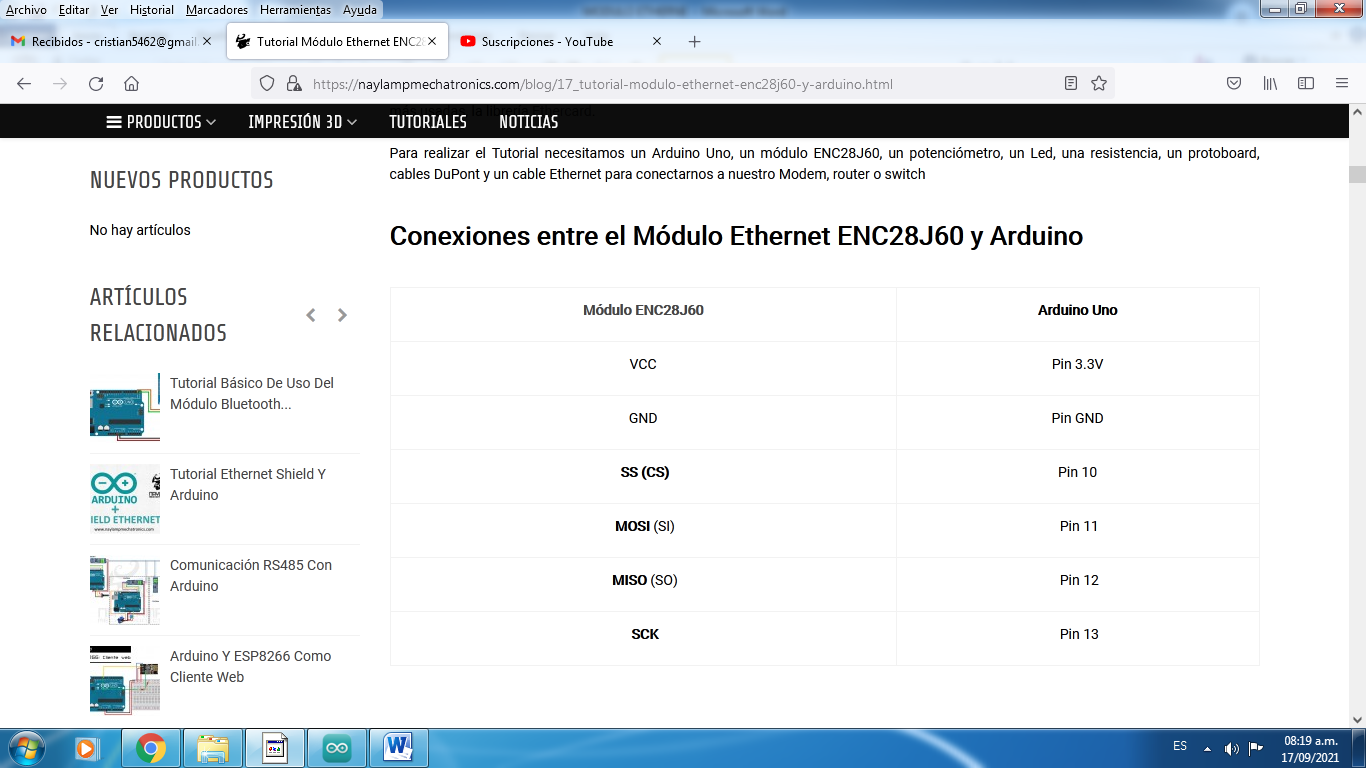
Con módulo **Arduino Shield Ethernet**, esta biblioteca permite que una placa **Arduino** se conecte a internet. Puede utilizar como un servidor capaz de aceptar conexiones entrantes o como un cliente que realiza las salientes. La biblioteca admite hasta cuatro conexiones simultáneas (entrante o saliente o combinadas).

La placa **Arduino** se comunica con el escudo mediante el bus SPI. Utiliza los pines digitales 11, 12, y 13 en la placa **Arduino** **Uno** y los pines 50, 51, y 52 en la placa **Arduino** **Mega**. En ambas placas, el pin 10 se utiliza como SS.



**Programa para el Módulo Ethernet ENC28J60**

Si es la Primera vez que estamos programando este módulo es necesario descargar e importar a nuestro IDE de Arduino la librería [Ethercard](https://github.com/jcw/ethercard).  
  
