Yun Shield User Manual

VERSION: 1.1

<table>
<thead>
<tr>
<th>Version</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>Initiate</td>
<td>2014-Jun-21</td>
</tr>
<tr>
<td>1.0</td>
<td>Release</td>
<td>2014-Jul-08</td>
</tr>
<tr>
<td>1.1</td>
<td>Add USB mount description Add LEDs description, change default password to</td>
<td>2014-Aug-02</td>
</tr>
<tr>
<td></td>
<td>dragino</td>
<td></td>
</tr>
</tbody>
</table>
Index:

1 INTRODUCTION .................................................................................. 4
   1.1 What is Yun Shield ................................................................. 4
   1.2 Specifications ......................................................................... 4
   1.3 Features .................................................................................. 4
   1.4 LEDs ...................................................................................... 5
   1.5 System Structure ..................................................................... 6

2 CONFIGURE YUN SHIELD ................................................................. 8
   2.1 Find the ip addresses ............................................................. 8
   2.2 Configure Method ................................................................. 9
       2.2.1 Access via web interface ............................................... 9
       2.2.2 Access via SSH ........................................................... 9
   2.3 Web Configure Pages ........................................................... 10
       2.3.1 General Set Up ........................................................... 10
       2.3.2 Arduino related set up .................................................. 10
       2.3.3 Upgrade ....................................................................... 11

3 USE WITH ARDUINO BOARDS ..................................................... 12
   3.1 Connects to Arduino Boards ............................................... 12
       3.1.1 Connect to Leonardo .................................................... 12
       3.1.2 Connect to Arduino Uno ............................................ 12
       3.1.3 Connect to Arduino Duemilanove/Diecimila .................. 14
       3.1.4 Connect to Arduino Mega2560 .................................. 14
   3.2 Detect Yun Shield ................................................................. 15
   3.3 Upload Sketch ....................................................................... 17
   3.4 Bridge Library ....................................................................... 17

4 EXAMPLES ....................................................................................... 18
   4.1 Example 1: Say hello to Linux ............................................. 18
   4.2 Example 2: Upload data to IoT Server ................................. 20
   4.3 Example 3: Log Data to USB flash ...................................... 23
   4.4 Example 4: Use MQTT ........................................................ 25

5 FAQ ............................................................................................... 26
   5.1 What is the difference between the official Arduino Yun and Yun Shield? ...... 26
   5.2 Is Yun Shield compatible with a variant Arduino Board? ................. 26
   5.3 How to set up /www/sd and /mnt/sd? .................................... 27
   5.4 Arduino IDE doesn’t detect Yun Shield ................................... 27
   5.5 Where can I find the source code of Yun Shield? ....................... 27
5.6 How to Upload, Download or Edit files in Yun Shield ........................................ 27
5.7 How to reset the Yun Shield? ............................................................................. 29
5.8 How to recover the Yun Shield in case firmware crash ..................................... 29

6 REFERENCE ............................................................................................................. 32
1 INTRODUCTION

1.1 What is Yun Shield

Yun Shield is one of the most powerful shields for Arduino Board. Yun Shield is designed to solve the Internet connectivity and storage issue for Arduino Board.

Yun Shield runs Open Source OpenWrt system (Same system as runs in Arduino Yun) and it is fully compatible with Arduino IDE v1.5.4 or later. Yun Shield is the ideally choice for Arduino Projects which require various internet connections and more storage.

Basically, Yun Shield + Leonardo equally to the official Arduino Yun, but Yun Shield is more flexible because it can work with other Arduino board such as Uno, Duemilanove, Mega etc. And Yun Shield uses external wifi antenna which provides stability and possibility for various environments.

1.2 Specifications

- Processor: 400MHz, 24K MIPS
- Flash: 16MBytes
- RAM: 64MBytes
- Power Input: 4.75v ~ 23v via Arduino VIN pin
- 1 x 10M/100M RJ45 connector
- 150M WiFi 802.11 b/g/n
- External Antenna via I-Pex connector
- 1 x USB 2.0 host connector, used for USB storage or 3G connection
- 1 x Reset button
- Compatible with 3.3v or 5v I/O Arduino.

