EL SHIELD ETHERNET *(Versión3-4-18)*

Conectando Arduino a una red Ethernet

OJETIVOS

- Montar el Shield Ethernet.
- Conectarlo a nuestra LAN.
- Configurar los valores por DHCP.
- Configurar los valores manualmente

MATERIAL REQUERIDO.

	**Arduino UNO o equivalente**.
	**Un Shield Ethernet**.
	**Un cable Ethernet RJ45**
	**Acceso a la red local y al Switch**.

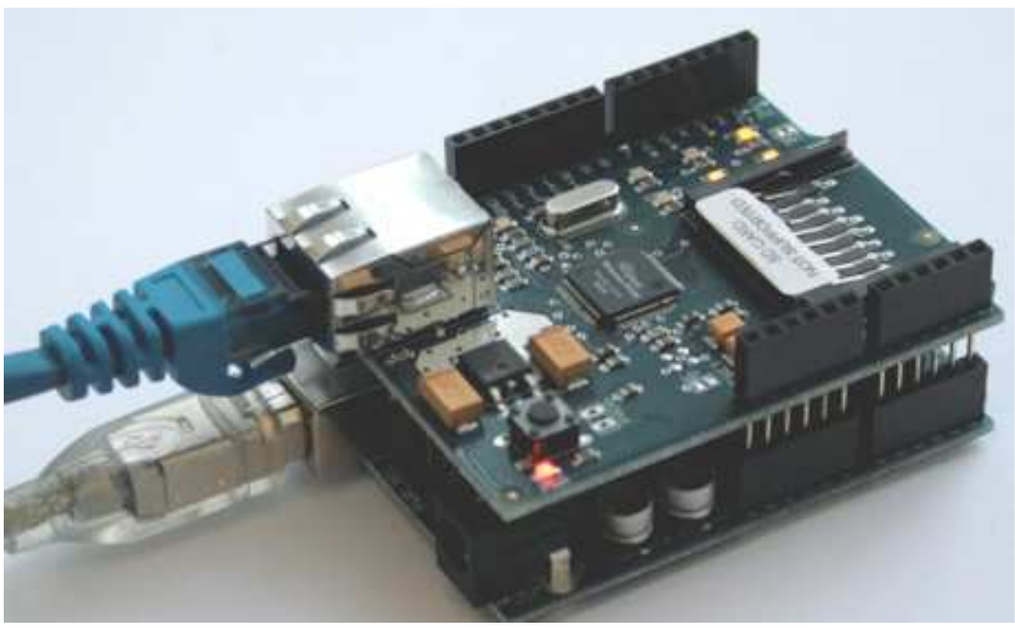
LOS SHIELD DE ARDUINO

La ventaja de trabajar con un Arduino, es que la comunidad ha ido creando cantidad de electrónica lista para usarse, sin necesidad de protoboard. En muchos casos son placas que vienen diseñadas para encajar encima de los pines hembra de que dispone Arduino, añadiéndole funcionalidad.
A este tipo de placas prediseñadas para encajar en los pines de nuestros Arduinos se les llama genéricamente **Shields** (Escudos), y en esta sesión vamos a utilizar un **shield Ethernet** que nos da acceso a la red local de que dispongamos, lo que nos abre unas posibilidades de lo más interesantes.

Para ello lo primero es encastrar el **shield** en nuestro Arduino:

- *Hacerlo con el Arduino apagado. Intentar colocar un Shield con Arduino bajo tensión es una ruleta rusa de consecuencias impredecibles.*
- *Los pines tienen que encajar con suavidad primero, y después presionar cuidadosamente hacia la placa Arduino. No forzar.*
- *Si alguno de los pines no se alinea adecuadamente, podemos ayudarle, con cuidado pero no forzar.*
- *Aseguraros de todos y cada uno de los pines encaja en los conectores de abajo.*

Una vez que haya encajado podemos alimentar el Arduino y luego conectar el cable de red, debería tener un aspecto similar al de la figura, veremos parpadear una serie de LEDs:



Arduino usa los pines digitales **10, 11, 12, y 13 (SPI)** para comunicarse con el W5100 en la ethernet shield. **Estos pines no pueden ser usados para e/s genéricas.**

El botón de reset en la shield resetea ambos, el W5100 y la placa Arduino.

La shield contiene varios LEDs para información:
ON: indica que la placa y la shield están alimentadas
LINK: indica la presencia de un enlace de red y parpadea cuando la shield envía o recibe datos
100M: indica la presencia de una conexión de red de 100 Mb/s (de forma opuesta a una de 10Mb/s)
RX: parpadea cuando el shield recibe datos
TX: parpadea cuando el shield envía datos
El jumper soldado marcado como **"INT"** puede ser conectado para permitir a la placa Arduino recibir notificaciones de eventos por interrupción desde el W5100, pero esto no está soportado por la librería Ethernet. El jumper conecta el pin INT del W5100 al pin digital 2 de Arduino.
El slot SD en la shield usa la librería **http://arduino.cc/en/Reference/SD** para manejarlo. El propio chip W5100 incluye el manejo de tarjetas SD.

