Ethernet

The Arduino Ethernet Shield allows an Arduino board to connect to the internet. It is based on the Wiznet W5100 ethernet chip (datasheet). The Wiznet W5100 provides a network (IP) stack capable of both TCP and UDP. It supports up to four simultaneous socket connections. Use the Ethernet library to write sketches which connect to the internet using the shield. The ethernet shield connects to an Arduino board using long wire-wrap headers which extend through the shield. This keeps the pin layout intact and allows another shield to be stacked on top

Ethernet Library

With the Arduino Ethernet Shield, this library allows an Arduino board to connect to the internet. It can serve as either a server accepting incoming connections or a client making outgoing ones. The library supports up to four concurrent connection (incoming or outgoing or a combination).

Arduino communicates with the shield using the SPI bus. This is on digital pins 11, 12, and 13 on the Uno and pins 50, 51, and 52 on the Mega. On both boards, pin 10 is used as SS.On the Mega, the hardware SS pin, 53, is not used to select the W5100, but it must be kept as an output or the SPI interface won't work.

Creating Server using Ethernet

Point to be noted

1. If router allocates dynamic IP then use it in the browser to fetch the page from served by the Arduino (Dynamic IP address you can find in Serial Monitor)
2. Upload your program before inserting the Ethernet Shield to the Arduion Board

```
#include <SPI.h>

#include <Ethernet.h>


// Enter a MAC address and IP address for your controller below.
```

```
// The IP address will be dependent on your local network:
byte mac[] = { 0x00, 0xAA, 0xBB, 0xCC, 0xDA, 0x02 };
// Set the static IP address to use if the DHCP fails to assign
IPAddress ip(192, 168, 1, 10);

// Initialize the Ethernet server library
// with the IP address and port you want to use
// (port 80 is default for HTTP):
EthernetServer server(80);



void setup()
{
  //pinMode(2, INPUT);
  Serial.begin(9600);
  // start the Ethernet connection and the server:
  if (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to configure Ethernet using DHCP");
    // no point in carrying on, so do nothing forevermore:
    // try to congifure using IP address instead of DHCP:
    Ethernet.begin(mac, ip);
  }
  Serial.println("IP address given below:");
  delay (1000);
  Serial.println(Ethernet.localIP());
```

```
  server.begin();
}

void loop()
{

  // listen for incoming clients
  EthernetClient client = server.available();
  if (client) {
    // an http request ends with a blank line
    boolean currentLineIsBlank = true;
    while (client.connected()) {
      if (client.available()) {
        char c = client.read();
        // if you've gotten to the end of the line (received a newline
        // character) and the line is blank, the http request has ended,
        // so you can send a reply
        if (c == '\n' && currentLineIsBlank) {
          // send a standard http response header
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html");
          client.println();
          client.println("<cke:html><cke:body><center><h1>Hi!! I am your IOT</h1>");

          client.println("</center></cke:body></cke:html>");
```

```
        break;
      }
      if (c == '\n') {
        // you're starting a new line
        currentLineIsBlank = true;
      }
      else if (c != '\r') {
        // you've gotten a character on the current line
        currentLineIsBlank = false;
      }
    }
  }
  // give the web browser time to receive the data
  delay(1);
  // close the connection:
  client.stop();
}
}
```

Creating simple web client – requesting www.google.com from arduino

```
/*
  Web client

 This sketch connects to a website (http://www.google.com)
 using an Arduino Wiznet Ethernet shield.

 Circuit:
 * Ethernet shield attached to pins 10, 11, 12, 13

 created 18 Dec 2009
 by David A. Mellis
 modified 9 Apr 2012
 by Tom Igoe, based on work by Adrian McEwen

 */

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

// Enter a MAC address for your controller below.
// Newer Ethernet shields have a MAC address printed on a sticker on the shield
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
// if you don't want to use DNS (and reduce your sketch size)
// use the numeric IP instead of the name for the server:
```

```
//IPAddress server(74,125,232,128);  // numeric IP for Google (no DNS)
char server[] = "www.google.com";    // name address for Google (using DNS)

// Set the static IP address to use if the DHCP fails to assign
IPAddress ip(192, 168, 1, 17);

// Initialize the Ethernet client library
// with the IP address and port of the server
// that you want to connect to (port 80 is default for HTTP):
EthernetClient client;

void setup() {
  // Open serial communications and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }

  // start the Ethernet connection:
  if (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to configure Ethernet using DHCP");
    // try to congifure using IP address instead of DHCP:
    Ethernet.begin(mac, ip);
  }
  // give the Ethernet shield a second to initialize:
```

