

Introduction

The *MyDoorOpener* project has attracted a surprisingly high number of enthusiasts over the years. Unfortunately, until today, the project accessibility was rather limited to experienced electronics hobbyists, due to the complexity involved in the hardware assembly portion of the project.

Today, this is about to change ... We received many requests to provide an easier way to build the project, particularly from people with a more limited knowledge of general electronics.

This doesn't mean you don't need to know anything about electronics ... You still need to have some basic understanding. What it does mean is that you can get around with a simplified components selection/purchase process. Also, you can set things up without lifting your soldering iron - you may not even own one for crying out loud! For many of you, this is great news, which makes the project a lot more accessible.

The simplified approach proposed in this document has the following pros and cons.

Pros
Single source for components purchase
Simplified components selection process
Simplified assembly procedure
No soldering required

Cons
Final assembly is not as robust as when soldering techniques are used
Must deal with limited industry available components, enclosure, etc









Components Supplier

The components for the simplified version of the project can be ordered directly from the following electronics parts suppliers (we are not affiliated with them in any way) :

Suppliers	
United States	http://www.robotshop.com/
Canada	http://www.robotshop.ca/
Europe	http://www.robotshop.eu/

Components List

Following is the list of components that should be ordered in order to build a single door version of the *MyDoorOpener* project :

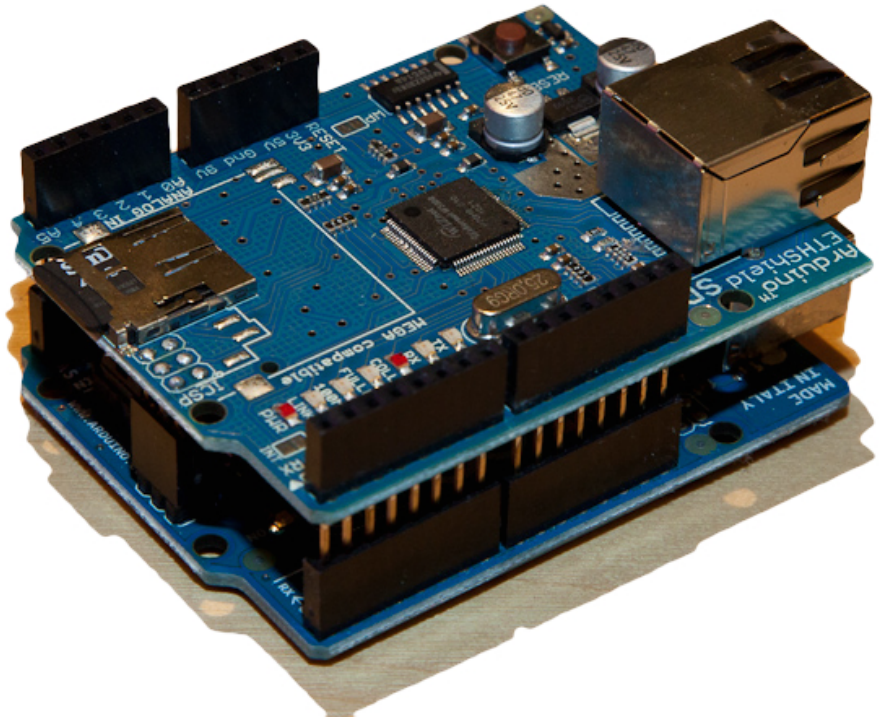
	Description	Qty	B/O	Price	Sub total
	SFE Arduino Project Enclosure Product code : RB-Spa-472	1	0	\$12.15	\$12.15
	Arduino Ethernet Shield Product code : RB-Ard-11	1	0	\$37.95	\$37.95
	DFRobot Single Relay Product code : RB-Dfr-35	1	0	\$10.45	\$10.45
	DFRobot Analog Sensor Cable 10pk Product code : RB-Dfr-73	1	0	\$6.18	\$6.18
	Arduino Uno USB Microcontroller Product code : RB-Ard-18	1	0	\$27.50	\$27.50
	Wall Adapter Power Supply - 9VDC 650mA Product code : RB-Spa-103	1	0	\$5.65	\$5.65
	Seedstudio Electronic Brick 3 Pin Terminal Module Product code : RB-See-58	1	0	\$3.03	\$3.03
	Seedstudio Electronic Brick 3 Wire Cable (5pk) Product code : RB-See-57	1	0	\$4.75	\$4.75

