Sarah Wood

Embedded Programming

Chapter 10

Assignment 2

• Search the Web for other Ethernet shield projects and build at least one

of them. A very ambitious project tries to implement a complete web

browser on the Arduino, for example.13

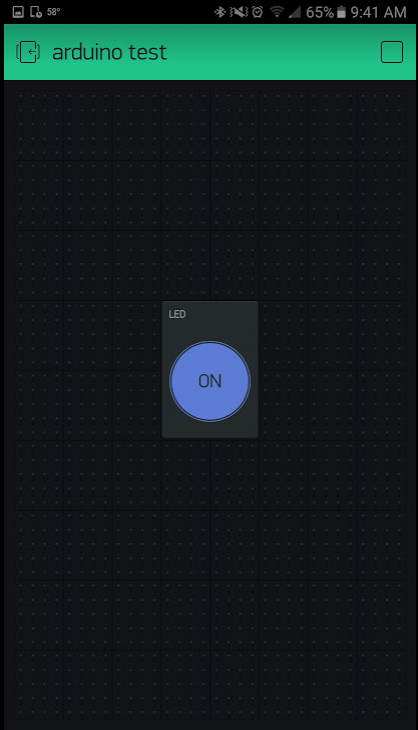
I selected a lighting controller

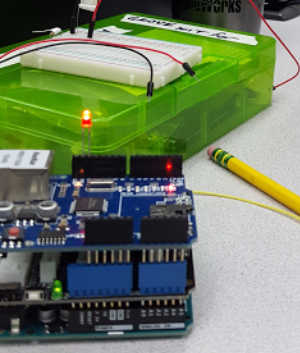
<http://www.blynk.cc/getting-started/>

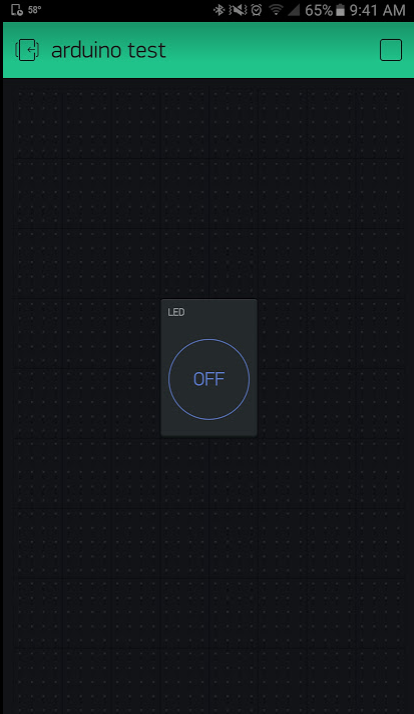
It communicates through an Android app to control a physical device (in this case, LED) hooked up to a networked Arduino. Output hardware could easily be altered. In the future, I may install an Arduino in my entryway to control an LED daisy-chain and a camera.

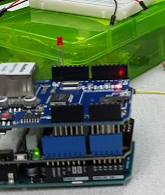
Screen Caps

I controlled the LED from my phone. A tiny bit lag-gy sometimes, but very cool nonetheless.

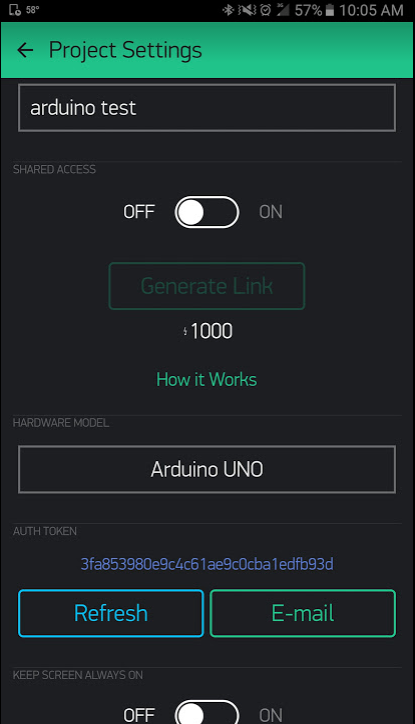


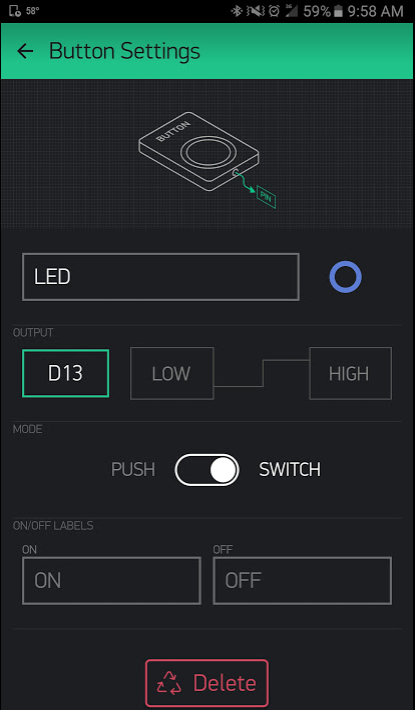






The interface is very easy to set up. Project setup is very easy. When you put a button into the phone app, you specify output device type, and pin #. You also set behavior type for the button.





I was very impressed with Blynk. Very simple to configure, and easy to use. I rooted around in the code, and it was easy to understand and well-organized.

Source Code

#define BLYNK\_PRINT Serial // Comment this out to disable prints and save space

#include <SPI.h>

#include <Ethernet.h>

#include <BlynkSimpleEthernet.h>

char auth[] = "**REDACTED**"; //You get an auth token on a per-project basis from Blynk

void setup()

{

Serial.begin(9600);

Blynk.begin(auth);

// You can also specify server.