1.3 Features

- Open source Linux (OpenWrt) inside
Low power consumption
Compatible with Arduino IDE 1.5.4 or later, user can program, debug or upload sketch to Arduino board via Arduino IDE.
Managed by Web GUI, SSH via LAN or WiFi
Software upgradable via network
Built-in web server
Support internet connection via LAN port, WiFi or 3G dongle.
Support USB flash to provide storage for Arduino projects.
Failsafe design provides robustly system.
Compatible with Arduino Leonardo, Uno, Due, Dueilanove, Diecimila, Mega

1.4 LEDs
There are four LEDs on the Yun Shield. The functions of these LEDs are:

PWR: Power Indicate LED. Turn on once there is power.
LAN: Indicate there is LAN connection when it is on or blinking.
WLAN: Indicate WiFi status.
SYS: LED for USB storage. It is on if the USB flash are link to Arduino Yun default SD directory /mnt/sd and /www/sd
1.5 System Structure

**POWER:**
The Dragino HE is the core module of Yun Shield. The HE module requires around 200ma current when in full load, so it is powered by the Arduino **VIN** pins to avoid overheated in the Arduino onboard 5v LDO. So when Yun shield is in used, the Arduino board should be powered by DC port instead of USB port. The DC input can be **7v ~ 15v**.
The USB Host of Yun Shield gets power from the Arduino +5v pin, since the +5v from Arduino comes from the +5V LDO, to avoid overheated on the Arduino Board, when the USB host is in used, it is recommended to use +7v DC.

**Interface:**
The RJ45, WiFi, USB Host and Failsafe are connected to the Dragino HE module directly. And the Dragino HE module use SPI and UART to communicate with Arduino Board. Yun Shield is compatible with 3.3v and 5v Arduino board. The *on board jumper SV1* is used to set the SPI and UART to 3.3v or 5v level.

The SPI interface is used to upload the sketches comes from the Arduino IDE. SPI interface only connects to Dragino HE during uploading so the Arduino SPI can still be used to connect to other SPI slave devices.

The UART interface is used for the Bridge class in Arduino, there are lots of examples explain how to use the bridge class in the Arduino IDE. It is the core of Yun solution. We must make sure the serial Interface of Arduino is not used by other hardware.
2 CONFIGURE YUN SHIELD

2.1 Find the ip addresses
The Yun Shield has a WiFi interface and a LAN port. Either of them has IP address, can be used for internet connection and device management.

**Factory IP of WiFi port**

At the first boot of Yun Shield, it will auto generate an unsecure WiFi network call *Dragino2-xxxxxx*

User can use their laptop to connect to this WiFi network. The laptop will get an IP 192.168.240.xxx and the Yun Shield has the default IP 192.168.240.1

**Fall Back IP**

A fall back IP 172.31.255.254/255.255.255.252 is assigned to Yun Shield’s LAN port so user can always access Yun Shield with this ip if their laptop has the IP 172.31.255.253/255.255.255.252.

**Detect IP from Arduino IDE**

If Yun Shield’s Ethernet port is connected to the uplink router or the WiFi interface is associated to the WiFi router. The PC in the same network can use Arduino IDE to detect Yun Shield’s IP address as described in *Detected Yun Shield*. 
2.2 Configure Method
The Yun Shield runs Open Source Linux system. If user has a PC at the same network as Yun Shield, user can access its system via either Web Interface or Secure Shell (SSH).

2.2.1 Access via web interface
The recommended browsers to configure Yun Shield are Firefox and Chrome. Simply type the IP address into your browser and you will see the log in page of Yun Shield.

Default User name and Password for Yun Shield is root/dragino.

2.2.2 Access via SSH
Via SSH access, user can access to the Linux system directly and customized the system to support more features and applications.

SSH Access:
IP address: IP Address of Yun Shield
Port: 22
User Name: root
Password: dragino (default)
2.3 Web Configure Pages

2.3.1 General Set Up
After log in, the GUI will show the WIFI / ETH interface status. Click the Configure button and now user can configure the device password and network parameters.

2.3.2 Arduino related set up

**Arduino Board Type**: Define the bootloader/mcu type/ fuse setting during Sketch upload. **Operation Mode**: make sure it is in Arduino Bridge mode so the Bridge class for Arduino Yun works.
2.3.3 Upgrade
Yun Shield firmware can be upgraded via the GUI for bug fixes / system improvement or new features.

Go to GUI ➔ Upgrade page and select the correct firmware to upgrade. The firmware used for web upgrade should be a sysupgrade type firmware, user can choose if keep settings or not after upgrade.