The enclosure is optional. All parts will fit within the enclosure, but we must say, it'll be a tight squeeze to get everything in there. If you're going for more than one door, this enclosure will not work.

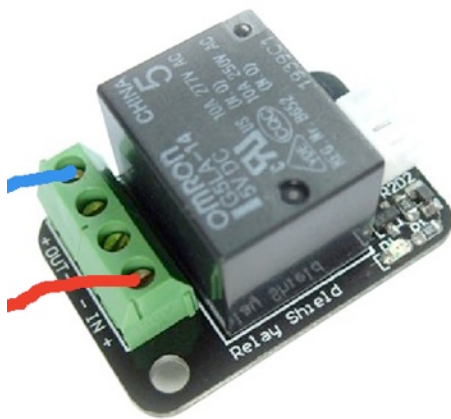
The prices above can give you an indication of rough costing for the project, but those prices may obviously change over time. Same applies for the actual part numbers and their availability. We can't guarantee anything, but we'll try our best to keep this document up to date (please report any discrepancy you find when purchasing your components).

Hardware Assembly

1. Snap the Arduino Ethernet Shield (RB-Ard-11) on top of the Arduino Uno Micro controller (RB-Ard-18).



2. Connect the DFRobot Single Relay (RB-Dfr-35) to your existing garage door opener wall button. This will require the installation of 2 low voltage wires (red and blue on the left illustration) going from the green terminal blocks labelled IN and OUT1, to your existing garage door wall mount button station.



When you open up your garage wall mount button station, you'll typically find 2 screws onto which you'll attach the other ends of the 2 low voltage wires (red and blue) coming from the relay block.

It doesn't matter which terminal from the wall mount button station gets linked with IN or OUT1, for as long as the entire setup can form a closed loop whenever the relay gets triggered.

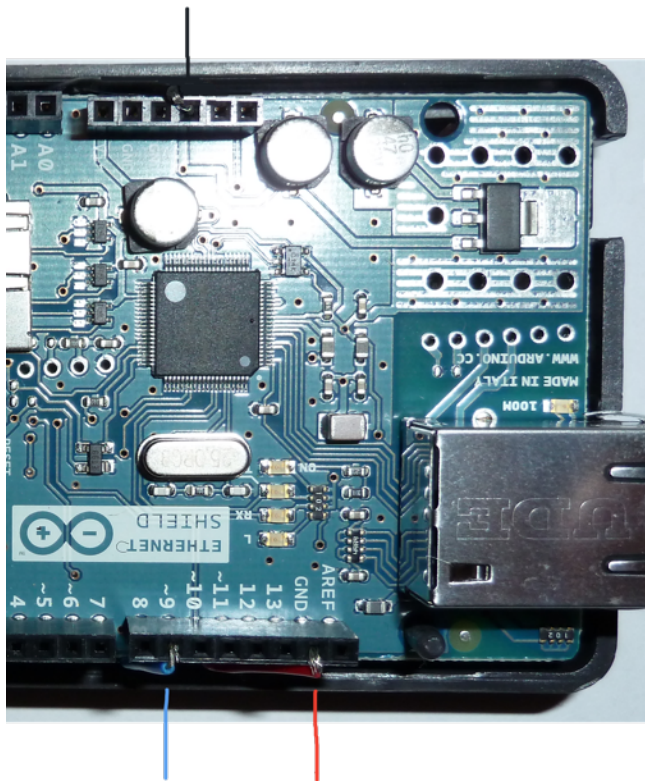
REF: [http://www.dfrobot.com/wiki/index.php?title=Relay_Module_\(Arduino_Compatible\)_SKU:_DFR0017](http://www.dfrobot.com/wiki/index.php?title=Relay_Module_(Arduino_Compatible)_SKU:_DFR0017)

3. Connect the DFRobot Single Relay (RB-Dfr-35) to the Arduino Ethernet Shield (RB-Ard-11). This will require the usage for one of the DFRobot Analog Sensor Cables (RB-Dfr-73).

