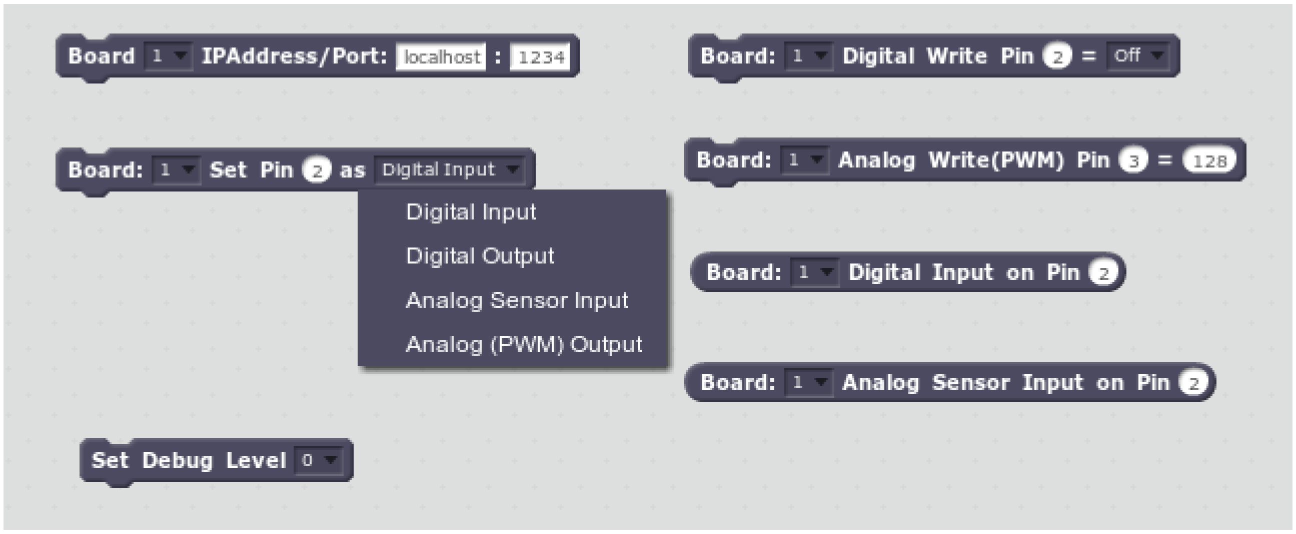
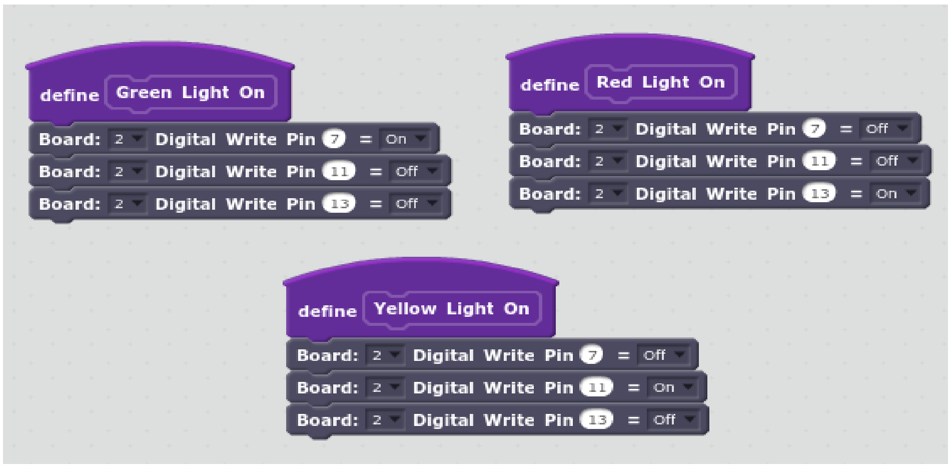
Xi4S

Xi For Scratch

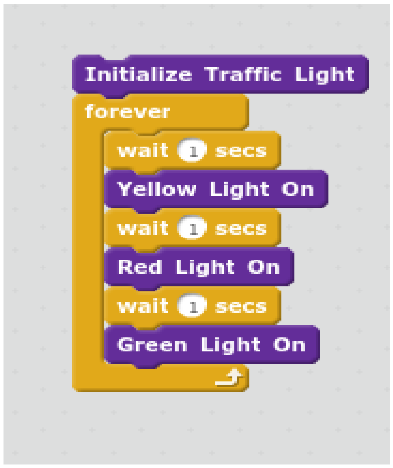
Installation and Usage Guide



```
Board 1 IPAddress/Port: localhost : 1234

Board: 1 Set Pin 2 as Digital Input
    Digital Input
    Digital Output
    Analog Sensor Input
    Analog (PWM) Output

Set Debug Level 0

Board: 1 Digital Write Pin 2 = Off

Board: 1 Analog Write(PWM) Pin 3 = 128

Board: 1 Digital Input on Pin 2

Board: 1 Analog Sensor Input on Pin 2
```



```
define Green Light On
Board: 2 Digital Write Pin 7 = On
Board: 2 Digital Write Pin 11 = Off
Board: 2 Digital Write Pin 13 = Off

define Red Light On
Board: 2 Digital Write Pin 7 = Off
Board: 2 Digital Write Pin 11 = Off
Board: 2 Digital Write Pin 13 = On

define Yellow Light On
Board: 2 Digital Write Pin 7 = Off
Board: 2 Digital Write Pin 11 = On
Board: 2 Digital Write Pin 13 = Off
```



```
Initialize Traffic Light
forever
    wait 1 secs
    Yellow Light On
    wait 1 secs
    Red Light On
    wait 1 secs
    Green Light On
```

The Cross-Platform Interconnect

Software Release v.002

Document Release Date: 19 September 2014

Table of Contents

1. Introduction

Xi4S is a JavaScript client implementation of the latest Scratch 2.0 Hardware Extension specification. With Xi4S, you will be able to connect one or several hardware boards to Scratch, in any combination, and have Scratch control them all. Scratch can receive data from one board and then use that data to control another board. Each board type has its own server - XiDuino for Arduino, XiBone for BeagleBone Black or XiPi for Raspberry Pi.

For Arduino boards, a single instance of the XiDuino server is capable of controlling a single or multiple Arduino boards. For the BeagleBone and Raspberry Pi devices, each device will have its own server instance - each BeagleBone Black will run its own copy of XiBone, and each Raspberry Pi will run its own copy of XiPi.

