

The Pillar Super Book Archetype

The Pillar team

June 22, 2019

Copyright 2017 by The Pillar team.

The contents of this book are protected under the Creative Commons Attribution-ShareAlike 3.0 Unported license.

You are **free**:

- to **Share**: to copy, distribute and transmit the work,
- to **Remix**: to adapt the work,

Under the following conditions:

Attribution. You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).

Share Alike. If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.

For any reuse or distribution, you must make clear to others the license terms of this work. The best way to do this is with a link to this web page:

<http://creativecommons.org/licenses/by-sa/3.0/>

Any of the above conditions can be waived if you get permission from the copyright holder. Nothing in this license impairs or restricts the author's moral rights.



Your fair dealing and other rights are in no way affected by the above. This is a human-readable summary of the Legal Code (the full license):

<http://creativecommons.org/licenses/by-sa/3.0/legalcode>

Contents

Illustrations	ii
1 Lesson 14 – Firmata implementation for the Pharo Programming Language.	1
1.1 What do we need?	1
1.2 Installation	2
1.3 Digital Pins	4
Bibliography	5

Illustrations

1-1	Arduino.	2
1-2	Standard Firmata Example.	3

Lesson 14 – Firmata implementation for the Pharo Programming Language.

Firmata is a generic protocol for communicating with microcontrollers from software on a host computer. It is intended to work with any host computer software package. Right now there is a matching object in a number of languages. It is easy to add objects for other software to use this protocol. Basically, this firmware establishes a protocol for talking to the Arduino from the host software. The aim is to allow people to completely control the Arduino from software on the host computer.

1.1 What do we need?

In this lesson, we will use a setup with Pharo and Firmata.

Components

- 1 Arduino Uno
- 1 Breadboard
- 1 LED
- 1 Resistor (330 ohms)
- 1 Soil Moisture Sensor
- Jumper wires



Figure 1-1 Arduino.

1.2 Installation

Arduino: Installing Standard Firmata

- Your first step should be to download the Arduino application from [Arduino Website](#).¹
 - Be sure to choose the latest version and also the correct download for your computer and operating system.
 - Once the software has downloaded, you can install the application using the method appropriate for your system.
 - For Mac OS X you will be downloading a ZIP file. Double-clicking on the ZIP should produce a single "Arduino" application file which you can then copy into your Applications folder.
 - For Windows, you should download the .EXE containing a full Windows installer. Double clicking on the .EXE should start the installation.
 - For Linux you will download a compressed TAR file. You can use the "tar" command to uncompress and unpack the application.
 - Plug in Your Arduino Board as shown in Figure 1-1.

Note: Disconnect any wires that may be attached to your Arduino and plug the board into the computer.

- You'll need to select the entry in the Tools > Board menu that corresponds to your Arduino board.

¹<https://www.arduino.cc/en/Main/Software>

1.2 Installation

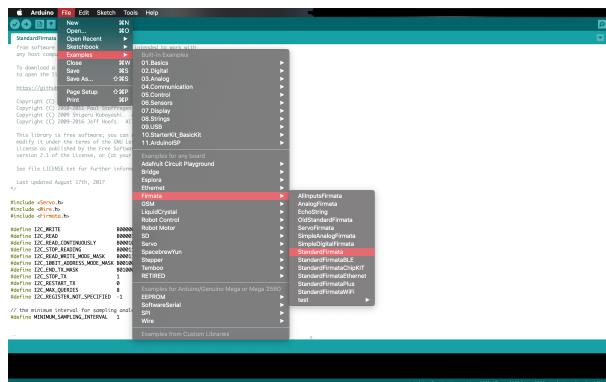


Figure 1-2 Standard Firmata Example.

- Here we are using Arduino UNO board and my port is /dev/tty.usbmodem14201 (You may have a different port name depend on your operating system).
- Upload the Standard Firmata Sketch
 - StandardFirmata is included with the Arduino IDE. To compile and upload StandardFirmata to your Arduino, open the IDE and select File/Examples/Firmata/StandardFirmata.
 - Now at the top of the text editor window, click the "Upload" button on the IDE.
 - At the bottom of the text editor window, you should see a small status window. This will report the progress as the code is compiled and then uploaded to the Arduino.
 - While the code is being uploaded, you should see some very small LED lights (the Transmit (TX) and Receive (RX) lights) on your Arduino board blinking as the data is transferred.

Pharo with Firmata

To load latest version of Firmata you can evaluate the following expression in playground:

```
Metacello new
baseline: 'Firmata';
repository: 'github://pharo-iot/Firmata';
load
```

We have tested this library so far with Arduino Uno and Funduino Uno. To connect to your firmatata enable device, you need the following things:

- Know the device's port name.
- Know its baud rate.
- Install firmata in it (you can do it using your arduino IDE).

1.3 Digital Pins

The digital inputs and outputs (digital I/O) on the Arduino are what allow you to connect the Arduino sensors, actuators, and chips (intergrated circuit). For example in Arduino UNO which is a microcontroller board based on the "ATmega328". It has a few types of pins: **digital pin** (6 can be used as PWM outputs), **analog pins** and orther pins like **power pins**

The digital pins on the Arduino can be configured as either inputs or outputs. There are only two state, either ON or OFF state and these states are represented by the binary values 1 and 0. You use digital signals in situations where the input or output will have one of those two values. In Arduino sketch terms, a ON state is known as HIGH (5V) and OFF state is known as LOW (0V). In other words, we can use them to obtain a digital value (for example, if a button is pressed or not), or to set a value (set if a led is turned on or off) but not both at the same time.

Writing to Digital Pins

To write to a digital pin, you should first set it to output mode using the `#digitalPin:mode:` message. The first argument of the message is the number of the pin, and the second is a numeric value representing the output mode value. We use the `FirmataConstants` class that encapsulates many of the different numeric values used by firmata. In the following example, we set the digital pin 13 in output mode, so we can write to it.

```
[ firmata digitalPin: 13 mode: FirmataConstants pinModeOutput.
```

We can then write to a digital port with the `#digitalWrite:value:` message, giving the pin number as first argument, and the value to write (0 or 1) as second argument. Thus, to turn on the pin 13 we can do:

```
[ firmata digitalWrite: 13 value: 1.
```

And to turn it off:

```
[ firmata digitalWrite: 13 value: 0.
```

<https://www.instructables.com/id/Arduino-Installing-Standard-Firmata/>

Bibliography