You will need to cut off the black connector end of the cable, strip the plastic shielding on the end of each wire, and have them individually connected, as follows:



Relay Pin	Wire Color	Arduino Pin
Input (1)	Blue	Digital Pin 9
Power (2)	Black	5V
GND (3)	Red	GND

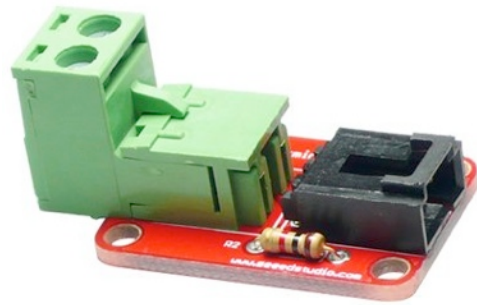


At this point in the assembly process, your Arduino Ethernet Shield (RB-Ard-11) should have 3 wires hooked up to it (black, blue and red), and should somewhat look like the picture on the left.

Make sure you have any excess wire tucked away nicely along the edges or even underneath the boards so that the enclosure can be properly closed.

There's no soldering here so make sure each wire end is tightly secured within its respective connector socket to avoid undesired or unexpected disconnections.

4. Connect the Seeedstudio 3 Pin Terminal Module (RB-See-58) to the Arduino Ethernet Shield (RB-Ard-11). This will require the usage for one of the Seeestudio Electronic Brick 3 Wire Cables (RB-See-57).



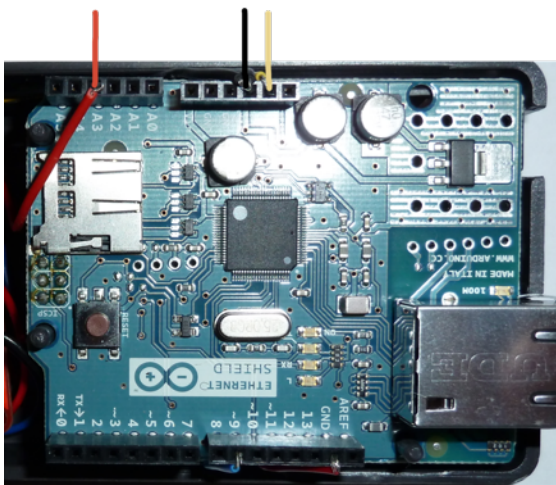
Once again, you will need to cut off one end of the cable, strip the plastic shielding, and individually connect each of the wires as follows:

Module Pin	Wire Color	Arduino Pin
Signal	Black	+5v
Vcc	Red	Analog Pin 3
GND	Yellow	+3v



This portion of the assembly will allow for status monitoring of your garage door. Is the door opened or is it closed ? How it works is that the Arduino will flow +3 volts (yellow wire) and +5 volts (black wire) to your status sensing device. Your status sensing device is expected to flow back on analog pin #3 (red wire), either +3 volts when the door is closed or +5 volts when the door is opened. This is how MyDoorOpener will determine whether the door is opened or closed, based on the voltage that flows back to analog pin #3. How this is achieved is beyond the scope of this document as many strategies are possible, depending on which type of sensing device you will be using (alarm panel, door sensor, etc). All that matters is that you respect the +3v (door is closed) and +5v (door is opened) protocol.

