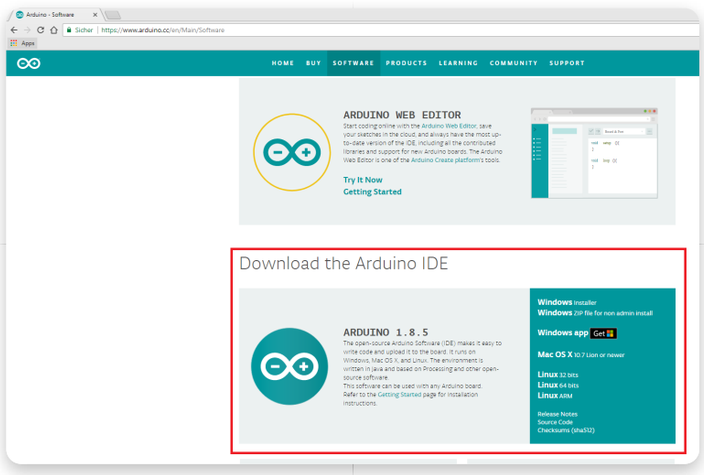
First things first, you'll need to download some on your computer.

You're looking for something called **Arduino IDE** (Integrated Development Environment).

Just go to arduino.cc and select "**Software**" from the menu at the top of the page.

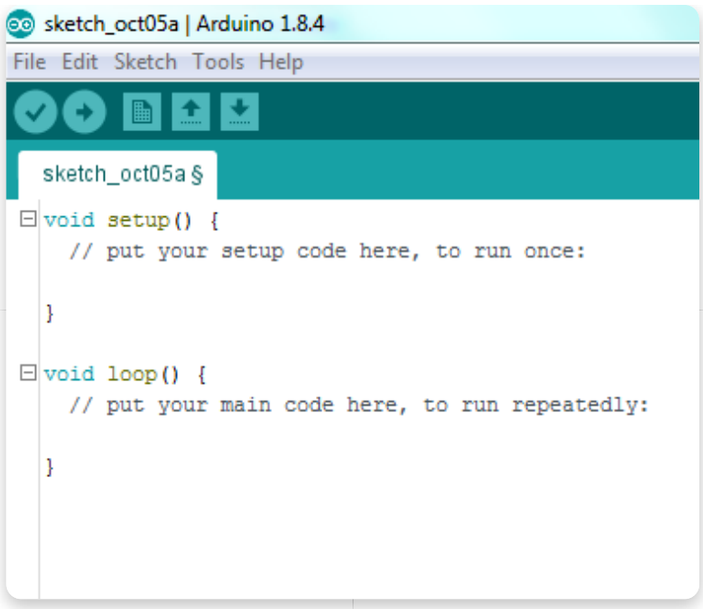


Now, you'll have to select the **software version** depending on your rig's operating system.



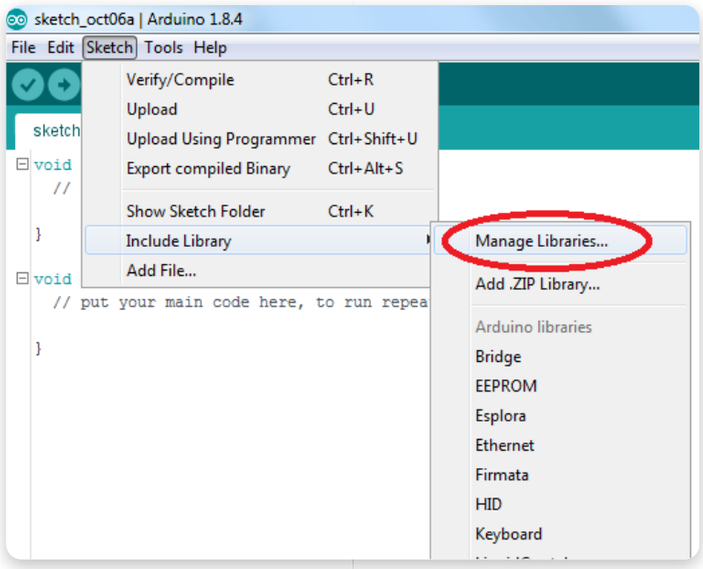
After the download has finished, install the IDE to a location of your choice.