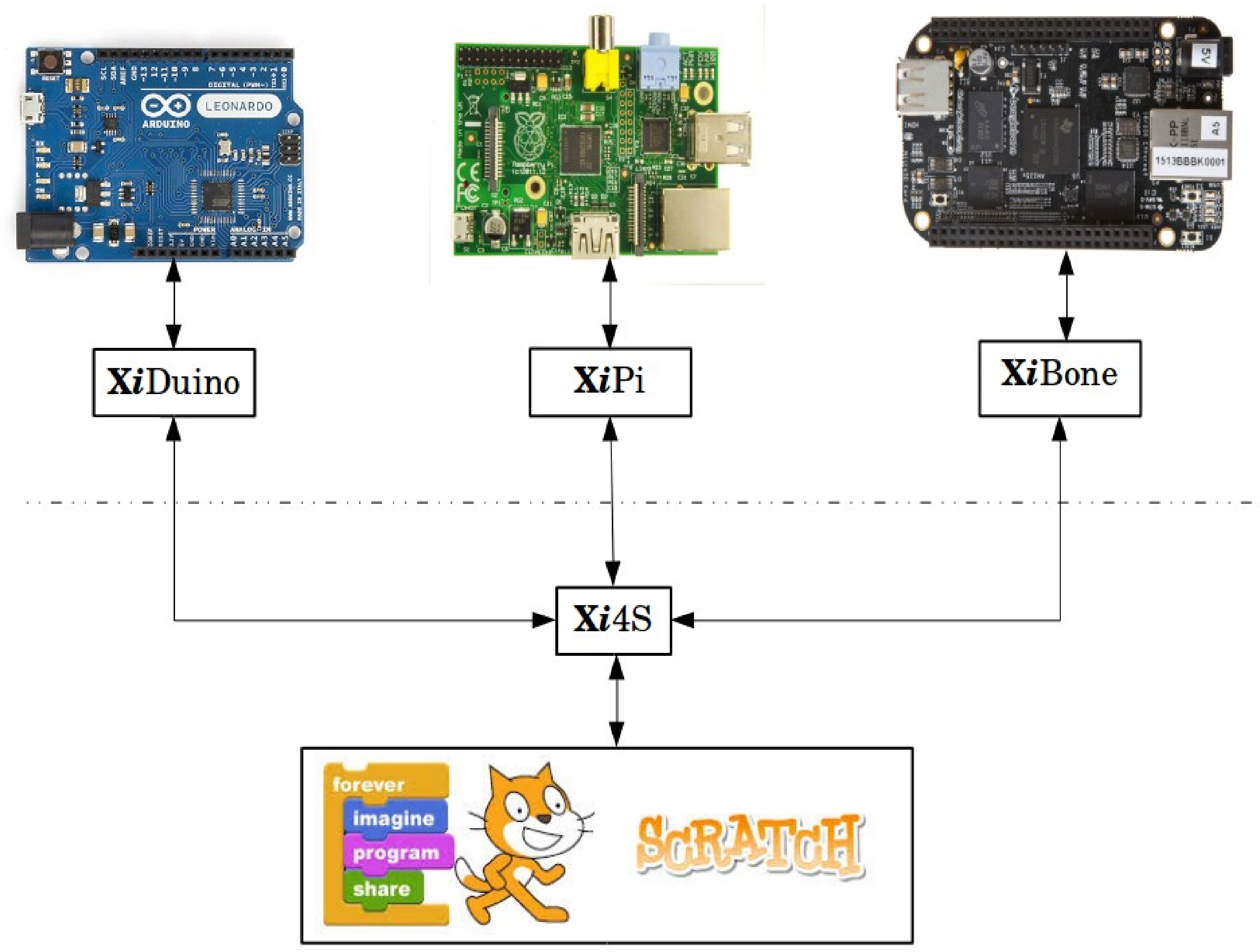


Figure 1: Xi4S Architectural Model

In a nutshell, Xi provides a consistent and easy to use Scratch programming interface that supports the most popular micro-controllers and single board computers. It makes programming these devices as easy as creating any other Scratch program.

1.1. Supported Operating Systems

Xi is operating system independent and designed to run on:

1.1.1. Linux

- Desktop – tested on Ubuntu 14.04
- BeagleBone Black – tested on the Debian eMMC 2014-05-14 version.
- Raspberry Pi – tested on the Raspbian Debian Wheezy 2014-06-20 version.

1.1.2. Windows 8.1

- Tested on Windows 8.1 32 bit, but Windows 7 should work as well.

1.1.3. MAC

- It has not yet been tested on MAC, but since Xi uses components that are known to work on the MAC, it is fully anticipated that it will work without issue.
- Use the Linux installation instructions as a guide.
- **NOTE:** If you are a MAC user and wish to try Xi4S, please send your results, good or bad, to MisterYsLab@gmail.com

1.2. Verified Supported Board Types

1.2.1. Arduino

Tested with Uno and Leonardo (Support for Yun planned for a future release).

1.2.2. BeagleBone Black

Tested with Rev A6A. (Should work for any version running Debian)

1.2.3. Raspberry Pi

Tested with Model B version 2. (Should work with A, B version 1, and B+ as well).

1.3. Functionality Matrix

This first release of Xi4S provides a small subset of the ultimate functionality goals. The table below summarizes the current level of support for each board type.

Board Type	Digital Input	Digital Output	Analog Input	Analog Output (PWM)
Arduino	X	X	X	X
BeagleBone Black	X	X	X	X
Raspberry Pi	X	X		

In addition, XiDuino provides automatic COM port detection (thanks to the Johnny-Five library). It also automatically launches your default browser to the Scratch web page when it is started.

If you are not using an Arduino, you will have to manually launch your web browser.

1.4. Browser Support

Xi4S works best with the Chrome browser, but works with Firefox as well. It will ***not*** work with Internet Explorer.

2. Installation

This section begins with the Xi server installation procedures. Each server type is addressed separately and for Arduino boards, a separate procedure is provided for Linux and Windows. Mac users can use the Linux procedures as a guide.

2.1. Xi Servers

A Xi server must be installed for all boards that you wish to connect to Scratch.

For Arduino boards, you need to install a single copy of XiDuino on your PC, independent of how many Arduino boards you are running.

For BeagleBone Black and Raspberry Pi, XiBone or XiPi respectively, needs to be installed on each board you are adding to the system.

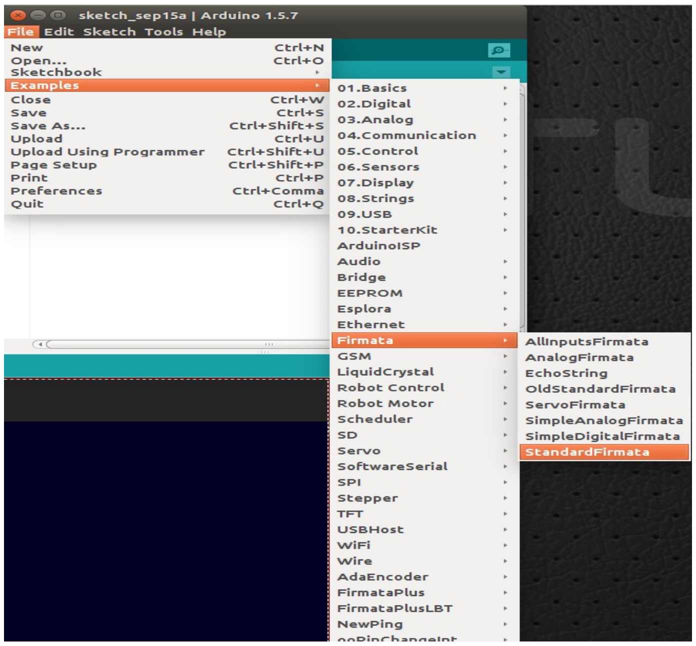
After all server files have been installed, go to the Xi4S Client installation section to install Xi4S into Scratch.

2.1.1. XiDuino

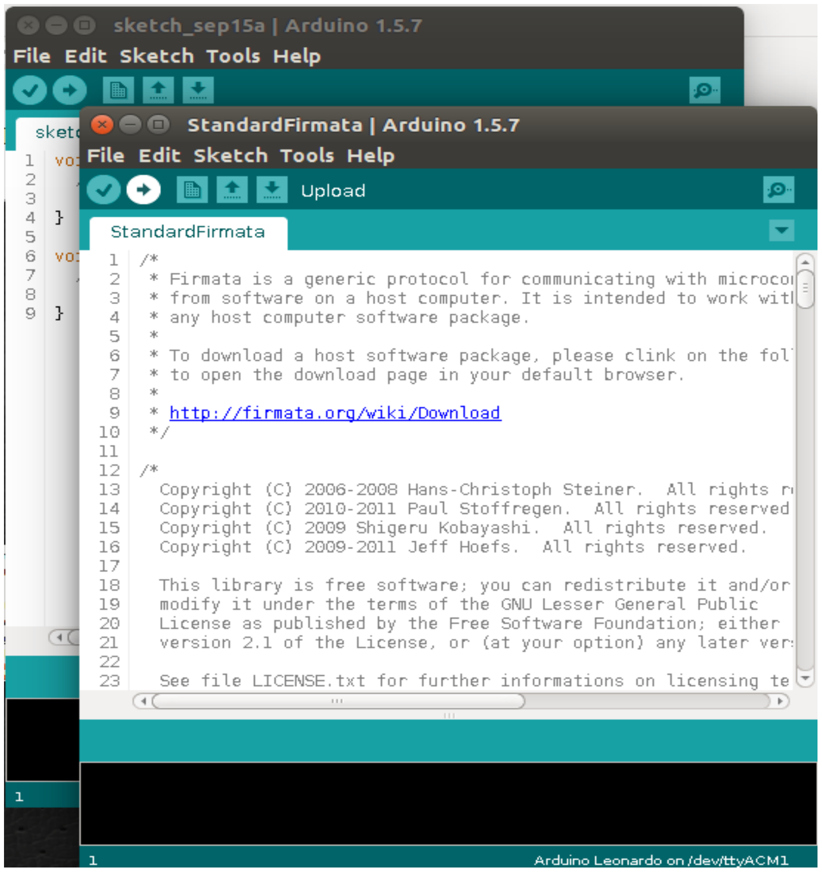
XiDuino requires that either Standard Firmata or FirmataPlus be installed on your Arduino board.

If you need to install StandardFirmata, first install the Arduino IDE for your platform from here: http://arduino.cc/en/Main/Software.

Next, open the IDE and then click on File/Examples/Firmata/StandardFirmata.