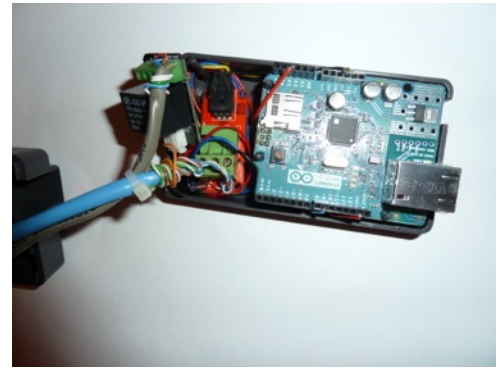
Use the green connecting block terminals to connect your status sensing device.



At this point in the assembly process, your Arduino Ethernet Shield (RB-Ard-11) should have 3 additional wires hooked up to it (black, yellow and red), and should somewhat look like the picture on the left.

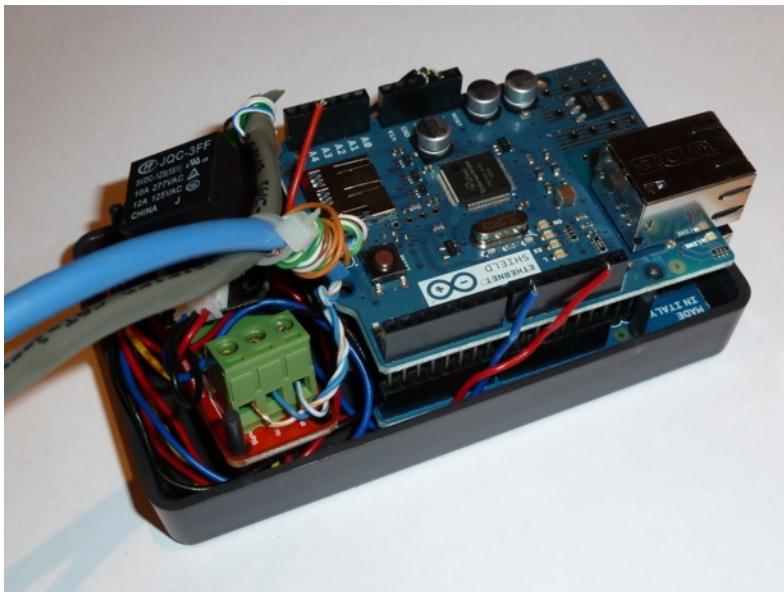
5. Place the Arduino Uno Micro controller (RB-Ard-18) and the Arduino Ethernet Shield (RB-Ard-11) assembly inside the enclosure (RB-Spa-472).

Make sure you have any excess wire tucked away nicely along the edges or even underneath the boards so that the enclosure can be properly closed.



Place the relay block (RB-Dfr-35) and terminal module (RB-See-58) at the rear end of the enclosure. Try to make everything fit as tightly as possible so that the enclosure can be properly closed. You can leverage the removable top cover to gain some extra mounting space and/or use that opening to have the wires running to your garage door and sensor.

Close the enclosure.



6. Connect the Arduino Ethernet Shield (RB-Ard-11) to your network using a standard RJ45 network cable and have it attached to your home network/router.
7. Connect the Arduino Uno Micro controller (RB-Ard-18) to the power supply (RB-Spa-103) and have it connected to an AC wall outlet.

This concludes the hardware assembly process. You can optionally use silicone to better secure each wire to its respective connector socket on the Arduino terminal blocks. If you do, make sure you test everything and confirm that it's properly working prior to doing so.