Normally it will take about 2 minutes to flash the new firmware. Then all the LEDS will blink together which indicates that the system reboot with the new firmware.

The firmware version info can be check from this link: Yun Firmware Change log
3 USE WITH ARDUINO BOARDS

The Yun Shield use SPI for uploading sketch and use UART port for Bridge class to talk to the AVR. While connects Yun Shield to Arduino Board, below points should be checked:

- Whether the Arduino Board is power by DC jack
- If the board type setting is correct in Yun Shield.
- If the board type setting is correct in Arduino IDE
- Whether the Arduino SPI and UART is not influenced by other hardware
- Make sure the UART mode is in Arduino Bridge mode.

3.1 Connects to Arduino Boards

3.1.1 Connect to Leonardo

Simply plug the Yun Shield on top of the Leonardo, and power the Leonardo via DC jack.

In Arduino IDE, the board type should select Arduino Yun.

3.1.2 Connect to Arduino Uno

1) In UNO, the uart connection between mega328P and mega16u2 will influence the bridge feature with Yun Shield. So we have to disconnect it by set the mega16u2 into reset mode. As below:

![Image of Arduino Uno connection](https://www.dragino.com/images/arduino_uno_connection.png)

Shortcut these two pins to put mega16u2 in reset mode

Note: USB upgrade/debug won’t work after this change, User will have to upgrade sketch and debug via Arduino IDE via WiFi (see examples)

2) Add a “Uno Yun” board type in the file: Arduino\hardware\arduino\avr\board.txt. as below and reopen the Arduino IDE:
3) Put the Yun Shield on top of Uno and power it via DC jack.
3.1.3 Connect to Arduino Duemilanove/Diecimila
1) In Duemilanove/Diecimila, the mega avr uart interface is connected to the FTDI chip, we have to disconnect them as shown in below picture:

![Image of Duemilanove with UART connection highlighted]

2) Add a “Duemilanove Yun” board type in the file: Arduino\hardware\arduino\avr\board.txt. user can use the UnoYun board type if they has mega328p. for mega328/mega168/mega168p, they can modify the upload.maximum_data_size/ upload.maximum_size and build.mcu accordingly.

3) Put the Yun Shield on top of Duemilanove and power it via DC jack.

3.1.4 Connect to Arduino Mega2560
1) In Mega2560, the uart connection between mega2560 and mega16u2 will influence the bridge feature with Yun Shield. So we have to disconnect it by set the mega16u2 into reset mode. As below:

![Image of Mega2560 with UART connection highlighted]

2) Add a “Mega2560 Yun” board type in the file: Arduino\hardware\arduino\avr\board.txt. as below and reopen the Arduino IDE:

##############################################################
### 3) Put the Yun Shield on top of Mega2560 and power it via DC jack.

#### 3.2 Detect Yun Shield

Make sure your laptop and Yun Shield are in the same network. The Yun Shield will broadcast data in this network and the Arduino IDE will receive this data and show the Yun Shield in **Tools → Port.**
3.3 Upload Sketch

1) In the Arduino IDE, choose the correct board type for the AVR module.
2) In Arduino IDE → port, choose the correct port. (should be Arduino Yun port with an ip address)
3) In the Yun Shield GUI → Sensor page, choose the correct board type for upload.
4) Compile the sketch and upload it to the Arduino Board. During upload, The Yun Shield will ask you to key in the password, by default, the password is dragino.

3.4 Bridge Library

The Bridge Library simplifies the communication between the Arduino Board and Yun Shield.

Bridge commands from the AVR (Arduino Board) are interpreted by Python on the Yun Shield. Its role is to execute programs on the GNU/Linux side when asked by Arduino, provide a shared storage space for sharing data like sensor readings between the Arduino and the Internet, and receiving commands from the Internet and passing them directly to the Arduino.

