This weeks demo is for Pin Control over ethernet and as a bonus a small OLED Demo

These sketchs use a “real” Arduino ethernet shield. And by that I mean one using a:
W5100 Ethernet Controller
http://arduino.cc/en/Main/ArduinoEthernetShield
http://www.ebay.com/itm/UNO-ATMega328-1280-MEGA2560-Ethernet-Shield-W5100-Arduino-Main-Board-M3Y2-/171438857683?pt=LH_DefaultDomain_0&hash=item27ea8d5dd3

Like everything else the price of these is dropping, but if you do buy a cheap one off eBay make sure it has the W5100 there are others out that dont and they dont work the same way.
I think I paid $10 or $11 for mine a year ago, now you can find them for $8 bucks or less.

Pin control over ethernet is based on this project:
http://bildr.org/2011/06/arduino-ethernet-pin-control/
alternative website:
https://web.archive.org/web/20140702134232/http://bildr.org/2011/06/arduino-ethernet-pin-control/

The above project sets up a webserver running on the Arduino. The address of the server is displayed on a serial terminal. (192.168.1.xxx) It is setup get a DHCP from your router.

Using the IP address you can set pins 2 to 9 to flash a LED.

http://192.168.1.45/?2 will flash the LED on PIN 2.
http://192.168.1.45/?23 will flash the LEDs on PIN2 and PIN 3 in order.

So some interesting patterns can be made from this:
http://192.168.1.45/?2345678998765432 will create something like K.I.T.T.s scanner.

```
//ARDUINO 1.0+ ONLY
//ARDUINO 1.0+ ONLY


#include <Ethernet.h>
#include <SPI.h>
boolean reading = false;

////////////////////////////////////////////////////////////////////
//CONFIGURE
////////////////////////////////////////////////////////////////////
  //byte ip[] = { 192, 168, 0, 199 };   //Manual setup only
  //byte gateway[] = { 192, 168, 0, 1 }; //Manual setup only
  //byte subnet[] = { 255, 255, 255, 0 }; //Manual setup only
```

```
  // if need to change the MAC address (Very Rare)
  byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

  EthernetServer server = EthernetServer(80); //port 80
//////////////////////////////////////////////////////////////////////////

void setup(){
  Serial.begin(9600);

  //Pins 10,11,12 & 13 are used by the ethernet shield

  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);

  Ethernet.begin(mac);
  //Ethernet.begin(mac, ip, gateway, subnet); //for manual setup

  server.begin();
  Serial.println(Ethernet.localIP());

}

void loop(){

  // listen for incoming clients, and process qequest.
  checkForClient();

}

void checkForClient(){

  EthernetClient client = server.available();

  if (client) {

        // an http request ends with a blank line
        boolean currentLineIsBlank = true;
        boolean sentHeader = false;
```

```
while (client.connected()) {
if (client.available()) {

if(!sentHeader){
// send a standard http response header
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
client.println();
sentHeader = true;
}

char c = client.read();

if(reading && c == ' ') reading = false;
if(c == '?') reading = true; //found the ?, begin reading the info

if(reading){
Serial.print(c);

switch (c) {
case '2':
//add code here to trigger on 2
triggerPin(2, client);
break;
case '3':
//add code here to trigger on 3
triggerPin(3, client);
break;
case '4':
//add code here to trigger on 4
triggerPin(4, client);
break;
case '5':
//add code here to trigger on 5
triggerPin(5, client);
break;
case '6':
//add code here to trigger on 6
triggerPin(6, client);
break;
case '7':
//add code here to trigger on 7
triggerPin(7, client);
```

```
            break;
            case '8':
            //add code here to trigger on 8
            triggerPin(8, client);
            break;
            case '9':
            //add code here to trigger on 9
            triggerPin(9, client);
            break;
            }

            }

            if (c == '\n' && currentLineIsBlank)  break;

            if (c == '\n') {
            currentLineIsBlank = true;
            }else if (c != '\r') {
            currentLineIsBlank = false;
            }

            }
            }

            delay(1); // give the web browser time to receive the data
            client.stop(); // close the connection:

  }

}

void triggerPin(int pin, EthernetClient client){
//blink a pin - Client needed just for HTML output purposes.
  client.print("Turning on pin ");
  client.println(pin);
  client.print("<br>");

  digitalWrite(pin, HIGH);
  delay(25);
  digitalWrite(pin, LOW);
  delay(25);
}
```