Software Configuration

1. Download the Arduino IDE software from the Arduino web site, based on the operating system you are using :

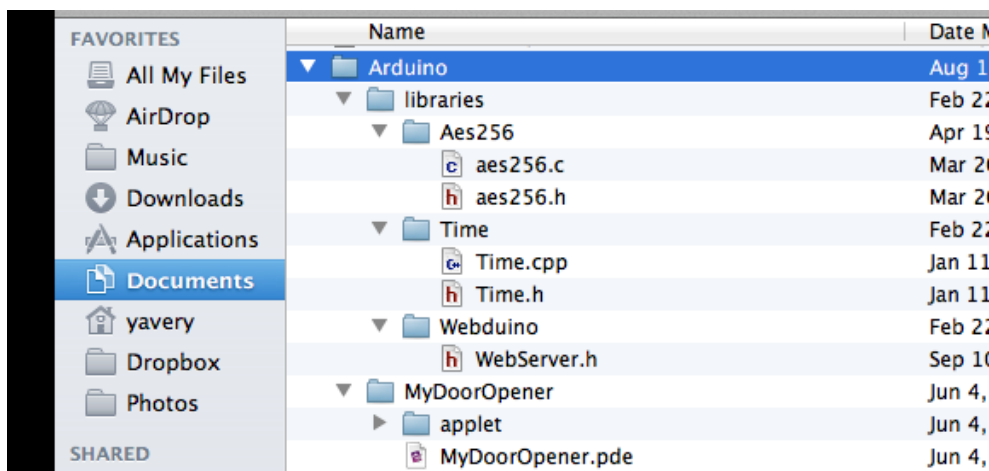
Operating System	Arduino Download
Windows	http://files.arduino.cc/downloads/arduino-0022.zip
Mac OSX	http://files.arduino.cc/downloads/arduino-0022.dmg
Linux 32bit	http://files.arduino.cc/downloads/arduino-0022.tgz
Linux 64bit	http://files.arduino.cc/downloads/arduino-0022-64-2.tgz

* Note that the version currently supported is **version 0022**. Using any other version of the Arduino IDE is not yet officially supported.

2. Download the MyDoorOpener project ZIP file :

<https://github.com/yavery/MyDoorOpener-Arduino/zipball/master>

3. Have the files extracted from the ZIP file so that the directory and file structure is similar to the following :



Respecting the location of libraries is important. If you don't, you will get compiler warnings and/or errors when compiling the project with the Arduino IDE.

- Open the MyDoorOpener.pde file inside the Arduino IDE. Edit the following lines to match your specific configuration (and save) :

Line #	Item	Description
~ 40	IP Address	Change to match the network IP address you want your Arduino to be accessible from. You will need to make sure this network IP address is reserved and never assigned to any other devices on your network.
~ 44	Password	Change to set the password you want to protect your Arduino with. This password will need to match with the password you will be entering in the iPhone application.

```

25
26 #include "Ethernet.h"
27 #include "WebServer.h"
28 #include "Time.h"
29 #include "aes256.h"
30
31 //*****
32 //*****
33
34 // EthernetShield MAC address.
35
36 static uint8_t mac[6] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
37
38 // EthernetShield IP address (DHCP reserved, never allocated to anyone else).
39
40 static uint8_t ip[4] = { 192, 168, 0, 13 };
41
42 // password required for operating door [max length = 16] (status fetching doesn't require password)
43
44 #define PASSWORD "mypassword"
45
46 //*****
47 //*****
48
49 // Arduino HTTP server listening port number.
50
51 WebServer webserver("", 80);
52