There are detail explain and lots of example to show how to use Bridge in the Arduino Official Website. Reference link is: http://arduino.cc/en/Reference/YunBridgeLibrary
4  EXAMPLES

4.1  Example 1: Say hello to Linux

**Introduction:**
This example is a hello test between the Arduino and Yun Shield. The example can be found on the Arduino IDE -> File -> Examples -> Bridge -> ConsoleRead. Tutorial of this example can be found on http://arduino.cc/en/Tutorial/ConsoleRead. Below listing the code and add some detail to understand it with the Yun Shield:

**Code:**
```c
#include <Console.h> //use Console class for Arduino IDE debug over WiFi, similar to Serial class,
String name;

void setup() {
    // Initialize Console and wait for port to open:
    Bridge.begin();
    Console.begin();

    // Wait for Console port to connect
    while (!Console);
    Console.println("Hi, what's your name?"); //Data flow: Arduino --> Yun Shield --> Arduino IDE
}

void loop() {
    if (Console.available() > 0) {
        char c = Console.read(); //read the next char received, data flow: IDE --> Yun Shield --> Arduino
        // look for the newline character, this is the last character in the string
        if (c == '\n') {
            //print text with the name received
            Console.print("Hi ");
            Console.println(name);
            Console.println("! Nice to meet you!");
            Console.println();
            // Ask again for name and clear the old name
            Console.println("Hi, what's your name?");
            name = ""; // clear the name string
        }
        else { // if the buffer is empty Console.read() returns -1
            name += c; // append the read char from Console to the name string
        }
    }
}
```

**Screen Shot:**
Console.print("Hi, what's your name?");

void loop() {
  if (Console.available() > 0) {
    char c = Console.read(); // read the next char received
    // look for the newline character, this is the last char received
    if (c == '\n') {
      // print text with the name received
      Console.print("Hi ");
      Console.print("Hi to meet you!");
      Console.print("Hi, what's your name?");
      new = ""; // clear the name string
    } else if (c == ' ' && new.length() == 0) {
      // if the buffer is empty Console.read() reads a blank
      new = c; // append the read char from Console to the name buffer
    }
  }
}

//
4.2 Example 2: Upload data to IoT Server

**Introduction:**
This example shows how to log data to the public IoT server “Xively”. The example is a modified version (change Serial to Console to fit for different Arduino Board and debug over WiFi) from Arduino IDE --> File --> Examples --> Bridge --> XivelyClient. Tutorial of this example can refer http://arduino.cc/en/Tutorial/YunXivelyClient.

Before upload the sketch, make sure:
- The Yun Shield already has internet access
- Input your FEED ID and API KEY according to the Tutorial. Note, The FEED ID should be within double quotation marks “”.
- Change Serial Class to Console class to fit for different AVRs.

Below listing the code and add some detail to understand it with the Yun Shield:

**Code:**
```
#include <Process.h>  // Process lib use to call Linux Commands in Yun Shield
#include <Console.h>  // Console lib, used to show debug info in Arduino IDE
#include "passwords.h"  // contains my passwords, see below

/*
NOTE: passwords.h is not included with this repo because it contains my passwords.
You need to create it for your own version of this application. To do so, make
a new tab in Arduino, call it passwords.h, and include the following variables and constants:

#define APIKEY "foo"                  // replace your pachube api key here
#define FEEDID "0000"                 // replace your feed ID
#define USERAGENT "my-project"       // user agent is the project name
*/

// set up net client info:
const unsigned long postingInterval = 60000;  // delay between updates to xively.com
unsigned long lastRequest = 0;  // when you last made a request
String dataString = "";

void setup() {
  // start console:
  Bridge.begin();
  Console.begin();

  while (!Console);  // wait for Network Serial to open
  Console.println("Xively client");

  // Do a first update immediately
  updateData();
  sendData();
  lastRequest = millis();
}

void loop() {
  Yun Shield User Manual 20 / 32
// get a timestamp so you can calculate reading and sending intervals:
long now = millis();

// if the sending interval has passed since your
// last connection, then connect again and send data:
if (now - lastRequest >= postingInterval) {
    updateData();
    sendData();
    lastRequest = now;
}

void updateData() {
    // convert the readings to a String to send it:
    dataString = "Temperature,;"
    dataString += random(10) + 20;
    // add pressure:
    dataString += 
    dataString += random(5) + 100;
}

// this method makes a HTTP connection to the server:
void sendData() {
    // form the string for the API header parameter:
    String apiString = "X-ApiKey: ");
    apiString += APIKEY;
    // form the string for the URL parameter:
    String url = "https://api.xively.com/v2/feeds/"
    url += FEEDID;
    url += ",.csv";
    // Send the HTTP PUT request, form the linux command and use Process Class to send this command to Yun
    // Is better to declare the Process here, so when the
    // sendData function finishes the resources are immediately
    // released. Declaring it global works too, BTW.
    Process xively;
    xively.begin("curl");
    xively.addParameter("-k");
    xively.addParameter("--request");
    xively.addParameter("PUT");
    xively.addParameter("--data");
    xively.addParameter(dataString);
    xively.addParameter("--header");
    xively.addParameter(apiString);
    xively.addParameter(url);
    xively.run();
    Console.println("done!");

    // If there's incoming data from the net connection,
    // send it out the Console:
    while (xively.available() > 0) {
        char c = xively.read();
        Console.write(c);
    }
}
Screen Shot:

- Xively Dashboard

Triggers provide ‘push’ capabilities by sending HTTP POST requests to a URL of your choice when a condition has been satisfied.
4.3 Example 3: Log Data to USB flash

Introduction:
This example shows how to log data to a USB flash. The sketch used in this example is same as http://wiki.dragino.com/index.php?title=Arduino_Yun_examples#Log_sensor_data_to_USB_flash. And the source code is in this link.

The Yun Shield will auto mount the USB flash to directory /mnt/sda1. And the sketch will append the sensor data to the file /mnt/sda1/data/datalog.csv. So make sure there is such file in the USB flash before running the sketch.

Code:
```cpp
#include <FileIO.h>  // FileIO class allow user to operate Linux file system
#include <Console.h> // Console class provide the interactive between IDE and Yun Shield