Finally upload to the Arduino board.


sketch_sep15a | Arduino 1.5.7
File Edit Sketch Tools Help
StandardFirmata | Arduino 1.5.7
File Edit Sketch Tools Help
Upload
StandardFirmata
Arduino Leonardo on /dev/ttyACM1

```
/*
 * Firmata is a generic protocol for communicating with microco
 * from software on a host computer. It is intended to work wit
 * any host computer software package.
 *
 * To download a host software package, please clink on the fol
 * to open the download page in your default browser.
 *
 * http://firmata.org/wiki/Download
 */

/*
  Copyright (C) 2006-2008 Hans-Christoph Steiner.  All rights r
  Copyright (C) 2010-2011 Paul Stoffregen.  All rights reserved
  Copyright (C) 2009 Shigeru Kobayashi.  All rights reserved.
  Copyright (C) 2009-2011 Jeff Hoefs.  All rights reserved.

  This library is free software; you can redistribute it and/or
  modify it under the terms of the GNU Lesser General Public
  License as published by the Free Software Foundation; either
  version 2.1 of the License, or (at your option) any later ver

  See file LICENSE.txt for further informations on licensing te
```

2.1.1.1 Linux Desktop

Install The Required Development Tools

Make sure that you have the following tools in place on your system. If you are using Ubuntu, they may be installed using the Ubuntu Software Center. Ubuntu package names are given in the table below.

For distributions other than Ubuntu, check your distribution package manager to determine if the required packages have been installed.

Tool	Ubuntu Software Center Package Name
C++ Compiler	build-essential
make	build-essential
curl	Command line tool for transferring data with URL syntax
Python 2.6 or 2.7	Python Interactive high-level object-oriented language (default version)

Installing node.js

Follow the directions for your distribution on this web page:
https://github.com/joyent/node/wiki/Installing-Node.js-via-package-manager

For Ubuntu 14.04, I used the following commands to install node.js

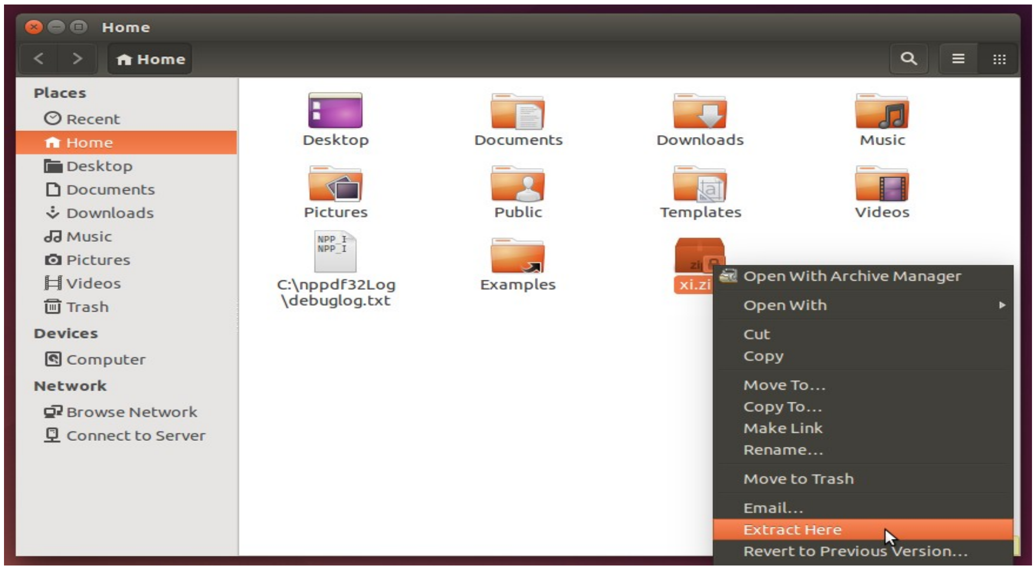
```
curl -sL https://deb.nodesource.com/setup | bash -

sudo apt-get install nodejs

node -v
```

The node -v command will display the version of node that was installed.

Download the Xi Distribution

Download the Xi distribution from github.

Place this file in your home directory and extract all of the files.



After extracting, open a command terminal and enter the following command.

```
cd ~/xi/servers/xiduino/linux
```

Next, change the permissions on both files in this directory by typing:

```
chmod ugo+x *
```

Run the Install Shell Script

In `~/xi/servers/xiduino/linux, enter the following command:`

```
bash xiduinoInstall.sh
```

Connect All Arduino Boards to Computer

Connect all Arduino boards to your computer using a separate USB cable for each.

Start XiDuino

To start XiDuino, type the following commands:

```
cd  ~/xiduino

bash xiduino.sh
```

A command terminal window will open and your default browser will be launched to the Scratch home page.

After a few moments, you command window should contain something like the following:

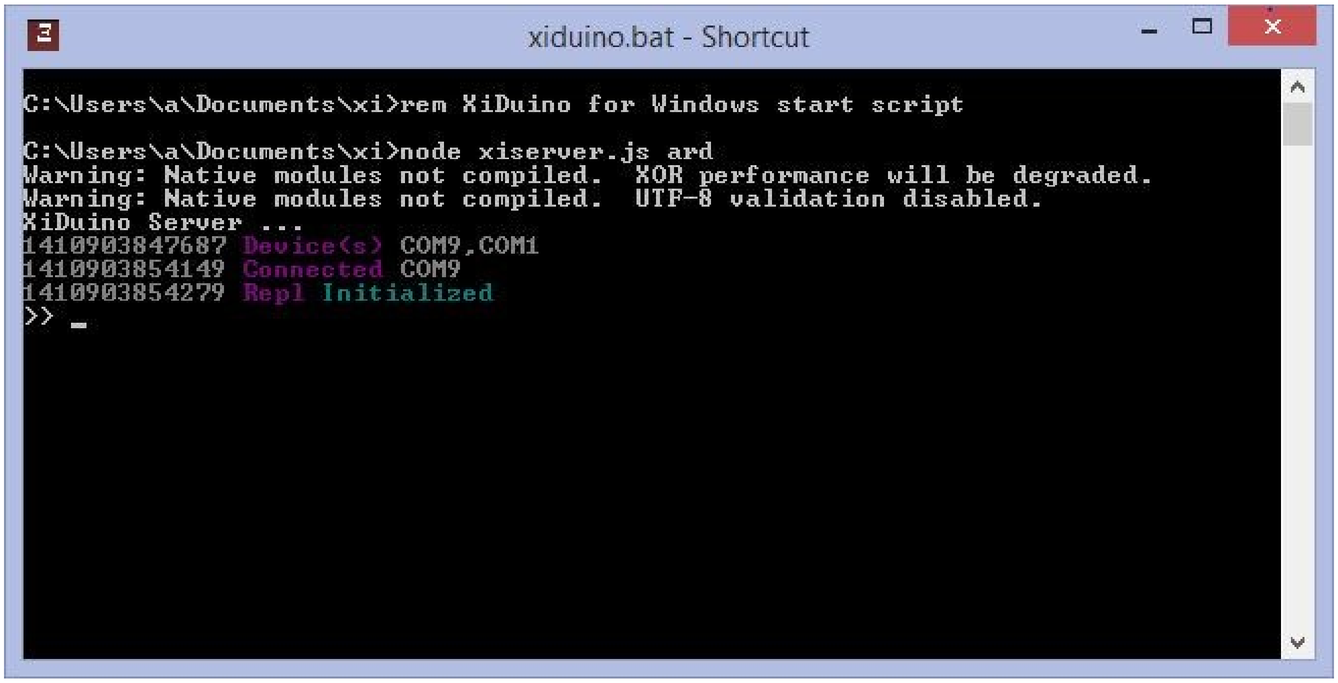
```
Starting XiDuino server...

XiDuino Server ...
```

```
1410474902988 Device(s) /dev/ttyACM0

1410474908009 Connected /dev/ttyACM0

1410474908009 Repl Initialized
```

2.1.1.2 Windows 7 or 8.1

Install The Optional Development Tools

This is an optional step. If you choose to skip it, when XiDuino starts, you may see the following warnings:



```
C:\Users\a\Documents\xi>rem XiDuino for Windows start script

C:\Users\a\Documents\xi>node xiserver.js ard
Warning: Native modules not compiled.  XOR performance will be degraded.
Warning: Native modules not compiled.  UTF-8 validation disabled.
XiDuino Server ...
1410903847687 Device(s) COM9,COM1
1410903854149 Connected COM9
1410903854279 Repl Initialized
>> _
```

XiDuino appears to run without problems in spite of these warnings, however if you wish to run a natively compiled version of websocket, you will need to load Python 2.6 or 2.7, and the Visual Express Development Package before running the xiduinoInstall.bat script.