Start the Arduino IDE and you'll see something similar to this:



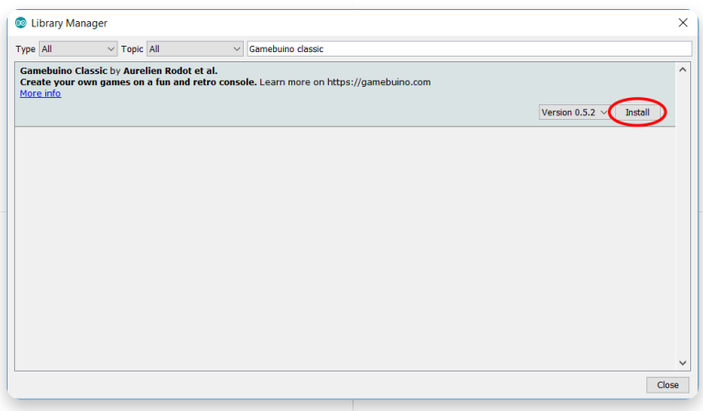
Now you're almost ready to develop something for the MAKERbuino.

The last thing missing is the Gamebuino library. To get the library just select **Sketch -> Include Library -> Manage Libraries...**



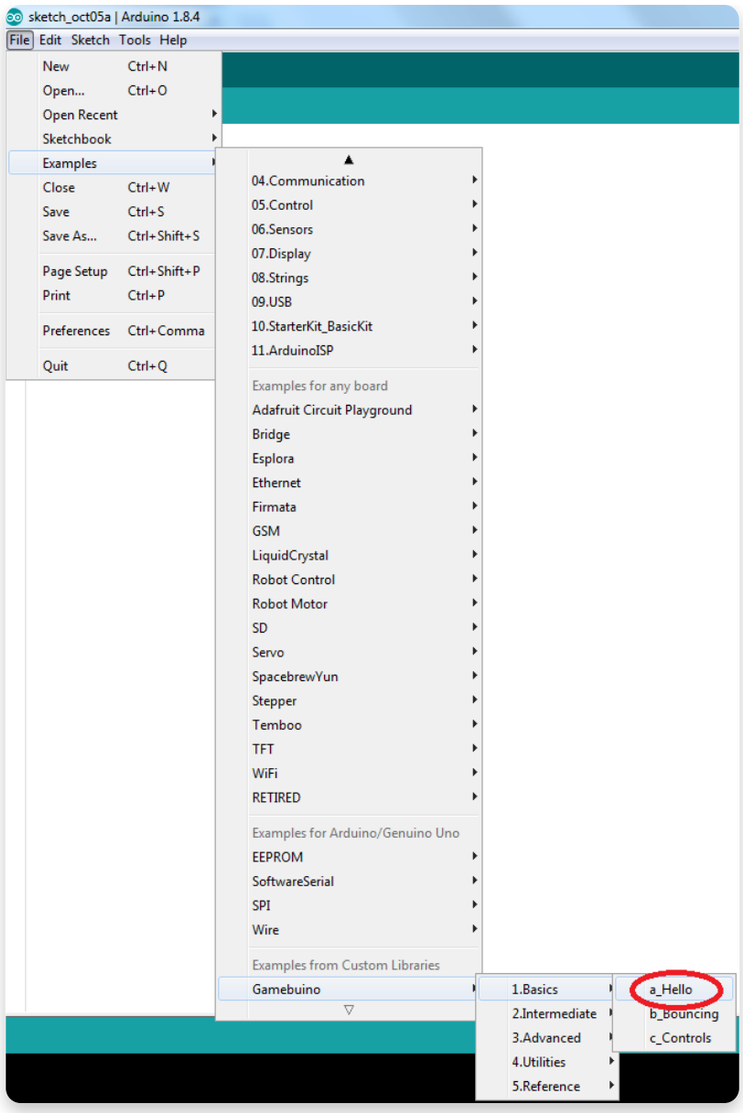
Type "**Gamebuino classic**" into the search bar, select the Gamebuino Classic library and hit install.