void setup() {
    // Initialize the Console
    Bridge.begin();
    Console.begin();
    FileSystem.begin();

    while(!Console);  // wait for Serial port to connect.
    Console.println("Filesystem datalogger\n");
}

void loop () {
    // make a string that start with a timestamp for assembling the data to log:
    String dataString;
    dataString += getTimeStamp();
    dataString += " , ";

    // read three sensors and append to the string:
    for (int analogPin = 0; analogPin < 3; analogPin++) {
        int sensor = analogRead(analogPin);
        dataString += String(sensor);
        if (analogPin < 2) {
            dataString += ",";  // separate the values with a comma
        }
    }

    // open the file. note that only one file can be open at a time,
    // so you have to close this one before opening another.
    // The USB flash card is mounted at "/mnt/sda1" by default
    File dataFile = FileSystem.open("/mnt/sda1/data/datalog.csv", FILE_APPEND);

    // if the file is available, write to it:
    if (dataFile) {
        dataFile.println(dataString);
        dataFile.close();
        // print to the serial port too:
        Console.println(dataString);
    }
    // if the file isn't open, pop up an error:
    else {
```
```
```java
Console.println("error opening datalog.csv");

delay(15000);  //Write every 15 seconds

// getTimeStamp function return a string with the time stamp
// Yun Shield will call the Linux "date" command and get the time stamp
String getTimeStamp() {
    String result;
    Process time;
    // date is a command line utility to get the date and the time
    // in different formats depending on the additional parameter
    time.begin("date");
    time.addParameter("+%D-%T");  // parameters: D for the complete date mm/dd/yy
    // T for the time hh:mm:ss
    time.run();  // run the command

    // read the output of the command
    while(time.available()>0) {
        char c = time.read();
        if(c != '\n')
            result += c;
    }

    return result;
}
```

**Screen Shot:**
4.4 Example 4: Use MQTT

**Introduction:**
MQTT is a machine-to-machine (M2M)/"Internet of Things" connectivity protocol. It was designed as an extremely lightweight publish/subscribe messaging transport. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium. For example, it has been used in sensors communicating to a broker via satellite link, over occasional dial-up connections with healthcare providers, and in a range of home automation and small device scenarios. It is also ideal for mobile applications because of its small size, low power usage, minimised data packets, and efficient distribution of information to one or many receivers.

This example shows how to use Yun Shield with SSL MQTT protocol.

**Install required packages**
Install below packages for MQTT:

```
root@Arduino:~# opkg update
root@Arduino:~# opkg install libcares
root@Arduino:~# opkg install libopenssl
root@Arduino:~# opkg install libmosquitto
root@Arduino:~# opkg install mosquitto-client
```

**Test with public MQTT test server:**
test.mosquitto.org is a hosts a publicly available Mosquitto MQTT server/broker. User can post and subscribe to this MQTT server for testing. To test this server with SSL connection, user has to download the certificate authority file `mosquitto.org.crt` to verify the server connection.