This is interesting, but really not useful. So I modified the sketch to toggle the pins on or off (from one state to another) - my code can be found here:
https://codebender.cc/sketch:60959

This was done using a array of zeros or ones - the array represents the LED states, the position inside the array is the PIN number,
in otherwords:
boolean ledPins[] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
^
This is PIN 0
the next zero is PIN 1, the next PIN 2, next PIN 3 etc.
So the question is why do we have 10 pins listed when there are only 8 LEDs -
the simple answer is I needed a place holder of two digits - (with this example, pins 0 and 1 are left open so as to easily reprogram the Arduino) So the LEDs are hooked to PINs 2 to 9.
Pin 10, 11, 12, 13 are used by the ethernet shield.
The main part of the sketch I modified is below:

```
void triggerPin(int pin, EthernetClient client){
//blink a pin - Client needed just for HTML output purposes.
  client.print("Toggling pin:  ");
  client.println(pin);
  client.print("<br>");

  digitalWrite(pin, !ledPins[pin]);
ledPins[pin]=!ledPins[pin];
}
```

The sketch gives output to both the serial monitor, and the website.

I liked how this worked, and thought this is a lot more useful if you need to remotely turn something on or off.

But I asked myself what if you want to read and display a vaule on the website:
So I modified the sketch one more time:
https://codebender.cc/sketch:60961

This time I wanted to read a Analog joystick: and I added this code:

```
case '0':
                    while(analogRead(0)>522) {
                    client.print("Joystick 0: ");
                    client.print(analogRead(0));
                    client.print("<br>");
                    }
```

```
                break;

                case '1':
                while(analogRead(1)>522) {
                client.print("Joystick 1: ");
                client.print(analogRead(1));
                client.print("<br>");
    }
                break;
```

To activate this part of the code you need to send the webserver a zero (0) or a one (1)

This part of the code will stay active as long as the joystick vaule is above 522 (522 is the center position) Which also mean that I am only reading two directions from Joystick - either down or left (but not up or right)

It's not the best way to make this work, and this was more of a test to make it work, then it was to make it work correctly! :-)

So I looked at the example library that is included with the Arduino IDE:
That sketch can be found here:
https://codebender.cc/sketch:61050
Which induces a couple of new things to the webserver - It also sets up a static IP which for a IOT device (or a control over IP device) might make more since to do.

Next thing it does is show how to have the page served auto refresh this can be a very useful thing to know.

It also will read all 6 analog ports and display the vaule on the web page.
For the demo I am only using analog Pin 0 and Pin 1 (So I modified the sketch to only read what I am using)