You need to restart the Arduino IDE after the installation has finished. Otherwise, the library won't show up.



To check whether the library was installed properly, you'll have to select **File -> Examples -> Gamebuino -> 1.Basics -> a_Hello.**

You may need to scroll down a little bit since the custom library examples are at the bottom of the list.



If you've chosen the right thing, you should now see the following source code:

a_Hello | Arduino 1.8.4

File Edit Sketch Tools Help

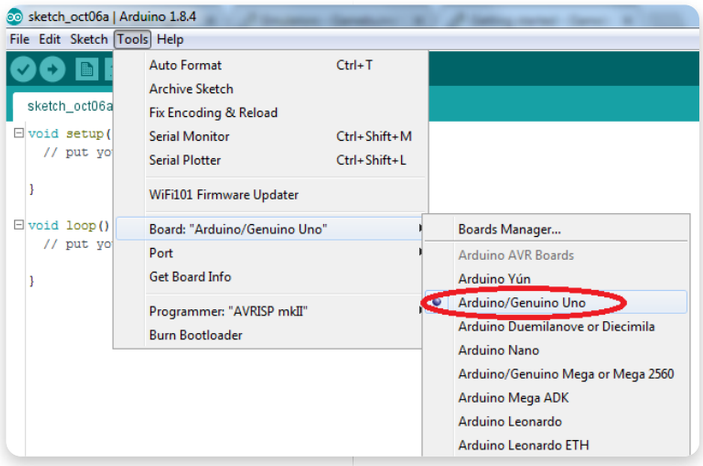
a_Hello

```
//imports the SPI library (needed to communicate with Gamebuino's screen)
#include <SPI.h>
//imports the Gamebuino library
#include <Gamebuino.h>
//creates a Gamebuino object named gb
Gamebuino gb;

// the setup routine runs once when Gamebuino starts up
void setup(){
  // initialize the Gamebuino object
  gb.begin();
  //display the main menu:
  gb.titleScreen(F("My first game"));
  gb.popup(F("Let's go!"), 100);
}

// the loop routine runs over and over again forever
void loop(){
  //updates the gamebuino (the display, the sound, the auto backlight... everything)
  //returns true when it's time to render a new frame (20 times/second)
  if(gb.update()){
    //prints Hello World! on the screen
    gb.display.println(F("Hello World!"));
    //declare a variable named count of type integer :
    int count;
    //get the number of frames rendered and assign it to the "count" variable
    count = gb.frameCount;
    //prints the variable "count"
    gb.display.println(count);
  }
}
```

The last thing to do is to choose the hardware we're using so that the Arduino IDE knows how to compile the code correctly.

This is done by selecting **Tools -> Board -> Arduino/Genuino Uno.** We need this Board because the MAKERbuino uses the same Microprocessor as the Arduino/Genuino Uno does.

We need to select this exact board because the MAKERbuino uses the same Microprocessor as the Arduino/Genuino Uno does.



And that's it, you're ready to roll! **Let's try out whether the "a_Hello" example works on the MAKERbuino.**

We can test this program in three different ways:

- Upload it to the MAKERbuino via the included RS232-to-USB converter board
- Export and copy the HEX-file to your MAKERbuino's SD-Card
- Use a MAKERbuino emulator on your computer

Uploading and running games

Uploading the program via the included RS232-to-USB converter

This is the easiest way to get your game up and running on the MAKERbuino.

You just need to connect the RS232-to-USB converter (the tiny red board included in the package) to your MAKERbuino, connect the converter to your PC via USB cable and hit the arrow symbol in the upper left corner of the Arduino IDE.

You can upload your code by selecting **Sketch -> Upload** as well. But let's do this step by step.