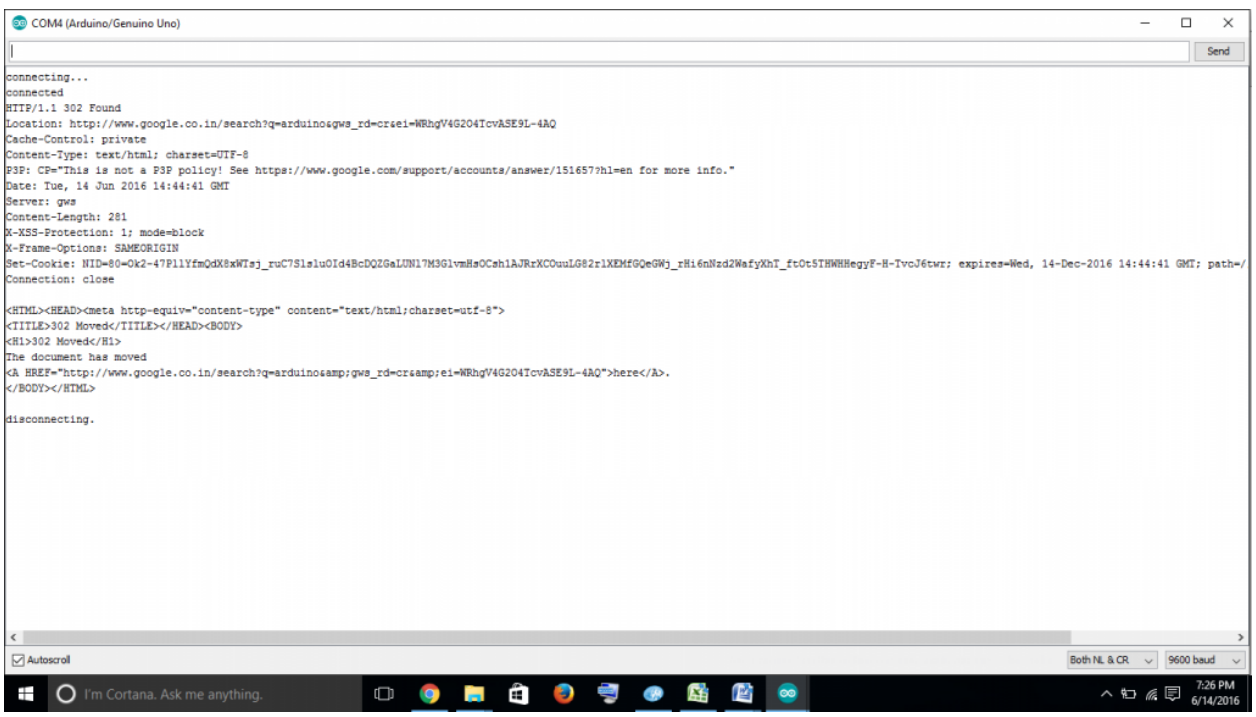
```
  delay(1000);
  Serial.println("connecting...");

  // if you get a connection, report back via serial:
  if (client.connect(server, 80)) {
    Serial.println("connected");
    // Make a HTTP request:
    client.println("GET /search?q=arduino HTTP/1.1");
    client.println("Host: www.google.com");
    client.println("Connection: close");
    client.println();
  } else {
    // if you didn't get a connection to the server:
    Serial.println("connection failed");
  }
}

void loop() {
  // if there are incoming bytes available
  // from the server, read them and print them:
  if (client.available()) {
    char c = client.read();
    Serial.print(c);
  }
```

```
  // if the server's disconnected, stop the client:
  if (!client.connected()) {
    Serial.println();
    Serial.println("disconnecting.");
    client.stop();

    // do nothing forevermore:
    while (true);
  }
}
```



Sending Arduino data to Web

http://data.sparkfun.com is used to create a stream to store data (Humidity and Temperature)

```
#include <Ethernet.h>

#include <SPI.h>



#include <dht.h>

byte mac[] = { 0x00, 0xAA, 0xBB, 0xCC, 0xDE, 0x01 }; // RESERVED MAC ADDRESS

EthernetClient client;

dht DHT;



#define DHT11_PIN 5



char s[200];

String data;



void setup()

{

Serial.begin(115200);



  if (Ethernet.begin(mac) == 0) {

    Serial.println("Failed to configure Ethernet using DHCP");

  }



Serial.println();

Serial.println("Type,\tstatus,\tHumidity (%),\tTemperature (C)");
```

```
}

void loop()
{
// READ DATA
Serial.print("DHT11, \t");
int chk = DHT.read11(DHT11_PIN);



// DISPLAY DATA
Serial.print(DHT.humidity, 1);
Serial.print(",\t");
Serial.println(DHT.temperature, 1);
int h=DHT.humidity;

int t1 = DHT.temperature;
float f2 = DHT.temperature-t1;
int t2 = trunc(f2 * 10000);

if (client.connect("data.sparkfun.com",80)) { // REPLACE WITH YOUR SERVER ADDRESS
  sprintf (s, "POST /input/EJO18W4KDYf2vodvrdpq?private_key=dqPGNEnVkoSwqRDqNDlr&humidity=%d&temperature=%d.%04d HTTP/1.0\r\n\r\n Host: data.sparkfun.com\r\n\r\n",h,t1,t2);
      client.println(s);
     Serial.println(s);
```

```
  client.println("Content-Type: application/x-www-form-urlencoded");
  client.print("Content-Length: ");
  client.println(data.length());
  client.println();
  client.print(data);
 }

 if (client.connected()) {
  client.stop();  // DISCONNECT FROM THE SERVER
 }

delay(2000);
}
```