Para usar la Ethernet Shield solo hay que montarla sobre la placa Arduino. Para cargar los sketches a la placa con el shield, conectarla al ordenador mediante el cable USB como se hace normalmente. Luego conectar la Ethernet a un ordenador, a un switch o a un router utilizando un cable ethernet standard (CAT5 o CAT6 con conectores RJ45). La conexión al ordenador puede requerir el uso de un cable cruzado (aunque muchos ordenadores actuales, pueden hacer el cruce de forma interna).

Para cualquier duda sobre el ethernet Shield consultar: http://arduino.cc/en/Main/ArduinoEthernetShield

Puntos a recordar del Ethernet Shield:

Opera a 5V suministrados desde la placa de Arduino
El controlador ethernet es el W5100 con 16K de buffer interno. No consume memoria.
El shield se comunica con el microcontrolador por el bus SPI, por lo tanto para usarlo siempre debemos incluir la libreria SPI.h.
Soporta hasta 4 conexiones simultáneas
Usar la librería Ethernet para manejar el shield. (Incluida en IDE Arduino)
El shield dispone de un lector de tarjetas micro-SD que puede ser usado para guardar ficheros y servirlos sobre la red. Para ello es necesaria la librería SD: (incluida en IDE Arduino)
Al trabajar con la SD, el pin 4 es usado como SS.
Arduino UNO se comunica con W5100 y la tarjeta SD usando el bus SPI a través del conector ICSP. **Por este motivo los pines 10, 11, 12 y 13 en el UNO y los 50, 51, 52 y 53 en el Mega no podrán usarse. En ambas placas los pines 10 y 4 se usan para seleccionar el W5100 y la tarjeta SD. El Ethernet y el SD no pueden trabajar simultáneamente y debemos tener cuidado al usar ambos de forma conjunta.**

Por si no sabemos como conectar el **Shield a la red Ethernet** (Y todos hemos tenido una primera vez), la idea es que usemos un cable Ethernet con RJ45 en ambos extremos, uno de ellos va al Shield y el otro al Router o Swicth de que dispongas.

Si te conectas con un **ADSL**, es probable que tengas instalado un Router Swicth cuya vista trasera se parezca a la imagen derecha.

Suelen incluir 4 entradas RJ45 (Y una RJ11, la típica telefónica)

- *Tendremos alguna de las bocas en uso con tu PC o similar.*
- *La WAN, es un RJ11, parecida pero un poco más estrecha, así que no te empeñes en forzarla si ves que no va.*
- *Normalmente cuando te instalan el Router, en la caja suele ir un cable Ethernet como el de la foto de la izquierda.*
- *Si tu acceso es por modem cable, necesitaras acceder a un Swicth.*

Conecta un extremo al Swicth, oirás un clic característico y el otro al Shield Ethernet y eso es todo, tu Arduino ya está conectado a la red local y probablemente a Internet.

PROBANDO LA CONEXIÓN CON DHCP

Tenemos que comprobar que la conexión es válida. No es posible conocer todos los modelos de Shields Ethernet, pero normalmente incluirán un LED bajo el conector que se encenderá con el cable conectado al Swicth, y se apagará si lo sueltas. (Y lo mismo ocurrirá en el Swicth, que encenderá un piloto en el frontal, cuando detecte la conexión Ethernet.

Vamos lo primero a buscar una dirección IP valida. Para ello y por simplificar vamos a comprobar si en vuestra red existe un servidor DHCP, que recordar servía para asignarnos una dirección IP al arrancar.

Vamos a probar con uno de los ejemplos que vienen con la librería Ethernet, que por cierto viene preinstalada en el IDE de Arduino.

Tenemos que importar en primer lugar un par de librerías, la SPI que ya conocemos y la Ethernet que es nueva. Podemos hacerlo directamente desde **el IDE de Arduino Programa\importarLibreria.**

```
#include <SPI.h>

#include <Ethernet.h
```

Ahora vamos a definir una dirección **MAC**. Lo normal es que no os de problemas la habitual, pero si vais a usar más de un Shield, necesitareis cambiar al menos una. Recordad que tiene que ser diferentes para cada nodo.

```
byte mac[ ] = {  0x00, 0xAA, 0xBB, 0xCC, 0xDE, 0x02 } ;
```

Y ahora vamos a crear una instancia de cliente Ethernet:

```
EthernetClient client;
```

En el setup, inicializamos la puerta serie:

```
void setup()

  {

     Serial.begin(9600);

     while (!Serial)

       {    ; // Solo para el Leonardo  }

  }
```

En las versiones actuales de la librería Ethernet, si no suministramos una dirección IP propia (Luego veremos cómo), al inicializar la Ethernet, intentara conseguir una dirección mediante DHCP automáticamente, y devolverá 1 si lo consigue y 0 en caso contrario:

```
if (Ethernet.begin(mac) == 0)    //Si devuelve error

  {

      Serial.println("No ha sido possible configurar la Ethernet por DHCP");

      while (true; )    ;          // No sigas, quédate aquí eternamente

  }
```

Si por el contrario devuelve un valor positivo, habrá rellenado un array Ethernet.localIP()[4],de 4 bites, con la dirección IP obtenida del DHCP

```
Serial.print("Mi direccion IP es: ");

for (byte B = 0; B < 4; B++)

  {

     Serial.print(Ethernet.localIP()[B], DEC);
```

```
      Serial.print(".");

    }

   Serial.println();
```

El programa imprimirá, la dirección IP obtenida.