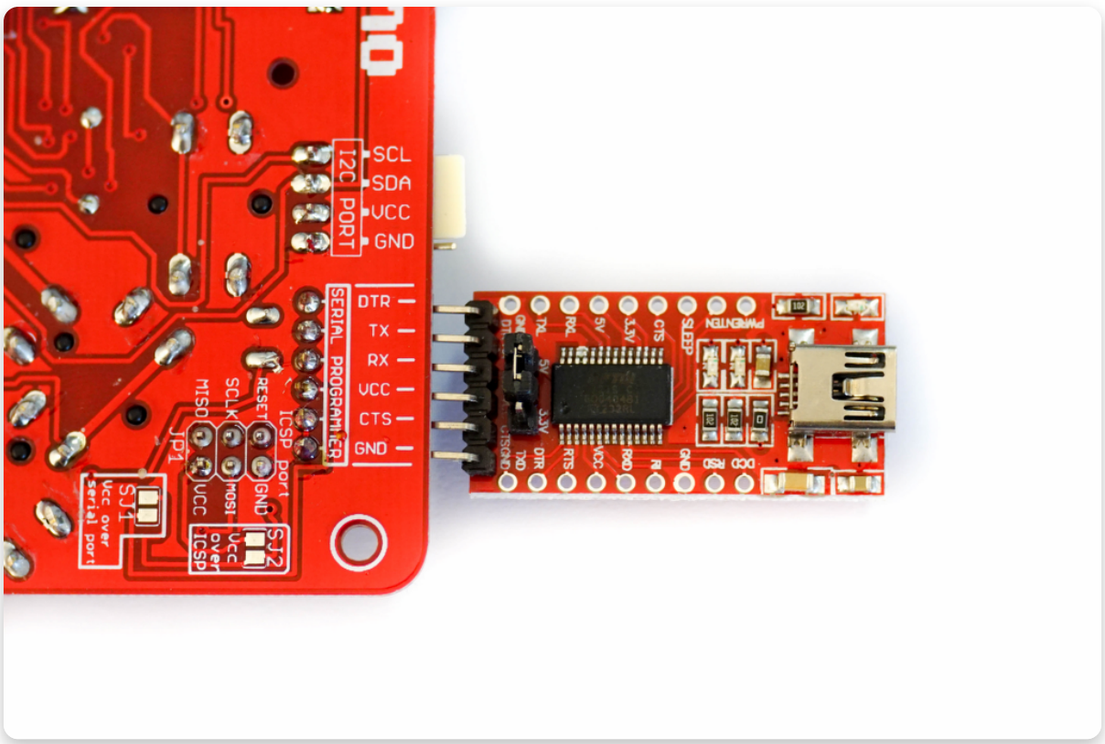
Let's start by connecting the RS232-to-USB converter to your MAKERbuino.

The RS232 or UART port is located at the top left corner of your MAKERbuino. It is a female header with 6 pins, labeled as Serial Programmer on the backside of the PCB.

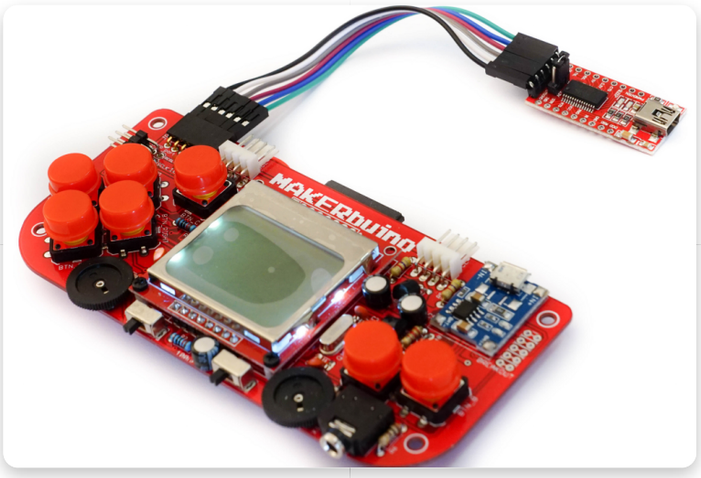
The description of each pin is printed on the backside of the PCB too.

If you compare the pin descriptions of the MAKERbuino and the RS232-to-USB converter you'll notice that they're in the same order except for one pair of pins:

RX and TX.



The reason is quite simple: RX means “receive” and TX means “transmit”.



So, if the converter is transmitting data, the MAKERbuino has to receive it. If the MAKERbuino is transmitting data, the converter has to receive it.