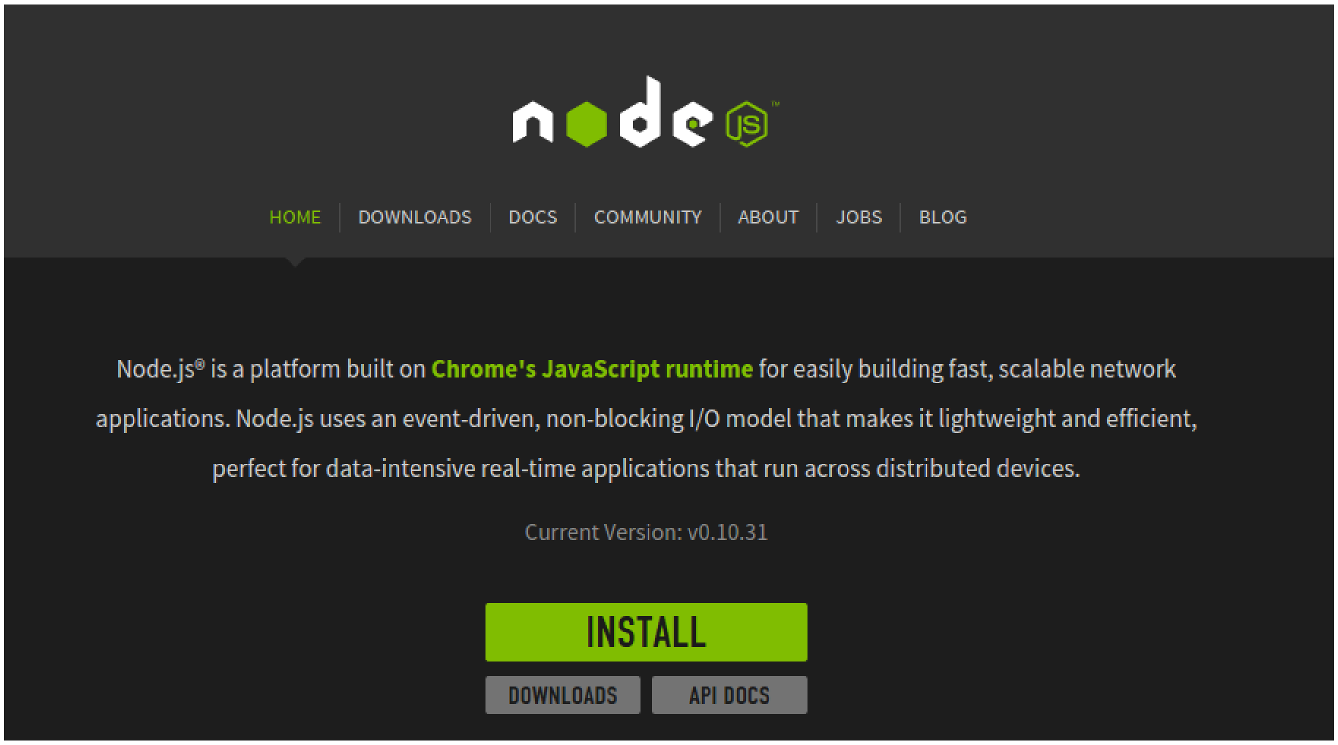
Be warned, the Visual Express Development package takes up a huge amount of disk space when it is installed.

To install Python 2.7, go to this web page, https://www.python.org/download/releases/2.7.8/ and download and install.

To install the Visual Express Package, go to this web page, http://www.visualstudio.com/en-us/products/visual-studio-express-vs.aspx, download and install.

Install node.js

To install node.js, go to this web page: http://nodejs.org/ and click on the green install button:

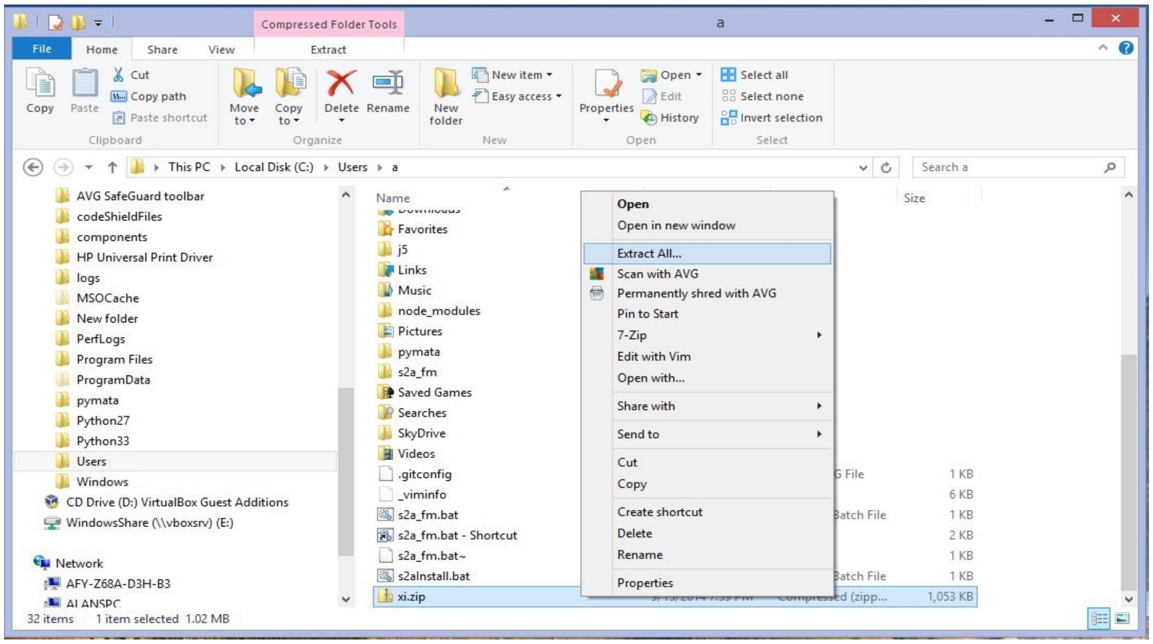


Node.js® is a platform built on **Chrome's JavaScript runtime** for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Current Version: v0.10.31

Download the Xi distribution from github.

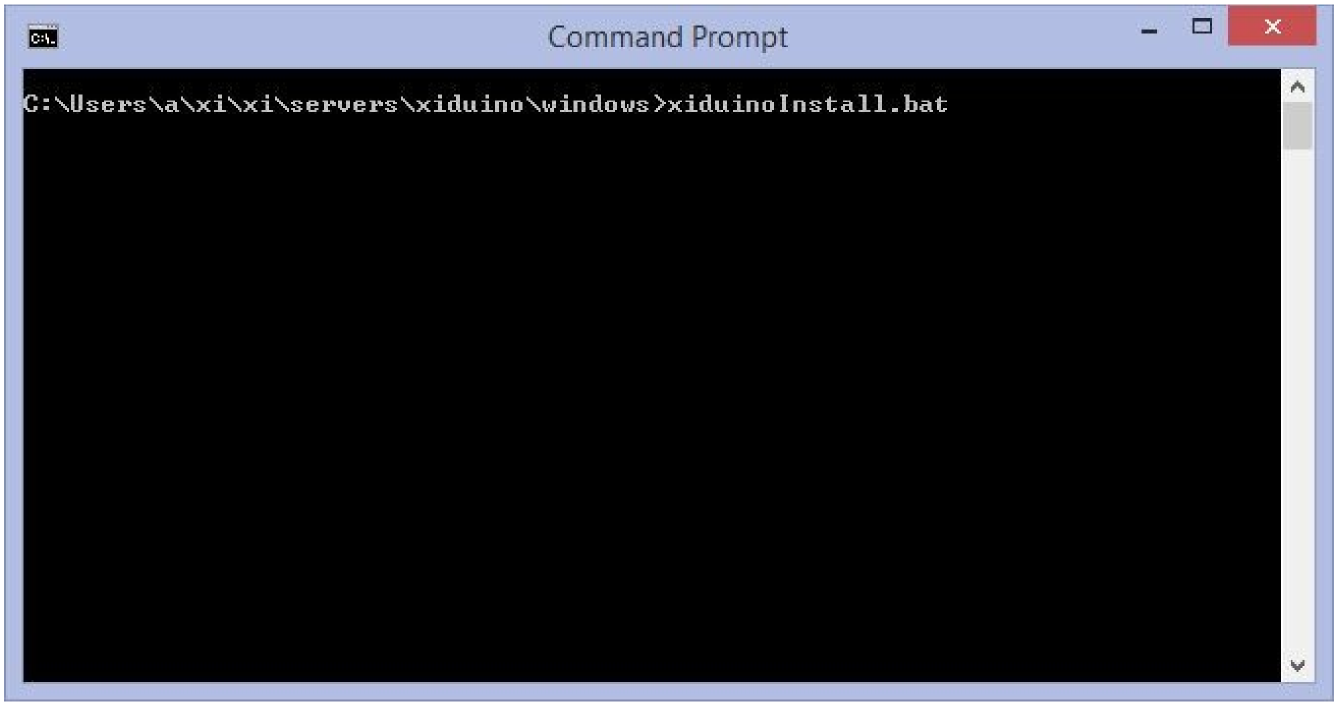
Extract the files from the Xi archive file using the Windows Explorer.



Run the Install .bat Script

Open a Command window and navigate to the xi\servers\xiduino\windows directory.

Type xiduinosInstall.bat and press return to begin the installation.