Antes de cerrar el tema el DHCP, aquí os dejo un programa que lo usa para conectar a la red e imprime los valores recibidos.

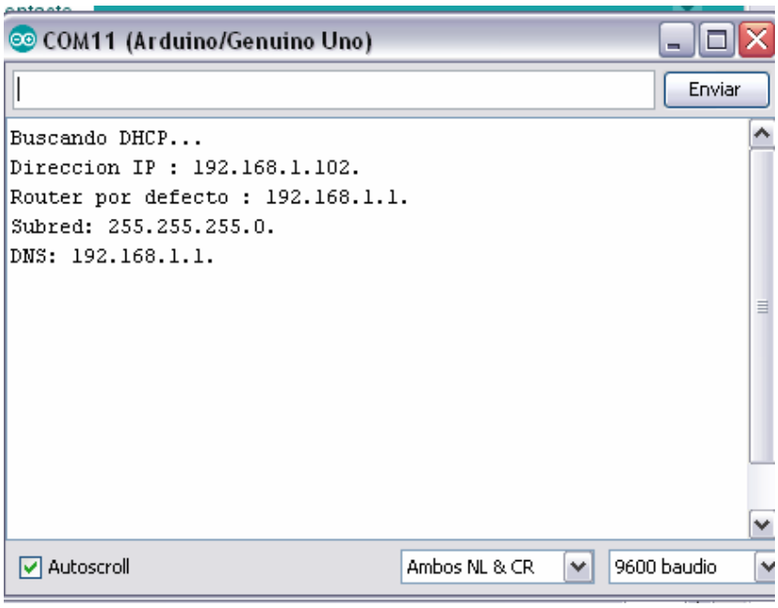
El programa siguiente resume lo anterior, imprime por el Monitor Serial los valores recibidos al conectarse a nuestra red. Se aconseja que sea el primer programa para conocer la Shield Ethernet.

```
/*  ---------------------------------------------------------------
    www.prometec.net
    Prog_61_1

    Obteniendo la direccion IP por DHCP.
    EL programa imprime la IP, subred, gateway y DNS asignados
---------------------------------------------------------------
*/

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

byte mac[] = {  0x00, 0xAA, 0xBB, 0xCC, 0xDE, 0x02};
EthernetClient client;

void setup()
  {  Serial.begin(9600);
     while (!Serial) { ; } // Soo para Leonardo
     Serial.println("Buscando DHCP...");

     if (Ethernet.begin(mac) == 0)
        {     Serial.println("Failed to configure Ethernet using DHCP");
              for (;;)   ; //No tiene sentido seguir
        }
  Serial.print("Direccion IP : ");
  for (byte B = 0; B < 4; B++) {
    Serial.print(Ethernet.localIP()[B], DEC);
    Serial.print(".");
  }
  Serial.println();

  Serial.print("Router por defecto : ");
  for (byte B = 0; B < 4; B++) {
    Serial.print(Ethernet.gatewayIP()[B], DEC);
    Serial.print(".");
  }
  Serial.println();

  Serial.print("Subred: ");
  for (byte B = 0; B < 4; B++) {
    Serial.print(Ethernet.subnetMask()[B], DEC);
    Serial.print(".");
  }
  Serial.println();
    Serial.print("DNS: ");
  for (byte B = 0; B < 4; B++) {
    Serial.print(Ethernet.dnsServerIP()[B], DEC);
```

```
    Serial.print(".");
  }
  Serial.println();
}

void loop() {

}
```

La salida sería la siguiente:

Probado en Router ProfesorDJB sin conexión a Internet



Si recurrimos al DHCP, tenemos que ser conscientes de que tiene un coste en uso de memoria. Si queremos economizarla especificamos solamente la IPAddress.ip y Arduino asignara por defecto al Gateway la dirección 1 dentro de la subred.

ACCEDIENDO A UNA PÁGINA WEB DESDE ARDUINO

El IDE incluye un ejemplo muy interesante de acceso a Google desde la Ethernet y para hacerle una consulta, pero NOSOTROS AQUÍ encaremos otro ejemplo para el caso de control de un LED vía una página WEB, que se visualizara de la siguiente manera:

Como observamos se debe tipear el en navegador la direccion IP de la placa Ethernet Arduino, en este caso 192.168.1.140.

Es recomendable fijar en el Router que cuando se detecte la placa mediante su MAC, este le asigne siempre la misma dirección IP.

Al presiona el boton ON se encendera el LED y al presionar OFF este se apagará.