Again this was interesting, and displayed a good example but I wanted to use a website to turn pins on or off - or display vaules
and it lead me to this website:
http://randomnerdtutorials.com/arduino-webserver-with-an-arduino-ethernet-shield/
This is a really good example of control and was close to what I wanted so
Of course I modified the sketch for what I really wanted to
https://codebender.cc/sketch:61051

```
/*
 Created by Rui Santos
 Visit: http://randomnerdtutorials.com for more arduino projects
```

```
 Arduino with Ethernet Shield
 */

#include <SPI.h>
#include <Ethernet.h>

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };   //physical mac address
byte ip[] = { 192, 168, 1, 178 };                      // ip in lan (that's what you need to use in your
browser. ("192.168.1.178")
byte gateway[] = { 192, 168, 1, 1 };                   // internet access via router
byte subnet[] = { 255, 255, 255, 0 };                  //subnet mask
EthernetServer server(80);                             //server port
String readString;

void setup() {
 // Open serial communications and wait for port to open:
  Serial.begin(9600);
   while (!Serial) {
          ; // wait for serial port to connect. Needed for Leonardo only
  }
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);

  // start the Ethernet connection and the server:
  Ethernet.begin(mac, ip, gateway, subnet);
  server.begin();
  Serial.print("server is at ");
  Serial.println(Ethernet.localIP());
}


void loop() {
  // Create a client connection
  EthernetClient client = server.available();
  if (client) {
          while (client.connected()) {
          if (client.available()) {
          char c = client.read();
```

```
          //read char by char HTTP request
          if (readString.length() < 100) {
          //store characters to string
          readString += c;
          //Serial.print(c);
          }

          //if HTTP request has ended
          if (c == '\n') {
          Serial.println(readString); //print to serial monitor for debuging

          client.println("HTTP/1.1 200 OK"); //send new page
          client.println("Content-Type: text/html");
          client.println();
          client.println("<HTML>");
          client.println("<HEAD>");
          // client.println("<meta name='apple-mobile-web-app-capable' content='yes' />");
          //client.println("<meta name='apple-mobile-web-app-status-bar-style' content='black-
translucent' />");
          client.println("<link rel='stylesheet' type='text/css' href='http://randomnerdtutorials.com/
ethernetcss.css' />");
          client.println("<TITLE>Project Turn on Stuff From Website</TITLE>");
          client.println("</HEAD>");
          client.println("<BODY>");
          client.println("<H1>Turn on Stuff from a Website</H1>");
          client.println("<hr />");
          client.println("<br />");
          client.println("<H2>Arduino with Ethernet Shield</H2>");
          client.println("<br />");
          client.println("<a href=\"/?button1on\"\">Turn On LED 1</a>");
          client.println("<a href=\"/?button1off\"\">Turn Off LED 1</a><br />");
          client.println("<br />");
          client.println("<br />");
          client.println("<a href=\"/?button2on\"\">Turn On LED 2</a>");
          client.println("<a href=\"/?button2off\"\">Turn Off LED 2</a><br />");
          client.println("<br />");
          client.println("<br />");
          client.println("<a href=\"/?button3on\"\">Turn On LED 3</a>");
          client.println("<a href=\"/?button3off\"\">Turn Off LED 3</a><br />");
          client.println("<br />");
          client.println("<br />");
          client.println("<a href=\"/?button4on\"\">Turn On LED 4</a>");
          client.println("<a href=\"/?button4off\"\">Turn Off LED 4</a><br />");
```

```
            client.println("<br />");
            client.println("<br />");
            client.println("<a href=\"/?button5on\"\">Turn On LED 5 </a>");
            client.println("<a href=\"/?button5off\"\">Turn Off LED 5</a><br />");
            client.println("<br />");
            client.println("<br />");
            client.println("<a href=\"/?button6on\"\">Turn On LED 6</a>");
            client.println("<a href=\"/?button6off\"\">Turn Off LED 6</a><br />");
            client.println("<br />");
            client.println("<br />");
            /*   client.println("<a href=\"/?7on\"\">Turn On LED 7</a>");
            client.println("<a href=\"/?7off\"\">Turn Off LED 7</a><br />");
            client.println("<br />");
            client.println("<br />");
            client.println("<a href=\"/?8on\"\">Turn On LED 8</a>");
            client.println("<a href=\"/?8off\"\">Turn Off LED 8</a><br />");
            client.println("<br />");
            client.println("<br />");
            */
            client.println("<p>Based on a sketch ");
            client.println("<p>Created by Rui Santos. Visit http://randomnerdtutorials.com for more
projects!</p>");
            client.println("<br />");
            client.println("</BODY>");
            client.println("</HTML>");

            delay(1);
            //stopping client
            client.stop();
            //controls the Arduino if you press the buttons
            if (readString.indexOf("?button1on") >0){
            digitalWrite(2, HIGH);
            }
            if (readString.indexOf("?button1off") >0){
            digitalWrite(2, LOW);
            }

            if (readString.indexOf("?button2on") >0){
            digitalWrite(3, HIGH);
            }
            if (readString.indexOf("?button2off") >0){
            digitalWrite(3, LOW);
            }
```

```
if (readString.indexOf("?button3on") >0){
digitalWrite(4, HIGH);
}
if (readString.indexOf("?button3off") >0){
digitalWrite(4, LOW);
}


if (readString.indexOf("?button4on") >0){
digitalWrite(5, HIGH);
}
if (readString.indexOf("?button4off") >0){
digitalWrite(5, LOW);
}

if (readString.indexOf("?button5on") >0){
digitalWrite(6, HIGH);
}
if (readString.indexOf("?button5off") >0){
digitalWrite(6, LOW);
}

if (readString.indexOf("?button6on") >0){
digitalWrite(7, HIGH);
}
if (readString.indexOf("?button6off") >0){
digitalWrite(7, LOW);
}

/*
if (readString.indexOf("?7on") >0){
digitalWrite(8, HIGH);
}
if (readString.indexOf("?7off") >0){
digitalWrite(8, LOW);
}

if (readString.indexOf("?8on") >0){
digitalWrite(9, HIGH);
}
if (readString.indexOf("?8off") >0){
digitalWrite(9, LOW);
}
*/
```

```
                //clearing string for next read
        readString="";

        }
        }
        }
}
}
```