That’s why these pins are crossed and why you need to connect RX->TX and TX->RX.

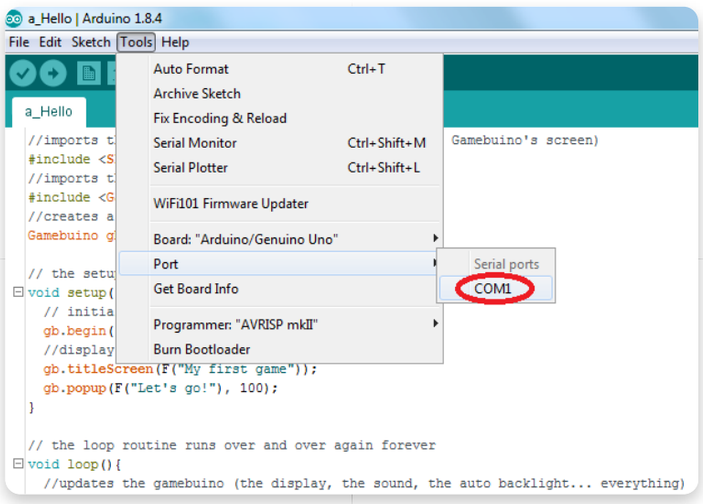
Turn your MAKERbuino ON after you’ve connected the converter!

The next step is to connect the converter to your PC.

This is quite easy as you only need a common mini USB cable to establish the connection between your PC and the converter.

Please Note: It may take a while until the converter is ready to be used if you've connected the converter for the first time.

Your Operating system needs to install the corresponding driver first.



A new COM-Port should appear once the installation is done. Now you need to tell the Arduino IDE which COM-Port should be used.

You can achieve this by selecting **Tools -> Port -> [NEW COM-Port].**

Now you're good to go! Just hit the arrow in the upper left corner of the IDE or select **Sketch -> Upload.**

A couple of seconds later, you should see something like this on your MAKERbuino:

Congratulations, you've just uploaded a program to your MAKERbuino!

If you want to go back to the SD card menu, power OFF your MAKERbuino, hold the C-button, turn the buino back on, release the C button and wait for 20 seconds.

Generating and copying the HEX-file to your MAKERbuino's SD card

Another way of transferring a program from your computer to the MAKERbuino is to copy the compiled HEX-File onto the device's SD-Card. You'll need a card reader for your PC/Laptop which is capable of handling (micro)SD-cards.

For this, you'll need a card reader for your PC/Laptop which is capable of handling micro or regular SD cards.

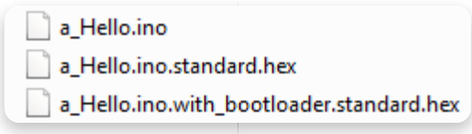
If you're exporting the Arduino sketch for the first time, the Arduino IDE will ask you for a location where you'd like to save your Sketch.

This is happening because the example file is read-only. Chose a location and save the Sketch.

The Arduino IDE will create a new folder with the name of the Sketch with the Sketch in it.

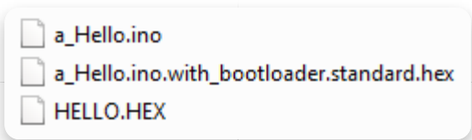
After saving the Sketch, just select **Sketch -> Export compiled Binary** and wait until the Arduino IDE has finished the compilation.

Open the folder you've saved your Arduino sketch to. You'll now see three new files:



You can delete the file called **a_Hello.ino.with_bootloader.standard.hex.**

Rename the file called a_Hello.ino.standard.hex. Use an 8:3 format.
This means 8 letters max for the name and 3 letters for the extension (.HEX). Every letter needs to be a capital letter too!
You can go with "HELLO.HEX", for example.



Important

Never use LOADER.HEX or SETTINGS.HEX as your program's name! These are very important MAKERbuino's system's files you shouldn't overwrite! If you replace the LOADER.HEX with a program/game you won't be able to return to the game selection menu (although you shouldn't panic, it's fixable).

Turn off your MAKERbuino and remove the SD card.
Insert the SD Card into your card reader. You should see a lot of files.