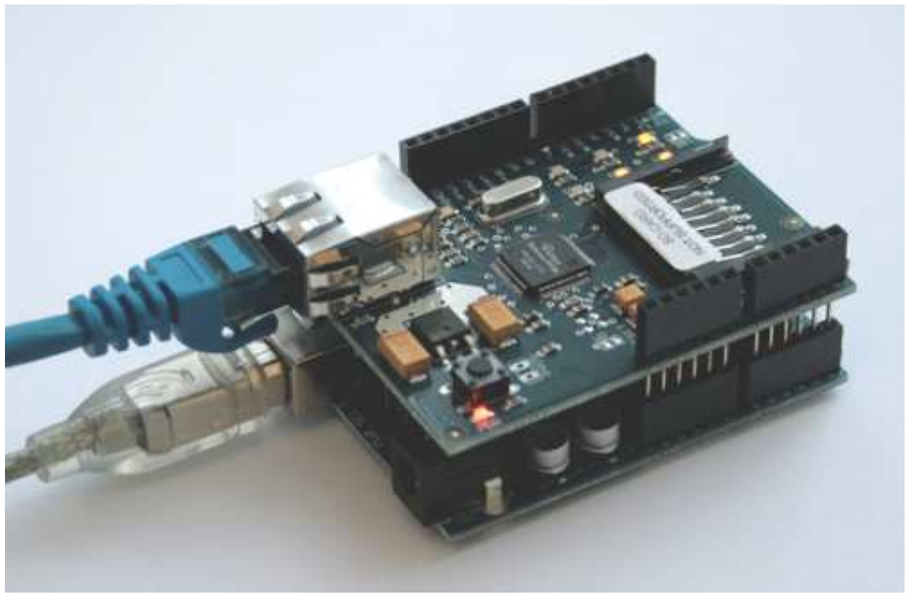
```
//Primer programa de control ensayado con exito  28-11-17
//Adaptacion Prof: Bolaños DJB

/*  TITULO: Control de un LED desde un servidor WEB con el Ethernet Shield W5100.
    AUTOR:
    MARIANO DEL CAMPO GARCÍA (@2016) --> INGENIERO TÉCNICO INDUSTRIAL ESPECIALIDAD ELECTRÓNICA


    DESCRIPCIÓN DEL PROGRAMA

    Con este programa vamos a controlar el encendido y apagado de un LED a través de internet, mediante
    el Servidor WEB creado con el Ethernet Shield que conectamos a Arduino, que tenemos comunicado con
    nuestro router WIFI.

   - Ethernet Shield colocada sobre Arduino UNO (Los pines coinciden al 100%).
   - Ethernet Shield conectada al router WIFI mediante cable Ethernet standard (conector RJ45)
   - Arduino UNO alimentado a través de una batería externa de Litio-ion de 12V/3000mAh (DC 12300)
*/

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

  // Declaración de la direcciones MAC,IP,GATEWAY y SUBNET.
  byte mac[]={0xDE,0xAD,0xBE,0xEF,0xFE,0xED};
  // Dentro del cdm de Windows ponemos ipconfig y buscamos en adaptador de LAN inalámbrica
  IPAddress ip(192,168,1,140); // 192.168.1.XX donde XX es una dirección que no esté utilizada
                               //En Router ProfesorDJB se reservo esta IP a esa MAC
  IPAddress gateway(192, 168, 1, 1);
  IPAddress subnet(255, 255, 255, 0);

  // Creamos un servidor Web con el puerto 80 que es el puerto HTTP por defecto
  EthernetServer server(80);

  int LED = 8; // Pin digital para el LED

  String estado = "OFF"; // Estado del LED inicialmente "OFF"

  void setup()
  {
```

```
  Serial.begin(9600); // Comienzo de la comunicación serie

  // Inicializamos la comunicación Ethernet y el servidor
  Ethernet.begin(mac, ip, gateway, subnet);
  server.begin();
  Serial.print("La IP del servidor local es:  ");
  Serial.println(Ethernet.localIP()); // Nos devuelve la IP del Ethernet Shield

  pinMode(LED,OUTPUT); // Pin digital 8 como salida
}

void loop()
{
  EthernetClient client = server.available(); // Creamos un cliente Web
  // Cuando detecte un cliente a través de una petición HTTP
  if (client)
  {
    Serial.println(); // Salto de línea
    Serial.println("Nuevo cliente");
    Serial.println();
    boolean currentLineIsBlank = true; // Una petición HTTP acaba con una línea en blanco
    String cadena=""; // Creamos una cadena de caracteres vacía
    while (client.connected())
    {
      if (client.available())
      {
        char c = client.read();// Leemos la petición HTTP carácter por carácter
        Serial.write(c);// Visualizamos la petición HTTP por el Monitor Serial
        cadena.concat(c);// Unimos el String 'cadena' con la petición HTTP (c).
        // De esta manera convertimos la petición HTTP a un String

         // Ya que hemos convertido la petición HTTP a una cadena de caracteres, ahora podremos buscar partes del texto.
         int posicion=cadena.indexOf("LED="); // Guardamos la posición de la instancia "LED=" a la variable 'posicion'

          if(cadena.substring(posicion)=="LED=ON") // Si en la posición 'posicion' hay "LED=ON"
          {
            digitalWrite(LED,HIGH);
            estado="ON";
          }
          if(cadena.substring(posicion)=="LED=OFF") // Si en la posición 'posicion' hay "LED=OFF"
```

```
          {
            digitalWrite(LED,LOW);
            estado="OFF";
          }

        // Cuando reciba una línea en blanco, quiere decir que la petición HTTP ha acabado y el servidor Web está listo
        // para enviar una respuesta
        if (c == '\n' && currentLineIsBlank)
        {
            // Enviamos al cliente una respuesta HTTP
            client.println("HTTP/1.1 200 OK");
            client.println("Content-Type: text/html");
            client.println();

            // Página web en formato HTML
            client.println("<html>");
            client.println("<head>");
            client.println("</head>");
            client.println("<body style='background-color: #40E0D0;'>");
            client.println("<br/><br/>");
            client.println("<h1 align='center'>LED controlado por Servidor Web con Arduino</h1>");
            // Creamos los botones.
            // Para enviar parámetros a través de HTML se utiliza el método URL encode.
            // Los parámetros se envían a través del símbolo '?'
            client.println("<div style='text-align:center;'>");
            client.println("<button onClick=location.href='./?LED=ON\' style='margin:auto;background-color: #84B1FF;color: snow;padding: 10px;border: 1px solid #3F7CFF;width:85px;'>");
            client.println("ON");
            client.println("</button>");
    client.println("<button onClick=location.href='./?LED=OFF\' style='margin:auto;background-color: #84B1FF;color: snow;padding: 10px;border: 1px solid #3F7CFF;width:85px;'>");
            client.println("OFF");
            client.println("</button>");
            client.println("<br/><br/>");
            client.println("<b>ESTADO DEL LED = ");
            client.print(estado);
            client.println("</b><br/>");
            client.println("</b></body>");
            client.println("</html>");
            break;
        }
        if (c == '\n')
```

```
        {
          currentLineIsBlank = true;
        }
        else if (c != '\r')
        {
          currentLineIsBlank = false;
        }
      }
    }
    // Dar tiempo al navegador para recibir los datos
    delay(10);
    client.stop();// Cierra la conexión
  }
}
```

