Processing + Arduino

Introduction

Processing is an open source computer programming language and integrated development environment (IDE) built for the electronic arts, new media art, and visual design communities with the purpose of teaching the fundamentals of computer programming in a visual context, and to serve as the foundation for electronic sketchbooks.

The project was initiated in 2001 by Casey Reas and Benjamin Fry, both formerly of the Aesthetics and Computation Group at the MIT Media Lab. In 2012, they started the Processing Foundation along with Daniel Shiffman, who joined as a third project lead.

One of the aims of Processing is to allow non-programmers to start computer programming aided by visual feedback. The Processing language builds on the Java language, but uses a simplified syntax and a graphics user interface.

[Wikipedia]

Basics

Setup()

It is the part of program we place all the things necessary to get started. E.g. the size of the window we will be playing with, the styles of the graphics we want to see etc. Its syntax is as follow:

```
void setup()

{

        //Place your Code here!

}
```

Draw()

The part of program that needs to be repeated again and again is usually placed here. It's like the loop() of Arduino code. You place the necessary things and changes that updates with iteration. It's also like the setup function, except it's called again and again.

```
void draw()

{

        //Place the Iterative Code here!

}
```

//Some useful Functions to be Remembered!

1. size(x,y): It tells us the size of console we are playing in, in terms of 'x' and 'y' coordinates.
2. line(x1,y1,x2,y2): The line between two coordinates.
3. translate(x,y): The object last treated is moved to new place for given values of 'x' and 'y'.

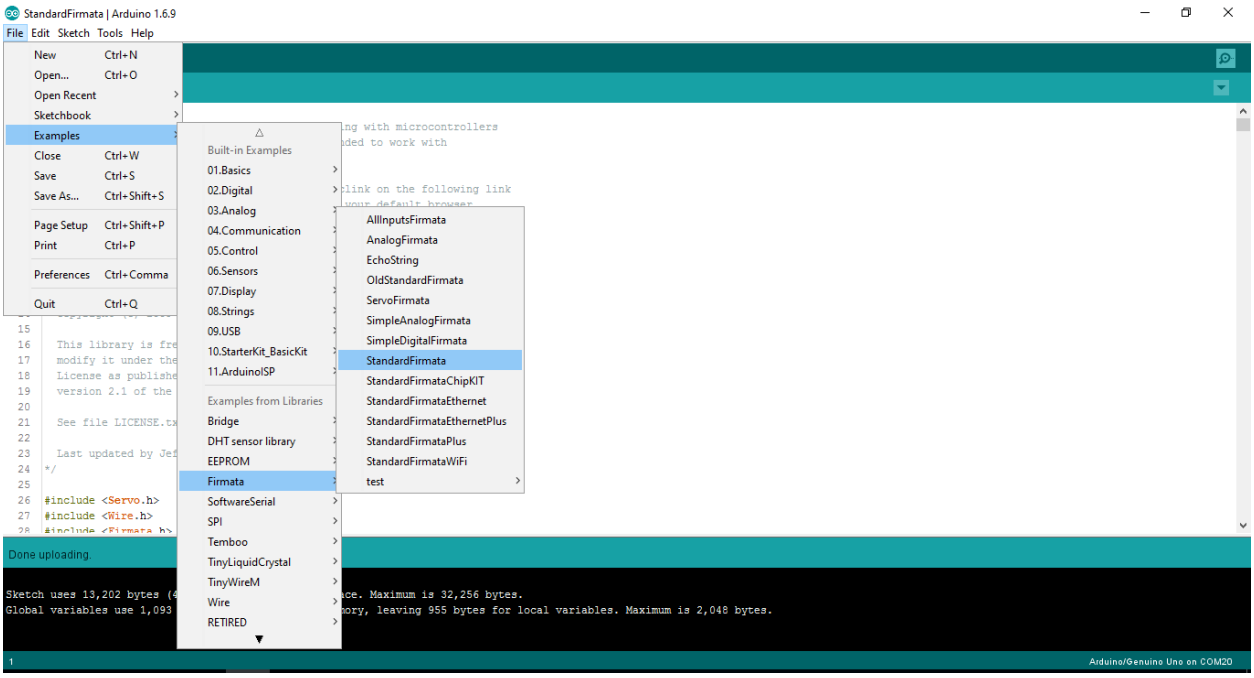
4. import processing.serial.*; // reference the serial library
5. import cc.arduino.*; // reference the Arduino library
6. Arduino arduino; // create a variable Arduino of the Arduino data type
7. println(Serial.list()); // List all the available serial ports:
8. arduino = new Arduino(this, Arduino.list()[0], 57600);

You can find out more at http://www.processing.org/reference for particular library function.