Copy your newly created and freshly renamed file (mine is called "**HELLO.HEX**") onto the SD card.

Remove the SD card from your card reader and insert it in your MAKERbuino again. Turn your MAKERbuino ON.

You'll find your program at the end of the list. Hit the A-button two times and the MAKERbuino will start loading your HEX-file.

You should see a similar screen after a few of seconds:

Splendid, you've just learned how to compile and copy a program/game to your MAKERbuino!

If you want to return to the SD card loader, just use the C-button trick mentioned at the end of **Uploading via the included RS232-to-USB converter section.**

Using an emulator

Emulators are a great way to test your program/game if you've just ordered your MAKERbuino but didn't receive it yet.

There are a couple of Gamebuino-compatible emulators available:

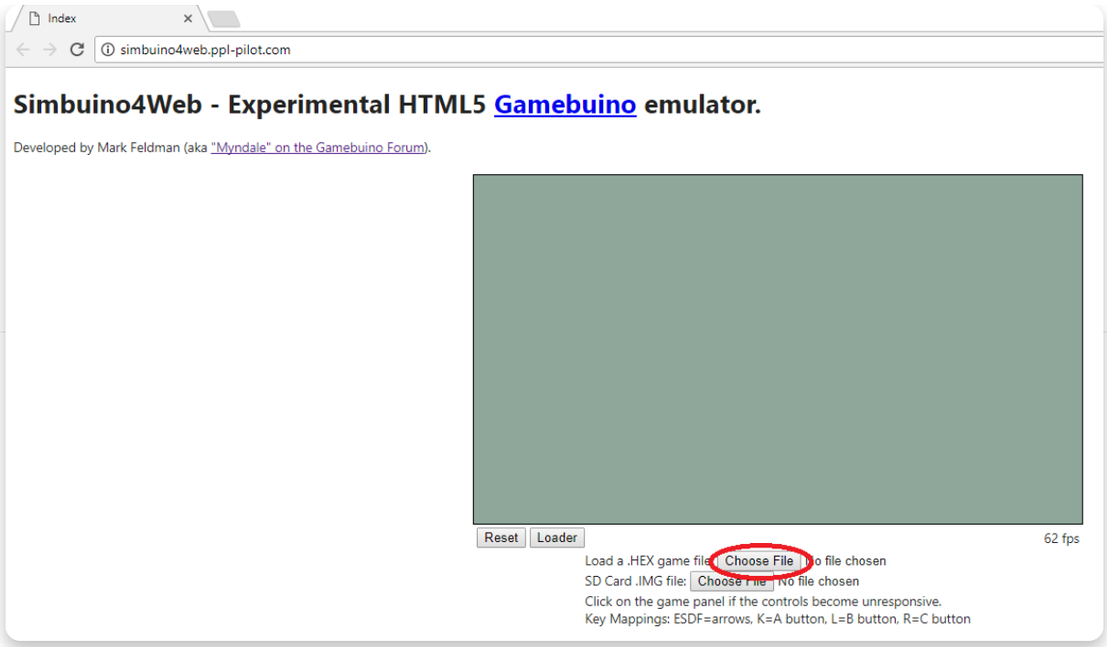
- Gbsim (Debian/Ubuntu and Windows)
- RetroMicro (Android)
- Simbuino (Windows)
- Simbuino4Web (HTML5)

We'll use Simbuino4Web as it is really easy to use and doesn't require any installation. You just need to upload your HEX-File and you can instantly test your game.

If you've already exported your compiled Binary a.k.a. HEX file (see **Copying the HEX-file to your SD card**), you're almost ready to go.

Now go to http://simbuino4web.ppl-pilot.com/. There are two different options. You can either upload a whole SD-card image file or just a single HEX-file. We'll be focusing on the single HEX-file.

Hit the "chose file" button next to "**Load a .HEX game file**" and locate your "**HELLO.HEX**" file. The emulation will start right after the upload has finished.



That was easy, wasn't it? An emulator comes in really handy if you want to test everything really quick!

And that's it! You've learned a lot today! You now know how to setup the Arduino IDE. And you know how to test your program/game in three different ways.