El archivo del texto del programa lo puede obtener en este tutor.



Como ya mencionamos la página se vería:

Y si vemos mediante el navegador el código fuente de la misma:

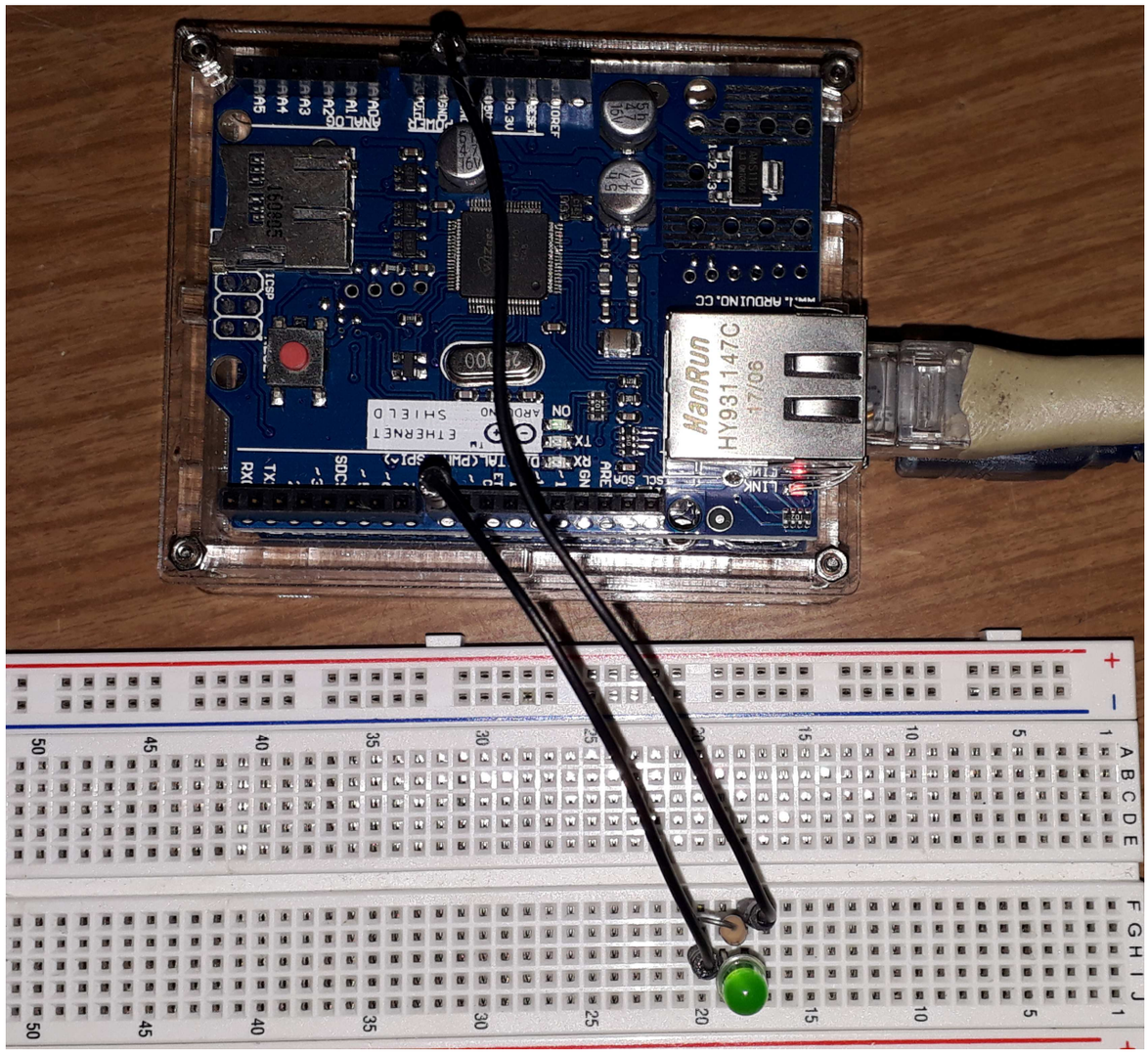
```
<html>
<head>
</head>
<body style='background-color: #40E0D0;'>
<br/><br/>
<h1 align='center'>LED controlado por Servidor Web con Arduino</h1>
<div style='text-align:center;'>
<button onClick=location.href='./?LED=ON' style='margin:auto;background-color: #84B1FF;color:
snow;padding: 10px;border: 1px solid #3F7CFF;width:85px;'>
ON
</button>
<button onClick=location.href='./?LED=OFF' style='margin:auto;background-color: #84B1FF;color:
snow;padding: 10px;border: 1px solid #3F7CFF;width:85px;'>
OFF
</button>
<br/><br/>
<b>ESTADO DEL LED =
OFF</b><br/>
</b></body>
</html>
```