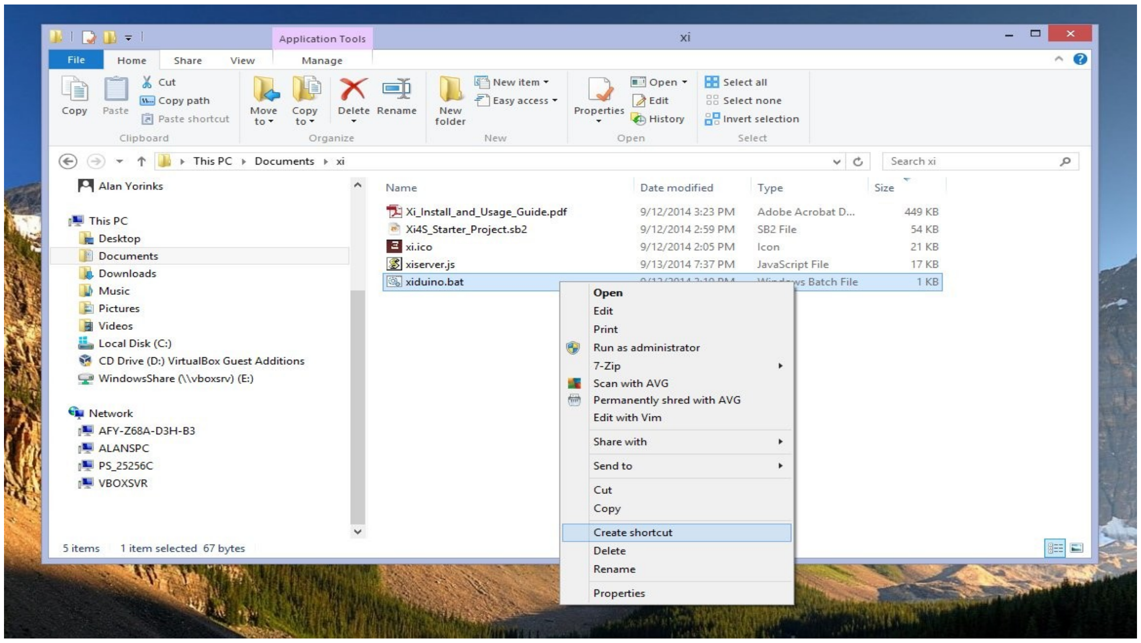


```
C:\Users\a\xi\xi\servers\xiduino\windows>xiduinoInstall.bat
```

This will take several minutes to complete.

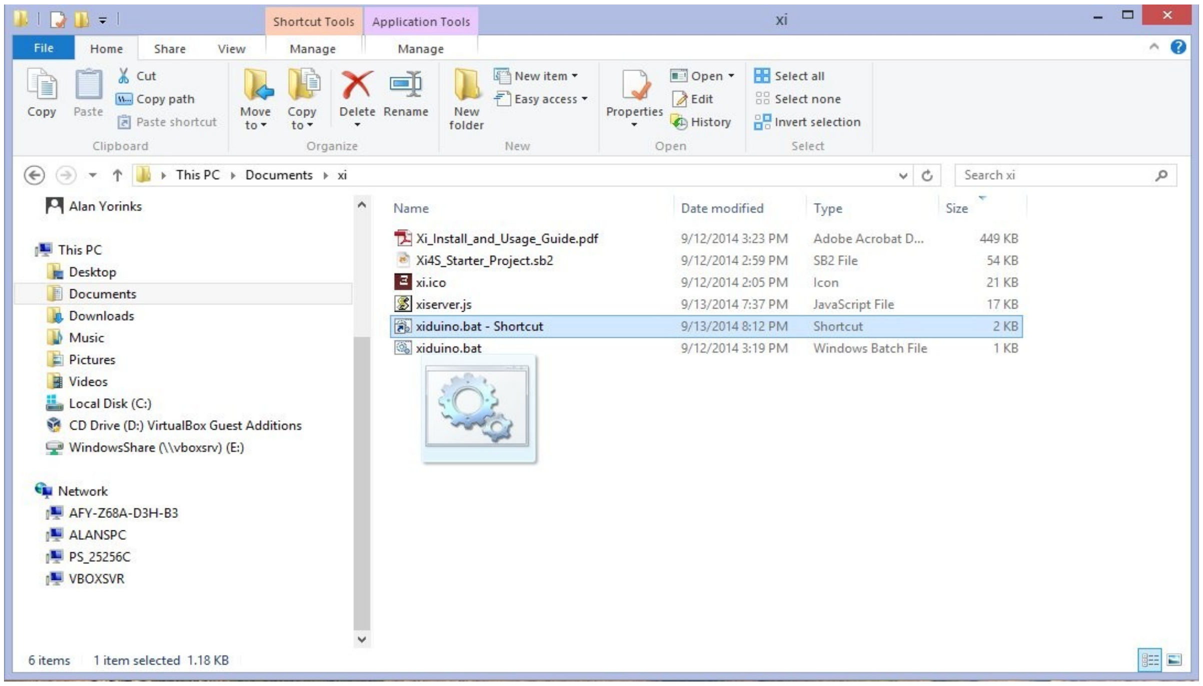
Creating a Desktop Launch Icon

After the installation scripts completes, open the Windows Explorer and navigate to your home Documents directory, and then select the xi directory. Right click on xiduino.bat and select "Create Shortcut" from the drop down menu.



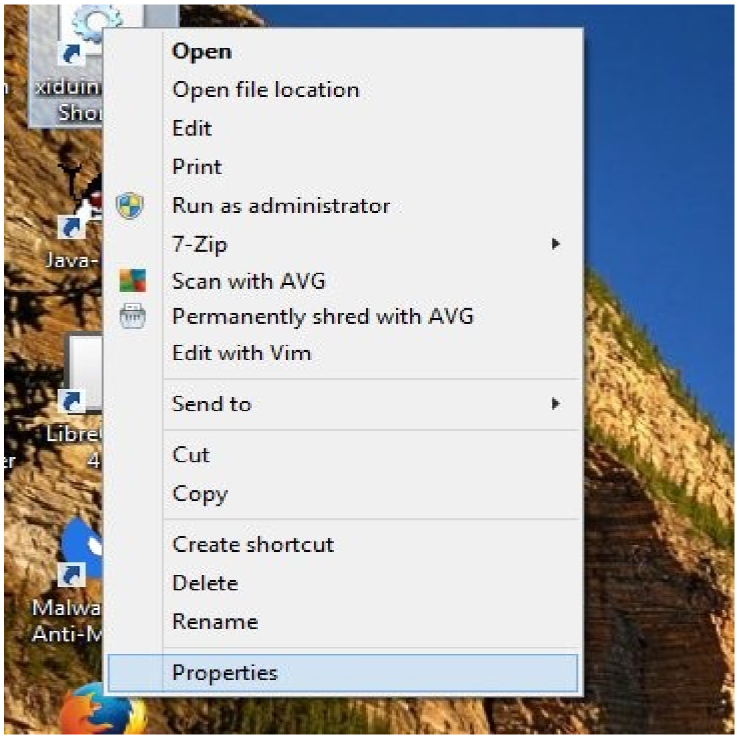
Name	Date modified	Type	Size
Xi_Install_and_Usage_Guide.pdf	9/12/2014 3:23 PM	Adobe Acrobat D...	449 KB
Xi4S_Starter_Project.sb2	9/12/2014 2:59 PM	SB2 File	54 KB
xi.ico	9/12/2014 2:05 PM	Icon	21 KB
xiserver.js	9/13/2014 7:37 PM	JavaScript File	17 KB
xiduino.bat	9/12/2014 2:10 PM	Windows Batch File	1 KB

Then drag the xiduino.bat – shortcut listed in the Explorer to the Windows desktop

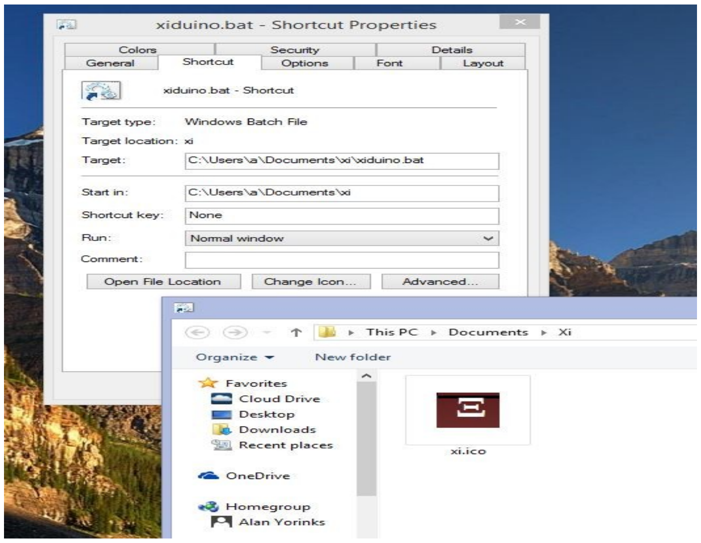


Name	Date modified	Type	Size
Xi_Install_and_Usage_Guide.pdf	9/12/2014 3:23 PM	Adobe Acrobat D...	449 KB
Xi4S_Starter_Project.sb2	9/12/2014 2:59 PM	SB2 File	54 KB
xi.ico	9/12/2014 2:05 PM	Icon	21 KB
xiserver.js	9/13/2014 7:37 PM	JavaScript File	17 KB
xiduino.bat - Shortcut	9/13/2014 8:12 PM	Shortcut	2 KB
xiduino.bat	9/12/2014 3:19 PM	Windows Batch File	1 KB

Right click on the default “gears” icon and select Properties, and then select Change Icon …



Go through the menus until you find the xi.ico file and select and apply it to the shortcut.