Open two SSH windows. In the first window, we subscribe to a topic with below command:

```
mosquitto_sub -h test.mosquitto.org -p 8883 --cafile mosquitto.org.crt -t dragino/device0/temperature
```

This command will listen to this topic and display any change on the server.

And in the other window, we publish the change by another command:

```
mosquitto_pub -h test.mosquitto.org -p 8883 --cafile mosquitto.org.crt -t dragino/device0/temperature -m 31
```

Once the publish command is issued, we can see the result will display in the subscribe command.
5 FAQ

5.1 What is the difference between the official Arduino Yun and Yun Shield?

In Hardware Aspect

Both Arduino Yun and Yun Shield have the same CPU, Memory size and RAM size for the Linux system. The Arduino Yun is an integrated of Linux Part and MCU part, Yun Shield is designed as a shield which can be used with exist Arduino boards.

Basically, The Yun Shield + Arduino Leonardo equally to an Arduino Yun, but Yun Shield is more flexible because it can be used with other Arduino boards such as Arduino Uno, Duemilanove, Diecimila etc.

Yun Shield is duplicable and producible: The design of Yun Shield is open and the most complicate and difficult parts are done in the Dragino HE module. User can purchase the Dragino HE module separately to customized their IoT project and release their variant Yun Solution.

Stable and Flexible WiFi performance: Arduino Yun use chip antenna design, if there is a shield on top of the Arduino Yun, the wifi will be greatly shielded and lead to a poor wifi performance. Instead, Yun Shield use external Antenna design, user can connect different type of antennas to the i-pex connector of Yun Shield, this make the installation is more flexible and possible to transfer the signal to several km distance.

In Software Aspect

The Yun Shield software is derived from Arduino Yun with some bugs fixed; feature added and support more board types.

5.2 Is Yun Shield compatible with a variant Arduino Board?

If the Arduino board is a variant from the boards described in Support Board Type, then it should be compatible. Below is the check list for the compatibility.

- The variant has 7~15v power in the VIN pin to power the Yun Shield.
- The variant has same definition and position of SPI pins in the ICSP header as the official board.
- The variant has same definition and position of D0 and D1 pins in the ICSP header in the official board.
- Check whether there are ICs connected to the SPI and UART of the AVR and evaluate if they will influence the communication between Yun Shield and the AVR MCU.

The system structure section well explains the working principle of Yun Shield, if user still not sure if Yun Shield is compatible with their board or having trouble in the compatibility. Then can send the board info to support@dragino.com and our support team will review and check it.
5.3 How to set up /www/sd and /mnt/sd?
To use the /www/sd and /mnt/sd as the same as Arduino Yun, user can prepare a USB flash and create directory /arduino and /arduino/www at the root of USB flash. Then input the USB flash into the Yun Shield and it will automatically create /www/sd and /mnt/sd and link to /arduino and /arduino/www.

5.4 Arduino IDE doesn’t detect Yun Shield
Check below points if this issue happens:
✓ The Arduino IDE version is 1.5.4 or later
✓ Your PC and Yun Shield are in the same network.
✓ If Yun Shield boot in advance than Arduino IDE, this may happen. So try to power off/on the Yun Shield and check again.

5.5 Where can I find the source code of Yun Shield?
The Yun Shield source can be found at: https://github.com/dragino/linino

5.6 How to Upload, Download or Edit files in Yun Shield
Yun Shield has a built-in Linux system and support SCP protocol. User can upload, download or edit the Linux files using SCP tools.

In windows OS, the scp tool is winscp. Install it and log into Yun Shield as below:
❖ Host Name: Yun Shield IP address
❖ User Name: root
❖ Password: arduino (default)
❖ Protocol: SCP

The log in process will alter two warning, just ignore it. After log in, a management panel will appear. The left part of this panel is your PC’s directories and the right part shows the directories of Yun Shield, you can upload/download files by dragging, or double click the files to modify its content.
5.7 How to reset the Yun Shield?
Yun Shield has a toggle button which can be used for reset. When the system of Yun Shield is running, user can press the toggle button to reset the device.

- If pressing the toggle button and release after 5 seconds, it will reset the WiFi setting and other settings will be kept.
- If pressing the toggle button and release after 30 seconds, it will reset ALL the settings to factory default.

5.8 How to recover the Yun Shield in case firmware crash
There are some cases that the Yun Shield fails to boot, for example upgrade an improper firmware or lost power during upgrade.
User is still able to recover the Yun Shield system by using the Failsafe u-boot of Yun Shield.