Que podemos comparar con el sector del programa Arduino correspondiente (vea lo de Arduino y lo de HTML

```
----------------------------------------
-----------------------------------------------
// Página web en formato HTML
            client.println("<html>");
            client.println("<head>");
            client.println("</head>");
            client.println("<body style='background-color: #40E0D0;'>");
            client.println("<br/><br/>");
            client.println("<h1 align='center'>LED controlado por Servidor Web con Arduino</h1>");
            // Creamos los botones.
            // Para enviar parámetros a través de HTML se utiliza el método URL encode.
            // Los parámetros se envían a través del símbolo '?'
            client.println("<div style='text-align:center;'>");
            client.println("<button onClick=location.href='./?LED=ON\' style='margin:auto;background-
color: #84B1FF;color: snow;padding: 10px;border: 1px solid #3F7CFF;width:85px;'>");
```

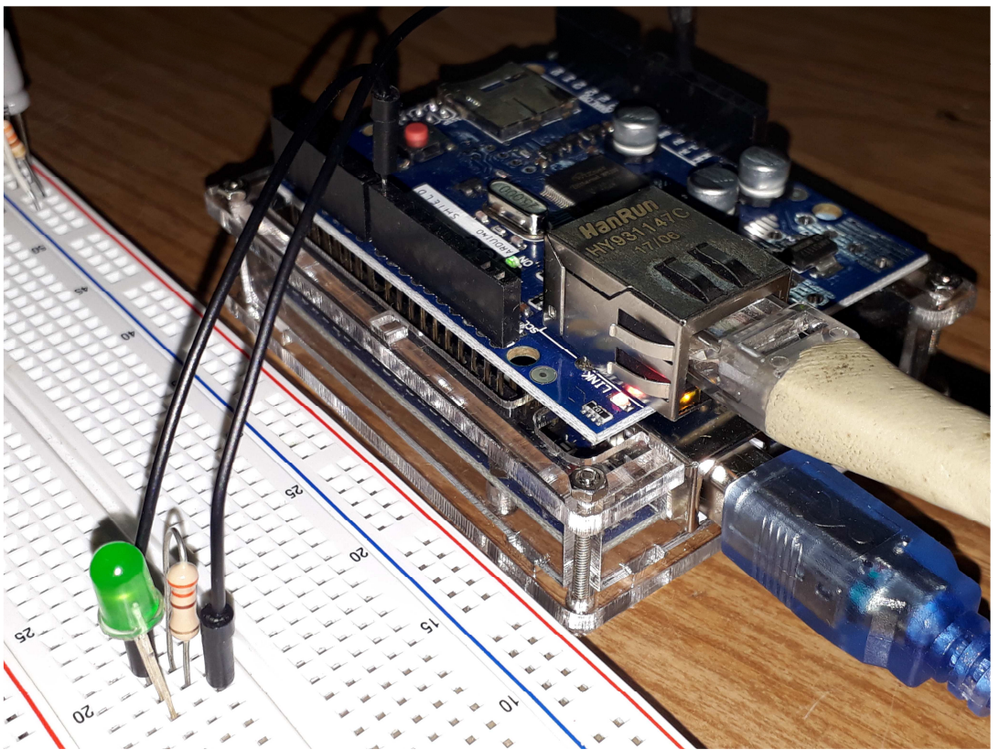
```
            client.println("ON");
            client.println("</button>");
            client.println("<button onClick=location.href='./?LED=OFF\' style='margin:auto;background-
color: #84B1FF;color: snow;padding: 10px;border: 1px solid #3F7CFF;width:85px;'>");
            client.println("OFF");
            client.println("</button>");
            client.println("<br/><br/>");
            client.println("<b>ESTADO DEL LED = ");
            client.print(estado);
            client.println("</b><br/>");
            client.println("</b></body>");
            client.println("</html>");

---------------------------------------------------------------
```

Fotografías del circuito real ensayado.