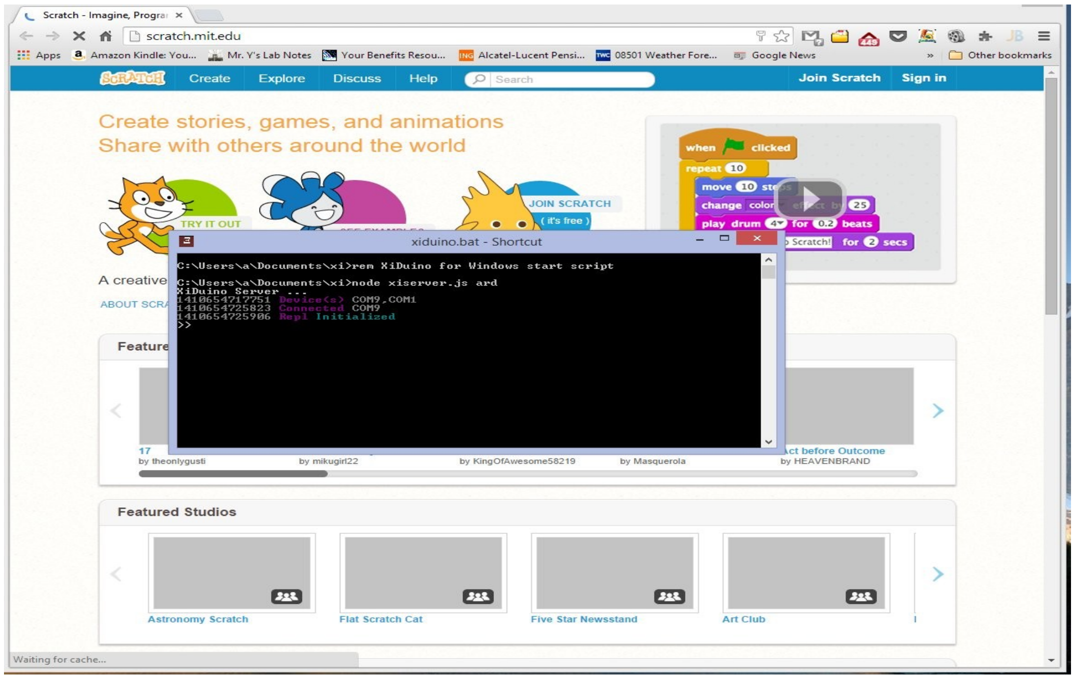


xiduino.bat - Shortcut

Target type: Windows Batch File
Target location: xi
Target: C:\Users\a\Documents\xi\xiduino.bat
Start in: C:\Users\a\Documents\xi
Shortcut key: None
Run: Normal window
Comment:

If you like, right click on the icon again and rename the icon title to XiDuino.

After you are done, when you double click the Xi icon, a command window will open up and the default browser will be launched. Remember to change your default browser to either Chrome or Firefox.



xiduino.bat - Shortcut

```
C:\Users\a\Documents\xi>rem XiDuino for Windows start script

C:\Users\a\Documents\xi>node xiserver.js ard
XiDuino Server ...
1410654717751 Device(s) COM9,COM1
1410654725823 Connected COM9
1410654725906 Repl Initialized
>>
```

2.1.1.3 Manually Specifying The COM Port For XiDuino

You may optionally command XiDuino to manually select a specific COM port.

2.1.1.3.1 For Linux

To do so for Linux, you will have to edit xiduino.sh and change the following line:

sudo node ~/xiduino/xiserver.js ard

to

sudo node ~/xiduino/xiserver.js ard 0 “YOUR_COM_PORT”

For example sudo node ~/xiduino/xiserver.js ard 0 “/dev/ttyACM0”

2.1.1.3.2 For Windows

To do so for Windows, you will have to edit xiduino.bat and change the following line:

node xiserver.js ard

to

node xiserver.js ard 0 “YOUR_COM_PORT”

For example node xiserver.js ard 0 “COM9”

2.1.2. XiBone

XiBone was tested using the Debian 2014-05-15 eMMC image.

To install, it is recommended that you login as a user other than root. Make sure that your BeagleBone Black is connected to the Internet.

2.1.2.1 Node.js

Node.js is pre-installed for the BeagleBone Black, so it does not need to be installed.

2.1.2.2 Required Supporting Development Packages

All supporting packages are already pre-installed.

2.1.2.3 Node Module Installation

Make sure that your BeagleBone Black is connected to the internet before continuing. Both the Installation and execution of XiBone requires that the date be set to the current date. The installation and execution shell scripts contain a call to ntpdate to make sure that the date is current.

1. Download the Xi distribution from github.
2. Extract the files from the downloaded archive.

3. In a command terminal, CD to the servers/xibone directory

4. Change the file permissions for xiboneInstall.sh

```
chmod ugo+x xiboneInstall.sh
```

5. Execute the install script by typing:

```
./xiboneInstall.sh
```

6. The installation will take several minutes. If you see some warnings, you can ignore them.

The installation script creates a directory called xibone that is located below the home directory. This directory is populated with all the files you need to run the XiBone server.

2.1.2.4 Starting the Server

To start the XiBone server type the following:

```
cd ~/xibone

bash xibone.sh
```

Here is the output from a typical XiBone session:

```
debian@beaglebone:~/xi/servers/xibone$ cd ~/xibone

debian@beaglebone:~/xibone$ bash xibone.sh

setting current data and time from ntp

13 Sep 18:09:26 ntpdate[5159]: step time server 138.236.128.36 offset
-0.038487 sec

Starting XiBone server...

XiBone Server ...

1410631769971 Device(s) BeagleBone-IO

1410631769998 Connected BeagleBone-IO

1410631770000 Repl Initialized

>>
```

To end a session, press CTRL-C twice.

2.1.3. XiPi

XiPi was tested using RASPBIAN Debian Wheezy, version Linux raspberrypi 3.12.22+ #691 PREEMPT Wed Jun 18 18:29:58 BST 2014 armv6l GNU/Linux.

To install XiPi, it is recommended that you login as a user other than root. Make sure that your Raspberry Pi is connected to the Internet.

2.1.3.1 Node.js

To install node.js, open an LXTerminal and type in the following commands:

```
sudo apt-get update

sudo apt-get upgrade

sudo wget http://node-arm.herokuapp.com/node_latest_armhf.deb

sudo dpkg -i node_latest_armhf.deb
```

To verify that the package has been installed properly, and to see the node's version type:

```
node -v
```

2.1.3.2 Required Supporting Development Packages

All supporting packages are already pre-installed.

2.1.3.3 Node Module Installation

1. Download the Xi distribution from github.
2. Extract the files from the downloaded archive.
3. In a command terminal, CD to the servers/xipi directory
4. Change the file permissions for xipiInstall.sh

```
chmod ugo+x xipiInstall.sh
```

Execute the install script by typing:

```
./xipiInstall.sh
```

5. The installation will take several minutes. If you see some warnings, you can ignore them.

The installation script creates a directory called xipi that is located below the home directory. This directory is populated with all the files you need to run the XiPi server.

2.1.3.4 Starting the Server

To start the XiBone server type the following:

```
cd ~/xipi

bash xipi.sh
```

Here is the output from a typical XiPi session:

```
pi@raspberrypi:~/xipi$ bash xipi.sh
Starting XiPi server...
XiPi Server ...
1410633708634 Device(s) RaspberryPi-IO
1410633708803 Connected RaspberryPi-IO
1410633708812 Repl Initialized
>>
```

To end a session, press CTRL-C twice.