// For more options, see Transports/Advanced/CustomEthernet example

//Blynk.begin(auth, "server.org", 8442);

//Blynk.begin(auth, IPAddress(192,168,1,100), 8888);

}

void loop()

{

Blynk.run();

}

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

BlynkEthernet.h (Blynk includes many headers, but this one does most of the heavy lifting)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

/\*\*

\* @file BlynkEthernet.h

\* @author Volodymyr Shymanskyy

\* @license This project is released under the MIT License (MIT)

\* @copyright Copyright (c) 2015 Volodymyr Shymanskyy

\* @date Jan 2015

\* @brief

\*

\*/

#ifndef BlynkEthernet\_h

#define BlynkEthernet\_h

#ifndef BLYNK\_INFO\_CONNECTION

#define BLYNK\_INFO\_CONNECTION "W5000"

#endif

#include <BlynkApiArduino.h>

#include <Blynk/BlynkProtocol.h>

#include <Adapters/BlynkArduinoClient.h>

static const byte \_blynkEthernetMac[] = { 0xDE, 0xED, 0xBA, 0xFE, 0xFE, 0xED };

class BlynkEthernet

: public BlynkProtocol<BlynkArduinoClient>

{

typedef BlynkProtocol<BlynkArduinoClient> Base;

public:

BlynkEthernet(BlynkArduinoClient& transp)

: Base(transp)

{}

void config(const char\* auth,

const char\* domain = BLYNK\_DEFAULT\_DOMAIN,

uint16\_t port = BLYNK\_DEFAULT\_PORT)

{

Base::begin(auth);

this->conn.begin(domain, port);

}

void config(const char\* auth,

IPAddress ip,

uint16\_t port = BLYNK\_DEFAULT\_PORT)

{

Base::begin(auth);

this->conn.begin(ip, port);

}

// DHCP with domain

void begin( const char\* auth,

const char\* domain = BLYNK\_DEFAULT\_DOMAIN,

uint16\_t port = BLYNK\_DEFAULT\_PORT,

const byte mac[] = \_blynkEthernetMac)

{

Base::begin(auth);

BLYNK\_LOG("Getting IP...");

if (!Ethernet.begin((byte\*)mac)) {

BLYNK\_FATAL("DHCP Failed!");

}

// give the Ethernet shield a second to initialize:

::delay(1000);

this->conn.begin(domain, port);

IPAddress myip = Ethernet.localIP();

BLYNK\_LOG("My IP: %d.%d.%d.%d", myip[0], myip[1], myip[2], myip[3]);

}

// Static IP with domain

void begin( const char\* auth,

const char\* domain,

uint16\_t port,

IPAddress local,

IPAddress dns,

const byte mac[]= \_blynkEthernetMac)

{

Base::begin(auth);

BLYNK\_LOG("Using static IP");

Ethernet.begin((byte\*)mac, local);

// give the Ethernet shield a second to initialize:

::delay(1000);

this->conn.begin(domain, port);

IPAddress myip = Ethernet.localIP();

BLYNK\_LOG("My IP: %d.%d.%d.%d", myip[0], myip[1], myip[2], myip[3]);

}

// Static IP with domain, gateway, etc

void begin( const char\* auth,

const char\* domain,

uint16\_t port,

IPAddress local,

IPAddress dns,

IPAddress gateway,

IPAddress subnet,

const byte mac[]= \_blynkEthernetMac)

{

Base::begin(auth);

BLYNK\_LOG("Using static IP");

Ethernet.begin((byte\*)mac, local, dns, gateway, subnet);

// give the Ethernet shield a second to initialize:

::delay(1000);

this->conn.begin(domain, port);

IPAddress myip = Ethernet.localIP();

BLYNK\_LOG("My IP: %d.%d.%d.%d", myip[0], myip[1], myip[2], myip[3]);

}

// DHCP with server IP

void begin( const char\* auth,

IPAddress addr,

uint16\_t port = BLYNK\_DEFAULT\_PORT,

const byte mac[] = \_blynkEthernetMac)

{

Base::begin(auth);

BLYNK\_LOG("Getting IP...");

if (!Ethernet.begin((byte\*)mac)) {

BLYNK\_FATAL("DHCP Failed!");

}

// give the Ethernet shield a second to initialize:

::delay(1000);

this->conn.begin(addr, port);

IPAddress myip = Ethernet.localIP();

BLYNK\_LOG("My IP: %d.%d.%d.%d", myip[0], myip[1], myip[2], myip[3]);

}

// Static IP with server IP

void begin( const char\* auth,

IPAddress addr,

uint16\_t port,

IPAddress local,

const byte mac[] = \_blynkEthernetMac)

{

BLYNK\_LOG("Using static IP");

Base::begin(auth);

Ethernet.begin((byte\*)mac, local);

// give the Ethernet shield a second to initialize:

::delay(1000);

this->conn.begin(addr, port);

IPAddress myip = Ethernet.localIP();

BLYNK\_LOG("My IP: %d.%d.%d.%d", myip[0], myip[1], myip[2], myip[3]);

}

// Static IP with server IP, DNS, gateway, etc

void begin( const char\* auth,

IPAddress addr,

uint16\_t port,

IPAddress local,

IPAddress dns,

IPAddress gateway,

IPAddress subnet,

const byte mac[] = \_blynkEthernetMac)

{

BLYNK\_LOG("Using static IP");

Base::begin(auth);

Ethernet.begin((byte\*)mac, local, dns, gateway, subnet);

// give the Ethernet shield a second to initialize:

::delay(1000);

this->conn.begin(addr, port);

IPAddress myip = Ethernet.localIP();

BLYNK\_LOG("My IP: %d.%d.%d.%d", myip[0], myip[1], myip[2], myip[3]);

}

};

#endif