This time, it makes a “real” web page with buttons that will turn on or off LEDs
(a personal note: I was only able to get PINs 2 to 7 to work and I’m not sure why as the code is the same for all the other LEDs)
You might take notice that when a button is pushed the URL will look very much like the 1st example we looked at.
http://192.168.1.178/?button1on

Also I didn’t add anything to read the analog pins - but it would be easy to add something that could read a tempature sensor or some other sensor.
Using these examples we can do just about anything with control over ethernet.
Thou there seems to be a limit to just what the web server can do.

*** BONUS *** BONUS *** BONUS ***

I2C 0.96" OLED display module

OLED
noun
noun: **OLED**; plural noun: **OLEDs**

1. a light-emitting diode containing thin flexible sheets of an organic electroluminescent material, used for visual displays.
2. "because OLEDs emit light, they consume significantly less power"

http://www.ebay.com/itm/0-96-I2C-IIC-SPI-Serial-128X64-OLED-LCD-LED-Display-Module-for-Arduino-white-it-/251536387002?pt=LH_DefaultDomain_0&hash=item3a90bcabba

A few example sketches can be found here
http://www.wide.hk/products.php?product=I2C-0.96%22-OLED-display-module-%28-compatible-Arduino-%29

http://www.adafruit.com/product/326

And the adafruit tutorial:
https://learn.adafruit.com/monochrome-oled-breakouts

To make these examples work you need to have the Adafruit_SSD1306 and the Adafruit_GFX libraries that can be found on github.

There is also this library that works:
https://github.com/stanleyhuangyc/MultiLCD/tree/master/MicroLCD

I have examples using both libraries.