2.2. Xi4S Client

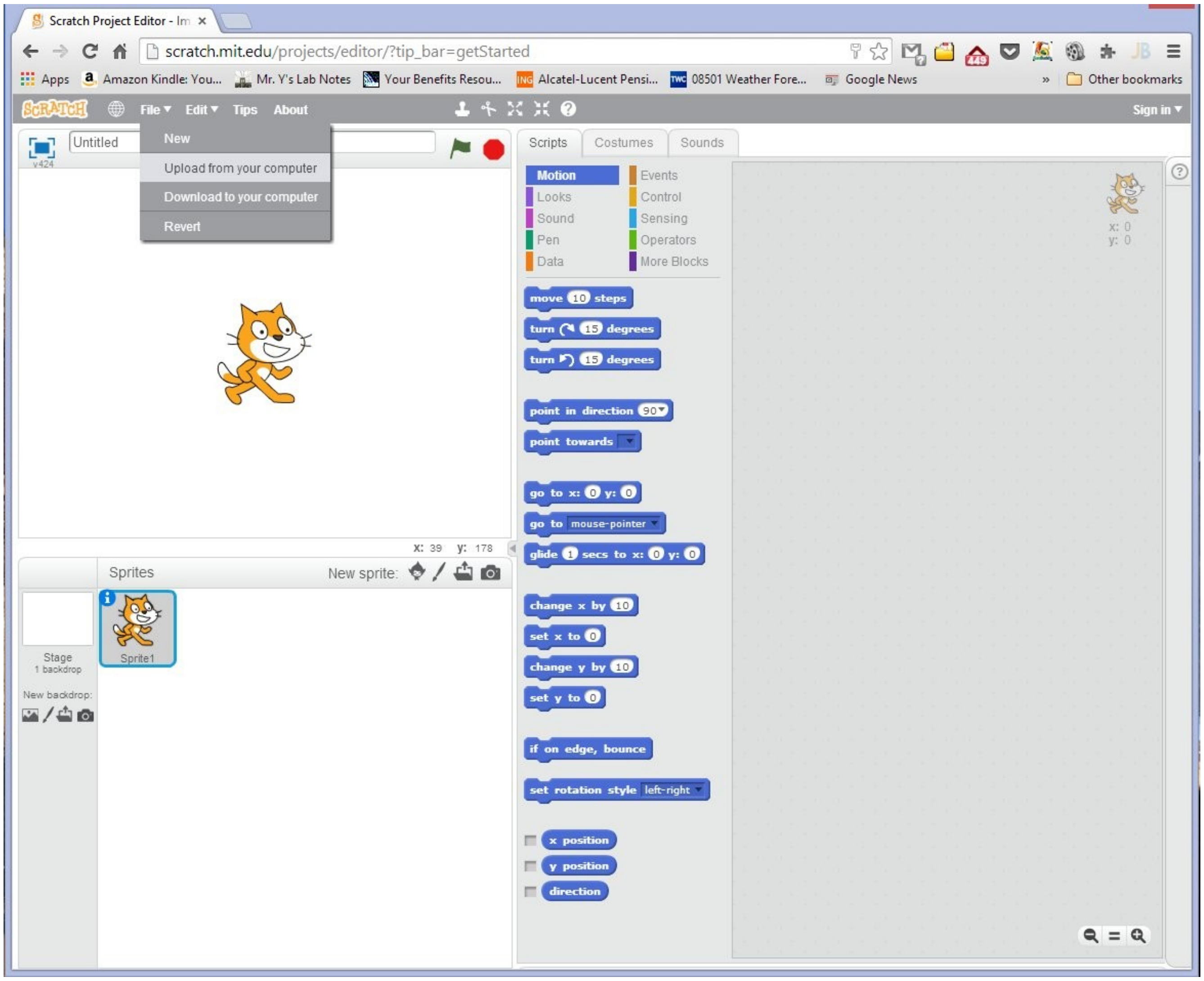
IMPORTANT NOTE: Scratch may prompt you to load a “plugin” when using Xi. **DO NOT LOAD THIS PLUGIN WHILE USING XI.** It is unnecessary to do so, and may in fact interfere with Xi's Arduino support.

The Scratch extension client, Xi4S, is automatically loaded into Scratch when an existing project that contains Xi4S is uploaded to the editor. Currently, this is the only option provided by the Scratch team to install the extension.

A starter project, containing the extension and extension blocks, is included with this distribution. It is located in the archived file you extracted earlier. It is located in the xi/clients/scratch/projects directory and the file is called Xi4S_Starter_Project.sb2.

To install it, open the Scratch on-line editor, click on File/Upload from your computer and select Xi4S_Starter_Project.sb2. After uploading the starter project, you probably will want to rename it and save it under a name that is meaningful to you.

You will not be able to share a project containing Xi4S, again a limitation imposed by the Scratch team, but you can save the program to your computer and make a copy of the file if you wish to share it.

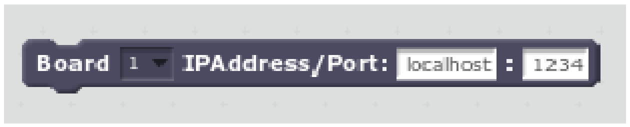


The source code for the extension is included in this package and is located in xi/clients/scratch/xi4s and the file is name xi4s.js. It is provided for those interested in gaining an understanding of how Xi4S implements the Scratch JavaScript Hardware Extension Specification.

3. Using the Xi4S Scratch Blocks

3.1. Selecting an IPAddress for the Connected Boards

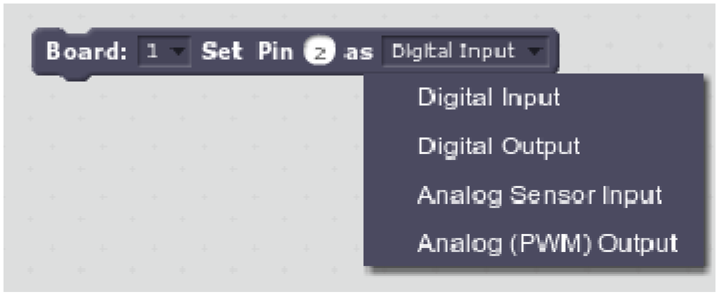
Before using any other blocks for a given board, you will need to set the board's IP address and port number.



Boards are identified by the Board number drop down list. Up to 10 boards can be supported. The IPAddress is the address of the board. If you are using Arduino boards, this address is typically “localhost”. For BeagleBone Black and Raspberry Pi, the address is set to the IP address of the board.

The port number, 1234, is the default port number for all of the servers. It typically should not be changed. If you need to change it, you will also need to change the port number in the xiserver.js file for that board.

3.2. Setting the Pin Mode



Currently, only 4 pin modes are supported as shown in the screen shot above. Additional modes will be added as the project matures.

3.3. Pin Designations

3.3.1. Arduino

The standard pin numbering is used. For PWM pins, you just type in the pin number and ***not*** the A prefix. So if you want Pin 6 to be a PWM pin, the pin number would be 6 (***not A6***).

3.3.2. BeagleBone Black

The BeagleBone Black uses a mapping of its pins to an Arduino Uno numbering scheme. The mapping matrix is below. I will be investigating if this mapping can be expanded in the future.

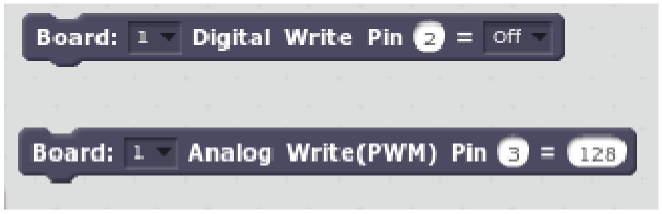
BBB Port	Arduino Pin	Type
P8_7	0	Digital
P8_8	1	Digital
P8_9	2	Digital
P8_13	3	PWM
P8_10	4	Digital
P9_14	5	PWM
P8_16	6	PWM
P8_11	7	Digital
P8_12	8	Digital
P9_21	9	PWM
P9_22	10	PWM - Currently not working
P9_28	11	PWM - Currently not working
P8_14	12	Digital
USR3	13	Digital / Default Led
P9_39	A0	Analog Input
P9_40	A1	Analog Input
P9_37	A2	Analog Input
P9_38	A3	Analog Input
P9_35	A4	Analog Input
P9_36	A5	Analog Input

3.3.3. Raspberry Pi

Pin numbers are identified by their pin number on the P1 header, so for example, if you want to use GPIO 17, specify pin 11.

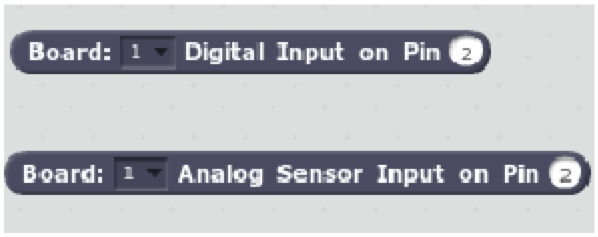
The author of the Raspi-IO package has written a package that supports the Wiring Pi interface. I will be investigating to see if that can be integrated into Xi as well.