An instruction in Windows is as below:

Set up TFTP server
Download the tftp server (recommend tftp32d.exe). And download the latest Yun firmware from [http://www.dragino.com/downloads/index.php?dir=motherboards/ms14/Firmware/Yun/](http://www.dragino.com/downloads/index.php?dir=motherboards/ms14/Firmware/Yun/). The firmware we need is the `kernel` and `rootfs-squashfs` files. Put these firmware files and tftp32d at the same directory. Start the tftp server.

Download Hercules utility
Download Hercules, this is the tool we use to transfer commands to Yun Shield in Failsafe mode.
Run Hercules and input correct parameters as below:

<table>
<thead>
<tr>
<th>Protocol:</th>
<th>UDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module IP:</td>
<td>192.168.255.1</td>
</tr>
<tr>
<td>Port:</td>
<td>6666</td>
</tr>
<tr>
<td>Local port:</td>
<td>6666</td>
</tr>
</tbody>
</table>

Connect your PC to Yun Shield
Connect the PC and Yun Shield via an Ethernet cable. Set up PC with below LAN IP **192.168.255.2** and netmask **255.255.255.0**. Disable PC’s firewall.

Power up Yun Shield to Failsafe mode
Press the Failsafe button and power up Yun Shield; user will see all the LEDs blink together, release the button after 10 seconds and there are some messages will pop up in the Hercules panel, which means the Yun Shield has been in Failsafe Netconsole mode and ready to access commands.

User can type the commands in Hercules to transfer and upgrade Yun Shield to the latest firmware with factory settings.

The update commands are as below, replace the xxx with the actually version.

**Note when typing command in Hercules:** user must add `<CR>` at the end of each command; `$` is a special char in Hercules, user should double it (key two `$$`) when typing.

**Upgrade Kernel**

```
tftpboot 0x81000000 ms14-arduino-yun-kernel-xxx.bin
erase 0x9f0a0000 +0x140000
cp.b 0x81000000 0x9f0a0000 $filesize
```

**Upgrade rootfs**

```
tftpboot 0x81000000 ms14-arduino-yun--rootfs-squashfs-xxx.bin
erase 0x9f050000 +0xe50000
cp.b 0x81000000 0x9f050000 $filesize
```

**Reset to the new firmware**

```
reset
```

**Warning:** User should use exactly address number in the erase and cp.b shows, wrong address number may properly destroy the boot-loader of Yun Shield and the device won’t boot anymore. Or destroy the radio data of Yun Shield which may lead to a poor wifi performance or incorrect MAC addresses.

**Recover in Linux is similar with Windows,** the mail different is that the tool use in Linux is `nc` and runs with `nc -kul 6666`. Below shows that the Yun Shield has been in Failsafe Netconsole mode and detected by nc.
$ boot
  boot application image from memory
$ clearclocks
  remove FLL and clock configuration from FLASH
$ cp
  memory copy
$ dhcp
  invoke DHCP client to obtain IP/hostname
$ erase
  erase flash memory
$ eraseenv
  erase environment sector in flash
$ http
  start web server to address addr
$ help
  print embedded help
$ info
  start web server for firmware recovery
$ lninfo
  print firmware header
$ md
  memory display (auto-incrementing)
$ memtest
  simple RAM test
$ mem
  memory write (fill)
$ mem
  memory modify (constant address)
$ ping
  send ICMP ECHO REQUEST to network host
$ printenv
  print environment variables
$ printenv
  print MAC addresses stored in flash
$ reset
  perform REBOOT of the CPU
$ run
  run commands in an environment variable
$ savenv
  save environment variables to FLASH
$ setclocks
  select clock configuration from predefined list
$ setenv
  set environment variables
$ setipmem
  save new MAC address in flash
$ start
  start network
$ startnc
  start network
$ startnc
  start serial console
$ stopboot
  boot image via network using TFTP protocol
$ version
  print U-Boot version
6 REFERENCE

✧ Yun Shield hardware source
   https://github.com/dragino/modules/tree/master/hardware/YunShield

✧ Yun Shield software source code
   https://github.com/dragino/linino

✧ Arduino Yun Bridge Official page
   http://arduino.cc/en/Reference/YunBridgeLibrary

✧ Arduino Yun Official Forum
   http://forum.arduino.cc/index.php?board=93.0