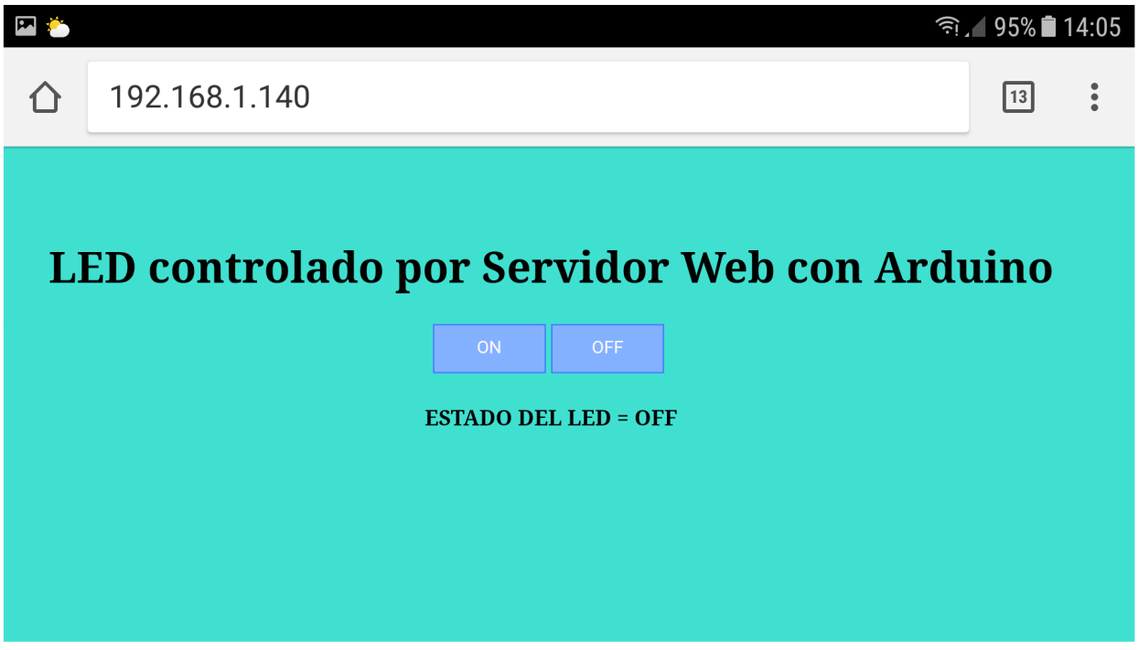


Como podemos observar la shield se encastra encima del Arduino 1, (recuerde que se llama shield escudo).

Podrá encontrar en el TutorArduino el caso de 2 LEDs.