Like “FAKE CLOCK” using the MicroLCD library

```
/*********************************************************************
* Demo sketch for MicroLCD library
* Distributed under GPL v2.0
* Copyright (c) 2013-2014 Stanley Huang <stanleyhuangyc@gmail.com>
* All rights reserved.
* For more information, please visit http://arduinodev.com
*********************************************************************/

#include <Arduino.h>
#include <Wire.h>
#include <MicroLCD.h>

int s = 0;
int m = 0;
int h = 12;

//LCD_SH1106 lcd; /* for SH1106 OLED module */
LCD_SSD1306 lcd; /* for SSD1306 OLED module */

const PROGMEM uint8_t smile[48 * 48 / 8] = {
0x00,0x00,0x00,0x00,0x00,0x00,0x80,0xC0,0xE0,0xF0,0xF8,0xF8,0xFC,0xFC,0xFE,0xFE,0x7
E,0x7F,0x7F,0x3F,0x3F,0x3F,0x3F,0x3F,0x3F,0x3F,0x3F,0x3F,0x3F,0x7F,0x7F,0x7E,0xFE,0x
FE,0xFC,0xFC,0xF8,0xF8,0xF0,0xE0,0xC0,0x80,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0xC0,0xF0,0xFC,0xFE,0xFF,0xFF,0xFF,0x3F,0x1F,0x0F,0x07,0x03,0x01,0x00,0x80,0x80
,0x80,0x80,0x80,0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x80,0x80,0x80,0x80,0x80,0x80,0x00,0
x01,0x03,0x07,0x0F,0x1F,0x3F,0xFF,0xFF,0xFF,0xFE,0xFC,0xF0,0xC0,0x00,
```

```
0xFE,0xFF,0xFF,0xFF,0xFF,0xFF,0x07,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x06,0x1F,0x1F,
0x1F,0x3F,0x1F,0x1F,0x02,0x00,0x00,0x00,0x00,0x06,0x1F,0x1F,0x1F,0x3F,0x1F,0x1F,0x02,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x07,0xFF,0xFF,0xFF,0xFF,0xFF,0xFE,
0x7F,0xFF,0xFF,0xFF,0xFF,0xFF,0xE0,0x00,0x00,0x30,0xF8,0xF8,0xF8,0xF8,0xE0,0xC0,0x80
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x80,0xC0,0xE0,
0xF8,0xF8,0xFC,0xF8,0x30,0x00,0x00,0xE0,0xFF,0xFF,0xFF,0xFF,0xFF,0x7F,
0x00,0x03,0x0F,0x3F,0x7F,0xFF,0xFF,0xFF,0xFC,0xF8,0xF0,0xE1,0xC7,0x87,0x0F,0x1F,0x3
F,0x3F,0x3E,0x7E,0x7C,0x7C,0x7C,0x78,0x78,0x7C,0x7C,0x7C,0x7E,0x3E,0x3F,0x3F,0x1F,0
x0F,0x87,0xC7,0xE1,0xF0,0xF8,0xFC,0xFF,0xFF,0xFF,0x7F,0x3F,0x0F,0x03,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x03,0x07,0x0F,0x1F,0x1F,0x3F,0x3F,0x7F,0x7F,0x7E,
0xFE,0xFE,0xFC,0xFC,0xFC,0xFC,0xFC,0xFC,0xFC,0xFC,0xFC,0xFE,0xFE,0x7E,0x7F,
0x7F,0x3F,0x3F,0x1F,0x1F,0x0F,0x07,0x03,0x01,0x00,0x00,0x00,0x00,0x00,0x00,
};

void setup()
{
   lcd.begin();
}

void loop()
{
     lcd.clear();
   lcd.setCursor(40, 1);
   lcd.draw(smile, 48, 48);
          delay(1000);

          lcd.clear();
          lcd.setCursor(25,1);
          lcd.setFontSize(FONT_SIZE_MEDIUM);
  lcd.print("FAKE Clock");
  delay(1500);

for (h = 1; h<12; h++) {
  for (m = 0; m<59; m++) {
          for (s = 0; s<59; s++) {

   lcd.clear();
   lcd.setCursor(35, 3);
   lcd.setFontSize(FONT_SIZE_MEDIUM);
   lcd.print(h);
          lcd.print(":");
          if (m < 10) {lcd.print(0);
          lcd.print(m); } else {lcd.print(m);}
          lcd.print(":");
```

```
        if (s < 10) {lcd.print(0);
        lcd.print(s); } else {lcd.print(s);}
        lcd.print("P");
        lcd.setCursor(20, 6);
        lcd.print("Nov 9, 2014");
   delay(1000);
}
 }}}
```

Both libraries are pretty easy to use, and following the examples given pretty easy to modify

Some tricky things to get right is the number of pixels and placement of the “cursor” on the screen.
But one you have done a few you get the hang of where it will be.