3.4. Command Blocks


Board: 1 Digital Write Pin 2 = Off
Board: 1 Analog Write(PWM) Pin 3 = 128

Digital and Analog writes are performed using these blocks. The board number, pin number and value are specified to control the output value of the selected pin.

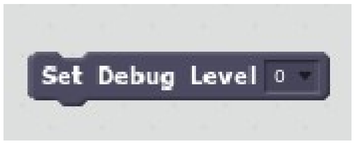
3.5. Reporter Blocks


Board: 1 Digital Input on Pin 2
Board: 1 Analog Sensor Input on Pin 2

These blocks retrieve the latest digital or analog data for the board and pin selected. Data is fetched from a cache located in the in the browser extension. There is no communication between Xi4S and the servers when these blocks are executed.

The cache is updated when a server detects a change in input value. The server notifies Xi4S of the change, and Xi4S updates its cache.

3.6. Debug Level



The Debug Level block sets the debug level of Xi4S. Debug output can be viewed in the JavaScript Console of the browser you are using.

Level 0 is the quietest level, and the console messages become more verbose as you increase the level number.

Debug information is available for the servers as well. By default the their level is set to 0 (quietest level).

If you wish to see debug information in the server console window, you will need to change the command line of the .bat or .sh script.

So for example, if you wish to increase the debug level for XiDuino to a level 2, you would change the command line in xiduino.sh or xiduino.bat from:

```
sudo node ~/xiduino/xiserver.js ard
```

to

```
sudo node ~/xiduino/xiserver.js ard 2
```

The other servers work similarly.

4. Some Usage Tips

4.1. Keeping Track of Boards While Programming

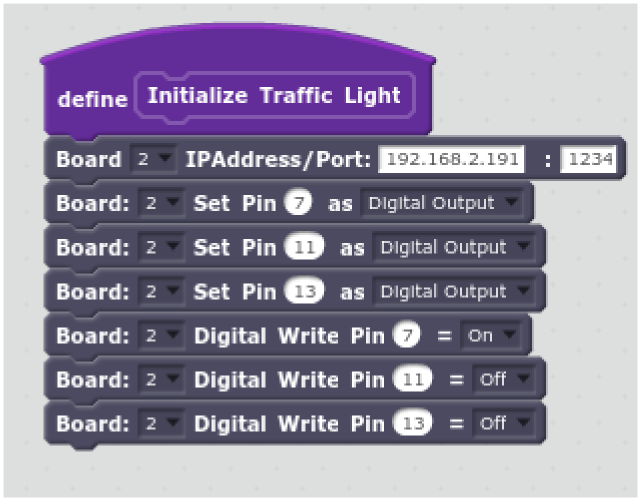
Keeping track of multiple boards when creating a Scratch project, can easily lead to confusion. To keep things simple, I suggest creating a separate Sprite for each board connected to Scratch. By naming each Sprite as `BoardName-BoardNumber`, you can easily distinguish between the boards. So for example, if you connect a Raspberry Pi as board #1 and an Arduino as board #2, the Sprite names for these boards might be RPI-1 and ARD-2.

4.2. Using “More” Blocks for a Higher Level Abstraction of the Boards and Their Connected Devices

The Xi4S block set was purposefully designed to represent boards at the pin level. This is appropriate for many applications, but may not desirable in all cases.

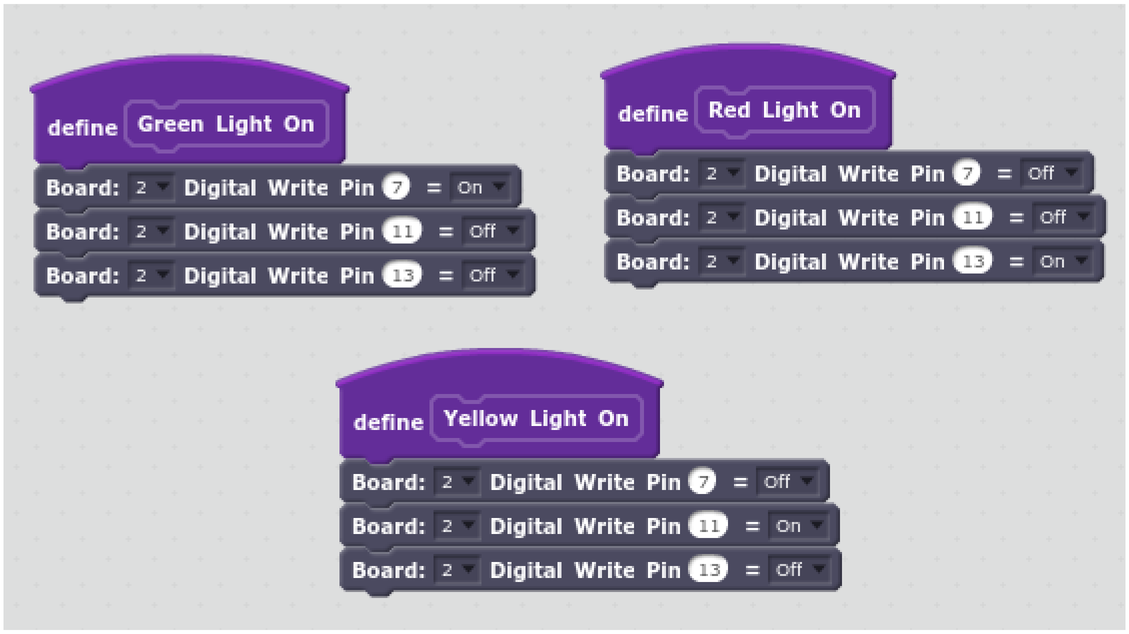
For example, you may wish to have a group of students control a “traffic light” that is assembled using a green, yellow and red LED connected to a Raspberry Pi. By utilizing “More” blocks you can give these students a view of the hardware without them having to know the low level details.

In the screen shot below, a block was assembled to create and initialize a “traffic light”. It sets up all the pins for the LEDs and turns the Green LED on.



```
define Initialize Traffic Light
Board 2 IPAddress/Port: 192.168.2.191 : 1234
Board: 2 Set Pin 7 as Digital Output
Board: 2 Set Pin 11 as Digital Output
Board: 2 Set Pin 13 as Digital Output
Board: 2 Digital Write Pin 7 = On
Board: 2 Digital Write Pin 11 = Off
Board: 2 Digital Write Pin 13 = Off
```

Additional blocks were assembled to turn on a single light while turning off the other two.

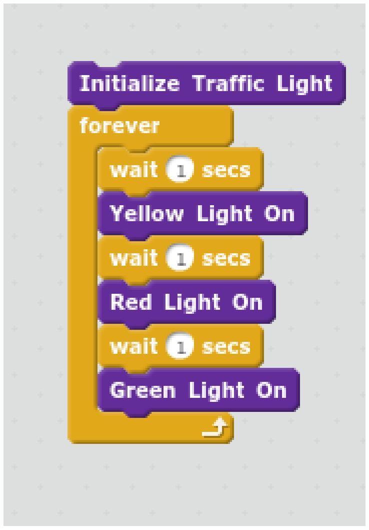


```
define Green Light On
Board: 2 Digital Write Pin 7 = On
Board: 2 Digital Write Pin 11 = Off
Board: 2 Digital Write Pin 13 = Off
```

```
define Red Light On
Board: 2 Digital Write Pin 7 = Off
Board: 2 Digital Write Pin 11 = Off
Board: 2 Digital Write Pin 13 = On
```

```
define Yellow Light On
Board: 2 Digital Write Pin 7 = Off
Board: 2 Digital Write Pin 11 = On
Board: 2 Digital Write Pin 13 = Off
```

A short script was written to cycle through changing of the traffic signals.



```
Initialize Traffic Light
forever
    wait 1 secs
    Yellow Light On
    wait 1 secs
    Red Light On
    wait 1 secs
    Green Light On
```

Providing this level of abstraction hides the hardware details and allows users to more readily solve the problem at hand.

In addition, by employing this technique, Xi may be used in conjunction with commercial products that have sensors and actuators that are hardwired to specific pins. By creating "More" blocks for these devices, the user can enjoy a plug and play experience, while still having the opportunity to get down to the pin level if so desired.

5. Getting Help

If you have any questions, suggestions, or encounter any problems, please don't hesitate to contact me at:

MisterYsLab@gmail.com

Alan Yorinks

6. Acknowledgements

I wanted to say thanks to my wife, who puts up with my "code and hardware tinkering".

Also, I wanted to say thanks to Rick Waldron, the author of the Johnny-Five library, for his help, advice and encouragement. Without his well written, excellently commented code I could not have written Xi in such a short time frame (proof of concept was written in less than 10 days). I am forever indebted to him for his gentle "nudges" towards the world of event-driven, non-blocking JavaScript programming.