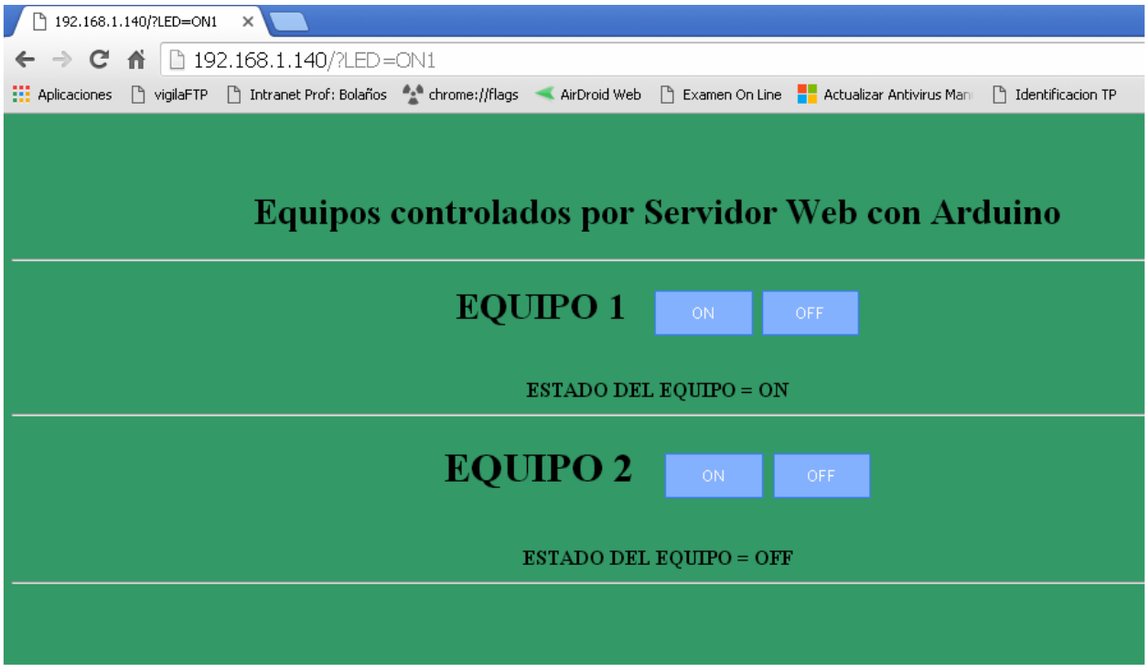


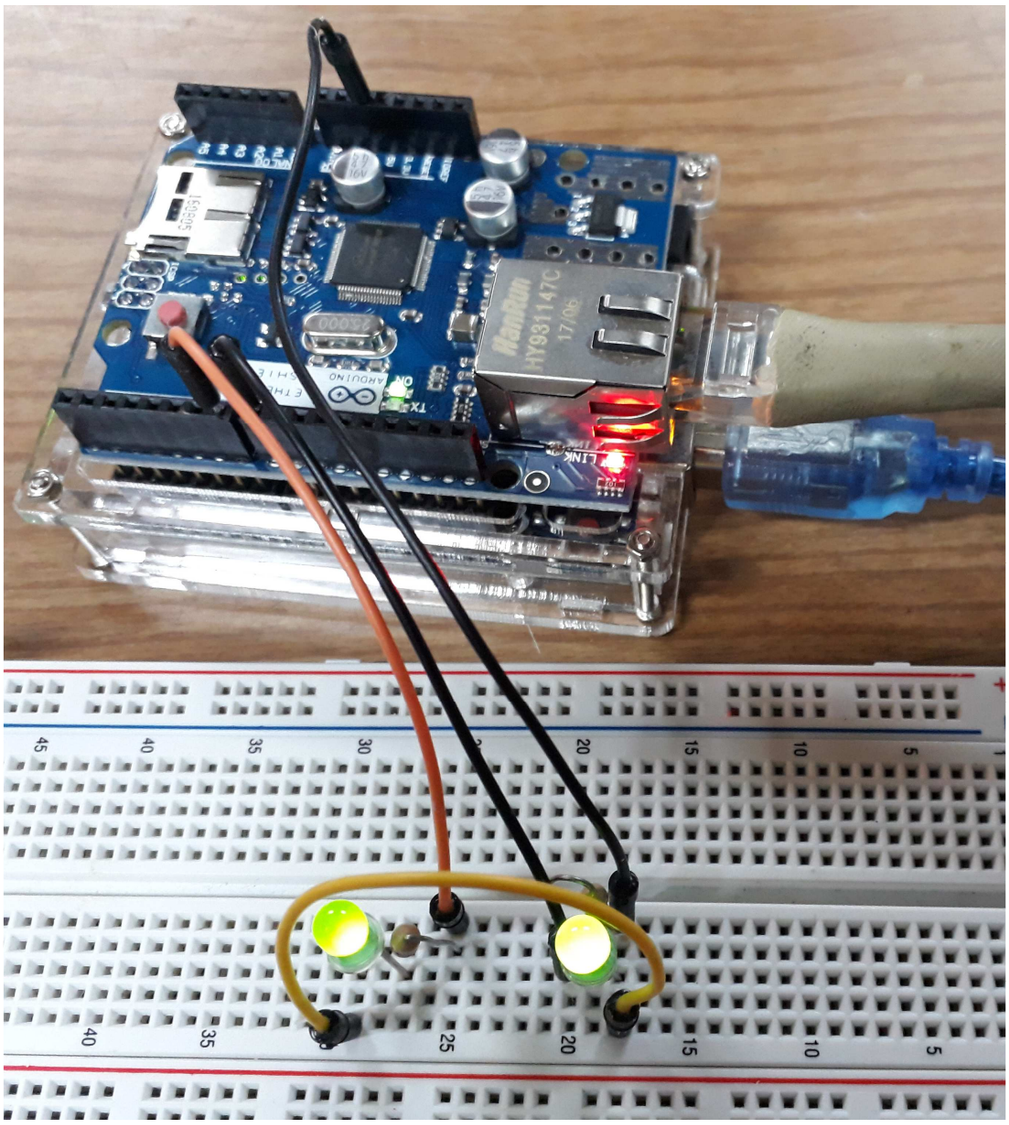
Es posible ver la página web desde un celular:



Caso de comandar 2 LEDs

Recuerde que el archivo en forma de texto lo encuentra dentro del tutor.





Posteriormente se realizo otro programa que accione mas botones, pero la limitación de memoria de Arduino Uno, solo se llego a 2 botones. Se recomendaría pasar a un Arduino superior o buscar alguna alternativa.

RESUMEN DE LA SESIÓN

.

- Vimos como conectar el Shield Ethernet a nuestros Arduinos.
- Mostramos algunas ideas para conectar el Shield a nuestro Swicth mediante un cable Ethernet RJ45.
- En las versiones actuales de la librería Ethernet, están soportados tanto el DHCP con el DNS, lo que nos ahorra muchos dolores de cabeza.
- Pero si por cualquier motivo, no podemos usar DHCP, o el uso de memoria es limitante, podemos configurar valores IP manualmente. Vea el apunte: **Si_no_tenemosDHCP.pdf**
- Vimos también un pequeño ejemplo de cliente Web para activar un LED.