```

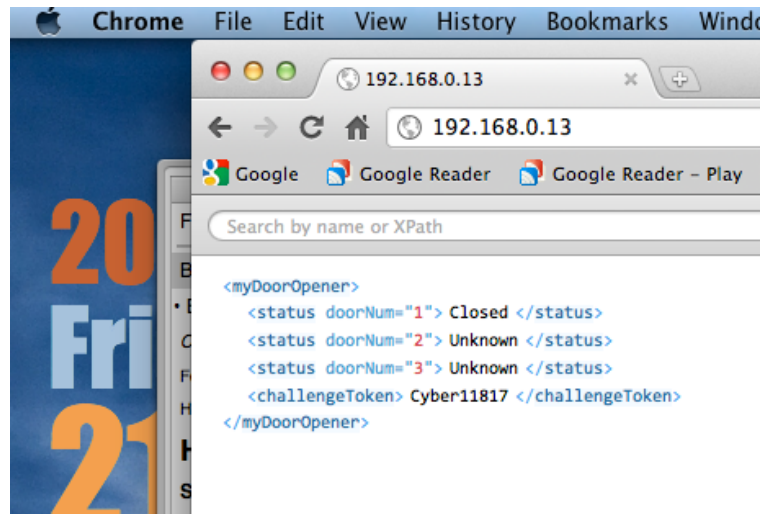
- Compile the MyDoorOpener.pde project from within the Arduino IDE, using the **Sketch -> Verify/Compile** menu option. Make sure there are no compiler warnings or errors.
- Connect your Arduino to your computer using a standard USB cable and upload the project, using the **File -> Upload to I/O Board** menu option from within the Arduino IDE.
- Once the upload is completed, you can unplug the Arduino from your computer. You will also need to either “reset” the Arduino (using the reset button on the board) or unplug the AC power from the Arduino for a few seconds. This will have for effect to have the MyDoorOpener program to be initiated.

This concludes the software configuration process.

Network Configuration

1. You should first test your installation by trying to access your Arduino with a web browser from within the same internal network. To do so, fire up your favorite web browser and type in the IP address you configured your Arduino to listen to, from step #4 of the software configuration process.

You should see a screen similar to the one on the right if things are working properly (your web browser must be able to display XML in order for this to work).



2. You should now configure your home router to do NAT port forwarding to your Arduino so that you can access it from the internet, using your home router's ISP assigned public IP address instead. This is required because the IP address assigned in step #4 of the software configuration process is an internal IP address which is not visible from the internet.

The Arduino is listening on port 80 (default) so you will need to input a NAT forwarding rule that will forward to that port, off the IP address you configured in step #4 of the software configuration process.

How NAT port forwarding works is specific from router to router, therefore it is beyond the scope of this document to cover these specifics. You can either look at your home router manual or Google around if you aren't sure how this can be achieved with your particular home router.

3. Another aspect you might want to look into is dynamic DNS. Since your home router most probably gets assigned an IP address by your ISP, it is subject to change at any time. In order to be shielded from such change, it is recommended to use a name rather than an IP address. The name will map to an IP address and with dynamic DNS services, that name to IP mapping will get maintained whenever your ISP allocates a new IP address to your home router.

Again, the details on how dynamic DNS works and the intricacies for the many dynamic DNS service providers out there is beyond the scope of this document. You can Google around to get more details on how such service works, the many service providers that offer this service, as well as how it can be configured with your particular home router.

This concludes the network configuration process.

iPhone Configuration

1. Download the iPhone application from the Apple iTunes store :

<http://itunes.apple.com/app/mydooropener/id359774310>

2. Once installed on your iPhone, open the application and click the small configuration gear located at the bottom right of the main screen.

3. Fill-in the configuration fields as follows :

Field Name	Description
URL	The URL used to connect to your Arduino from the internet. This is not the internal IP address for your Arduino, but a name or IP address that is visible from the internet. Typically, this will be a URL with the name or IP address of your home router on which you'll have configured a NAT forwarding rule that will forward to your Arduino. This URL must not include the port number. This URL must have the form http://xxx.yyy.zzz where xxx.yyy.zzz is either a DNS name or an IP address.
Port	The Port number to be used when connecting to your Arduino. Once again, this is the port on your home router, not the actual port on the Arduino (although they could be the same).
Password	The password for your Arduino, as configured in the software configuration process. This must match as-is, and is case sensitive.
Refresh Delay	Number of seconds to wait between automatic status refresh requests from the iPhone app to your Arduino.
Number of Doors	The number of doors you have configured on your Arduino. This is typically 1. You can also enter a value of 2 or 3 but this is a more advanced topic that is not covered in this document.

We hope this document was helpful in setting up your **MyDoorOpener** project.

If you have questions or if you find inaccuracies in this document, please drop us a note at support@mydooropener.com .

Thank you for using MyDoorOpener !