Integration with Arduino

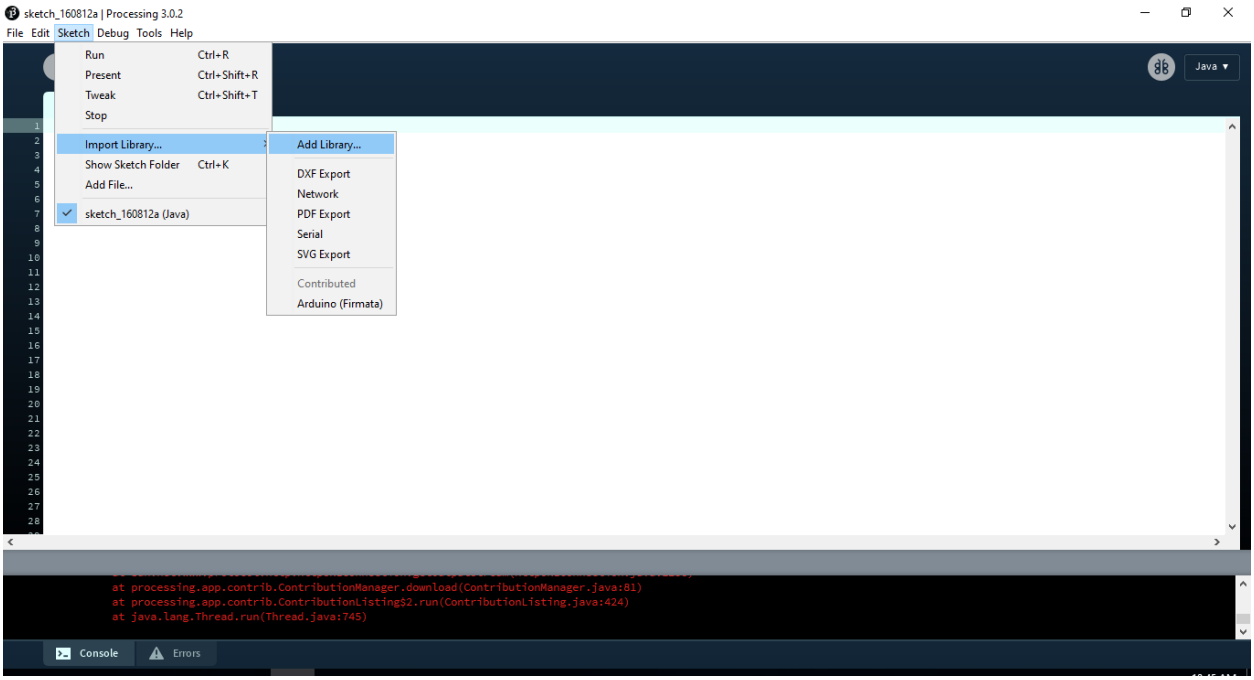
To integrate Processing with Arduino, we need a special piece of code that tells/instructs computer to communicate between processing IDE and Arduino on serial port. We call it a protocol, and in this case it is Firmata (See http://www.firmata.org for more information). In our Arduino IDE, there is Firmata pre-installed, so we don't need something extra to do. Only thing that is necessary is to upload Standard Firmata to Arduino so that it can be controlled directly by Processing IDE. For this, open Arduino and go to:

File->Examples->Firmata->StandardFirmata

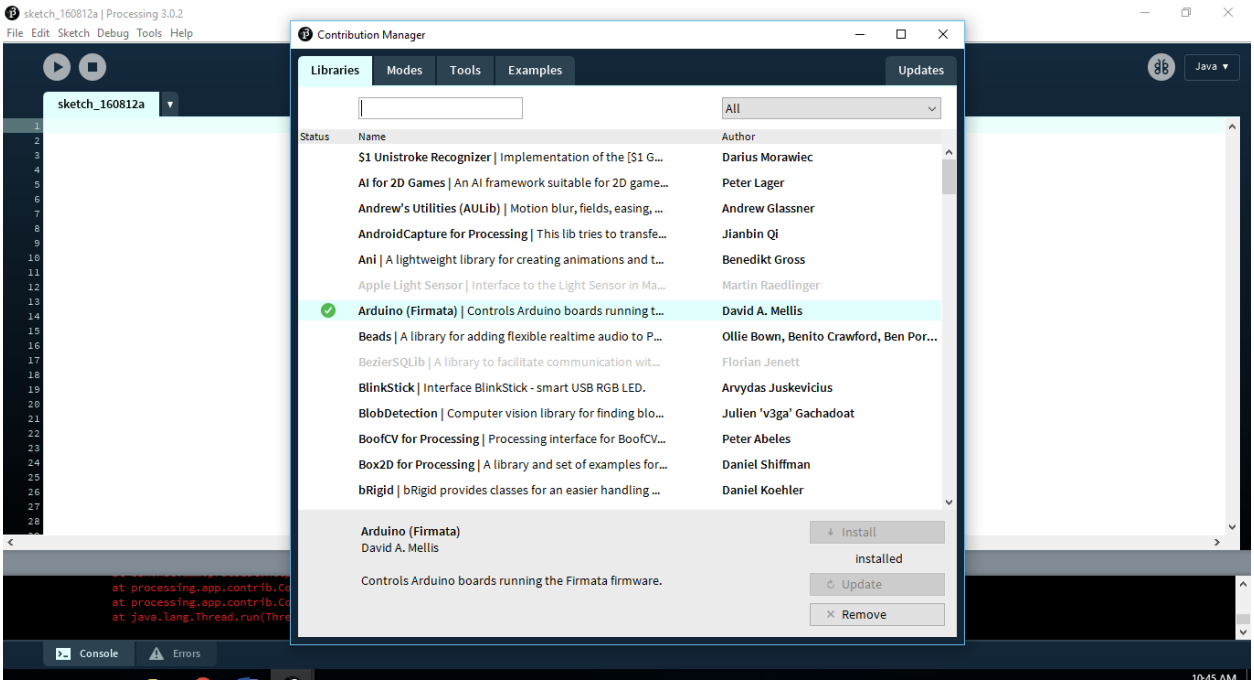


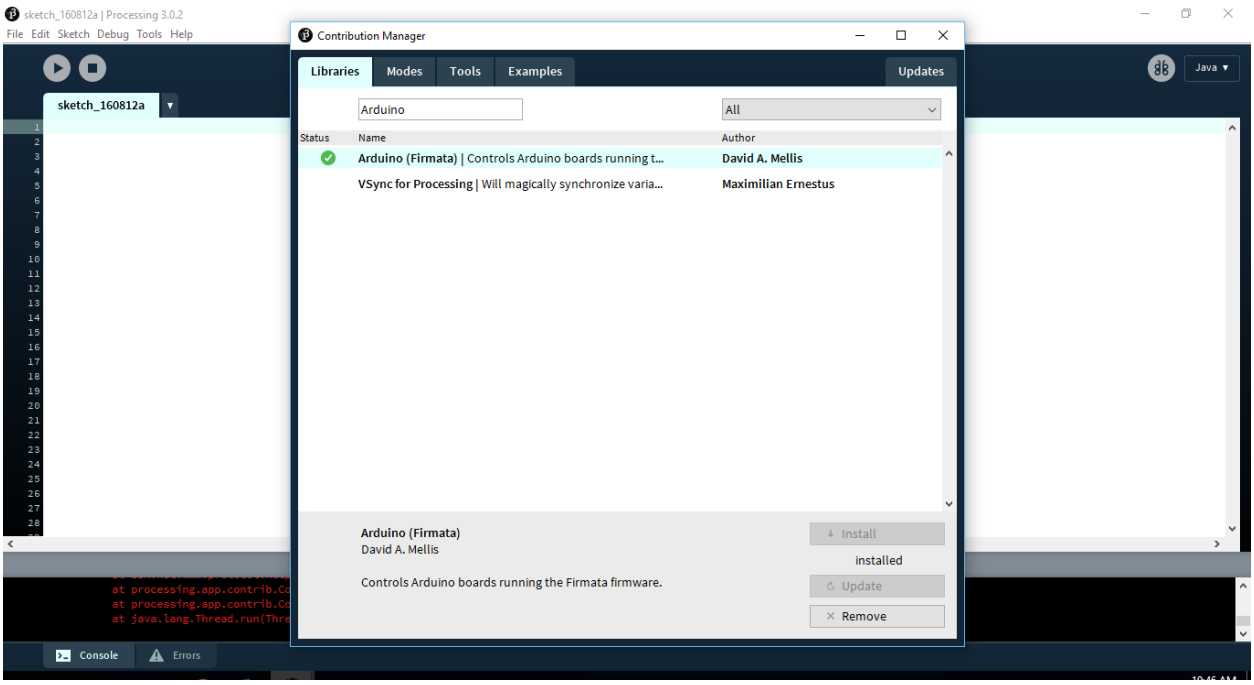
Now that you have uploaded Firmata to Arduino, it will understand the further communication process through Processing environment. Useful syntax given above gives detail of libraries needed to communicate between the two. Follow instructions given in the companion sketches to move further. But before that you need to include a library of Arduino in Processing as well. This because the Processing must be informed about the communication which is going to happen. Go to the following menu to do this:

Sketch-> Import Library-> Add Library …



Now search Arduino in the Search bar and install it.



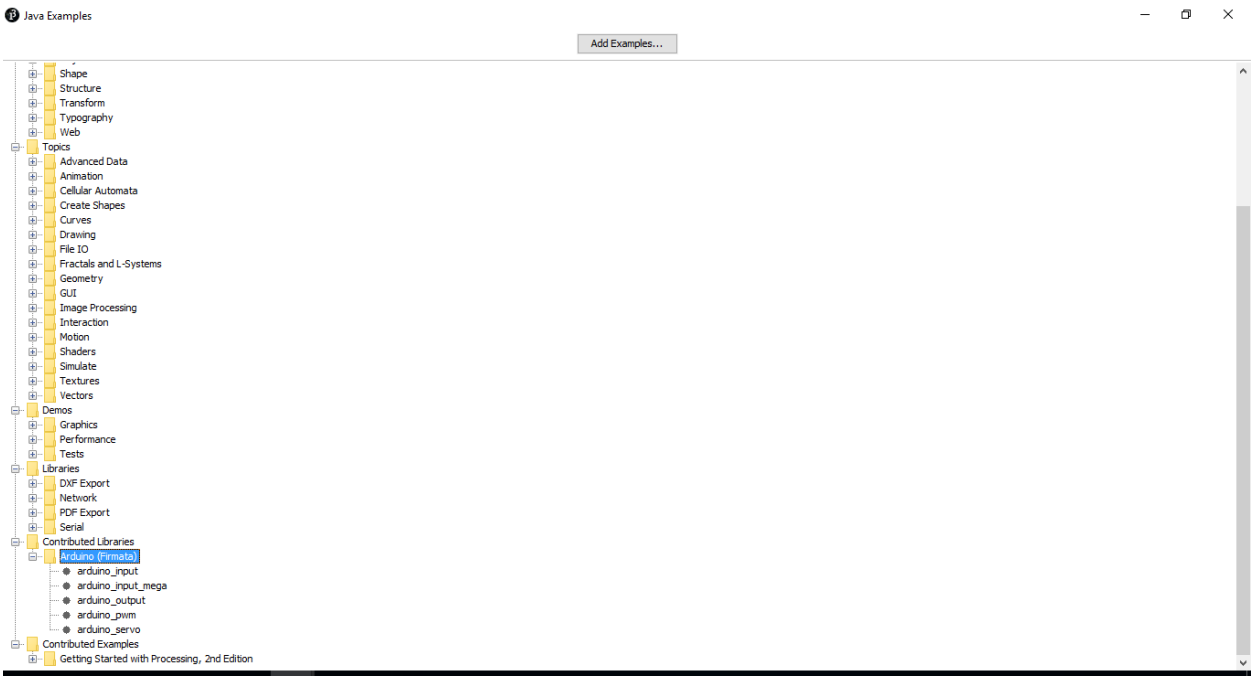


Now that you have developed a good protocol of communication, and that's called Firmata, you will be able to write sketches via Processing and observe the results on Arduino. Now possibilities are limited only by your imagination, so let's see what you can do first!

Tasks

After that you have developed a Firmata Protocol between Arduino and your Computer, you will need to understand the syntax used to speak to Arduino through processing environment. For this purpose, go to Processing bar menu and open following directory:

File->Examples->Contributed Libraries->Arduino (Firmata)



These input and output examples can help you understand how to take inputs and give outputs to Arduino terminals. There are also some example sketches with which you can control Servo Motors and Pulse Width Modulate and output (Analog output).

Now that you are able to provide out and take input separately, will you be able to take an input and lit an LED on base of that result? So let's try it!

What would you like to do further, don't forget to suggest us for upcoming events. Keep Making!