Una vez importado la librería escribimos el siguiente sketch:

#include <EtherCard.h> //LIBRERIA

static byte mymac[] = {0xDD,0xDD,0xDD,0x00,0x01,0x05};

static byte myip[] = {192,168,1,177};

byte Ethernet::buffer[700];

const int ledPin = 2;

char\* EstadoLed="OFF";

void **setup** () {

**Serial**.begin(9600);

**Serial**.println("Test del Modulo ENC28J60");

  if (!ether.begin(sizeof Ethernet::buffer, mymac, 10))

**Serial**.println( "No se ha podido acceder a la controlador Ethernet");

 else

**Serial**.println("Controlador Ethernet inicializado");

  if (!ether.staticSetup(myip))

**Serial**.println("No se pudo establecer la dirección IP");

**Serial**.println();

  pinMode(ledPin, OUTPUT);

  digitalWrite(ledPin, LOW);

}

static word homePage() {

 BufferFiller bfill = ether.tcpOffset();

 bfill.emit\_p(PSTR("HTTP/1.0 200 OK\r\n"

      "Content-Type: text/htmlrnPragma: no-cachernRefresh: 5\r\n\r\n"

      "<html><head><title>Naylamp Mechatronics</title></head>"

      "<body>"

      "<div style='text-align:center;'>"

      "<h1>Test del Módulo ENC28J60</h1>"

      "Tiempo transcurrido : $L segundos"

      "<br /><br />Estado del LED: $S<br />"

      "<a href=\"/?status=ON\"><input type=\"button\" value=\"ON\"></a>"

      "<a href=\"/?status=OFF\"><input type=\"button\" value=\"OFF\"></a>"

      "<br /><br />Potenciómetro: $D (resolución de 1024)"

      "<br /><br />"

      "<a href='http://www.naylampmechatronics.com/'>www.naylampmechatronics.com</a>"

      "</body></html>"

      ),millis()/1000,EstadoLed,analogRead(0));

  return bfill.position();

}

void **loop**() {

  word len = ether.packetReceive();

  word pos = ether.packetLoop(len);

  if(pos) {

    if(strstr((char \*)Ethernet::buffer + pos, "GET /?status=ON") != 0) {

**Serial**.println("Comando ON recivido");

      digitalWrite(ledPin, HIGH);

      EstadoLed = "ON";

    }

    if(strstr((char \*)Ethernet::buffer + pos, "GET /?status=OFF") != 0) {

**Serial**.println("Comando OFF recivido");

      digitalWrite(ledPin, LOW);

       EstadoLed= "OFF";

    }

    ether.httpServerReply(homePage()); // se envia página Web

  }

}

## ****Ahora expliquemos lo principal del código:****

static byte mymac[] = {0xDD,0xDD,0xDD,0x00,0x01,0x05};

static byte myip[] = {192,168,1,177};

Aquí configuramos nuestra  Mac y la IP , podemos poner cualquier valor, evitando duplicar con algún otro equipo conectado a nuestra red.  
  
En void setup () inicializamos el módulo Ethernet y reportamos por el puerto serial, aquí también configuramos el pin del led como salida.  
  
En la función **static word homePage()** escribimos nuestra página web, la cual debe estar en formato HTML:

static word homePage() {

 BufferFiller bfill = ether.tcpOffset();

 bfill.emit\_p(PSTR("HTTP/1.0 200 OK\r\n"

      "Content-Type: text/htmlrnPragma: no-cachernRefresh: 5\r\n\r\n"

      "<html><head><title>Naylamp Mechatronics</title></head>"

      "<body>"

      "<div style='text-align:center;'>"

      "<h1>Test del Módulo ENC28J60</h1>"

      "Tiempo transcurrido : $L segundos"

      "<br /><br />Estado del LED: $S<br />"

      "<a href=\"/?status=ON\"><input type=\"button\" value=\"ON\"></a>"

      "<a href=\"/?status=OFF\"><input type=\"button\" value=\"OFF\"></a>"

      "<br /><br />Potenciómetro: $D (resolución de 1024)"

      "<br /><br />"

      "<a href='http://www.naylampmechatronics.com/'>www.naylampmechatronics.com</a>"

      "</body></html>"

      ),millis()/1000,EstadoLed,analogRead(0));

  return bfill.position();

}

Notar que para enviar datos de variables hay que usar los marcadores $L  $S  $D para después enviar los datos de tipo Long, cadena, y decimal respectivamente.  
  
También estamos configurando para que el navegador vuelva a cargar la página cada 5 segundos esto se hace con “Refresh: 5”, es necesario cunado estamos mostrando datos que cambian constantemente. El tiempo de actualización dependerá del tipo de dato que se está mostrando. En algunas aplicaciones esto no es necesario o se opta por poner un botón para actualizar. Si no se desea simplemente quitar “Refresh: 5” en la función de la página web.  
  
La siguiente comparación se hace para ver si se han enviado datos desde el navegador mediante el método GET   
      if(strstr((char \*)Ethernet::buffer + pos, "GET /?status=ON") != 0) {

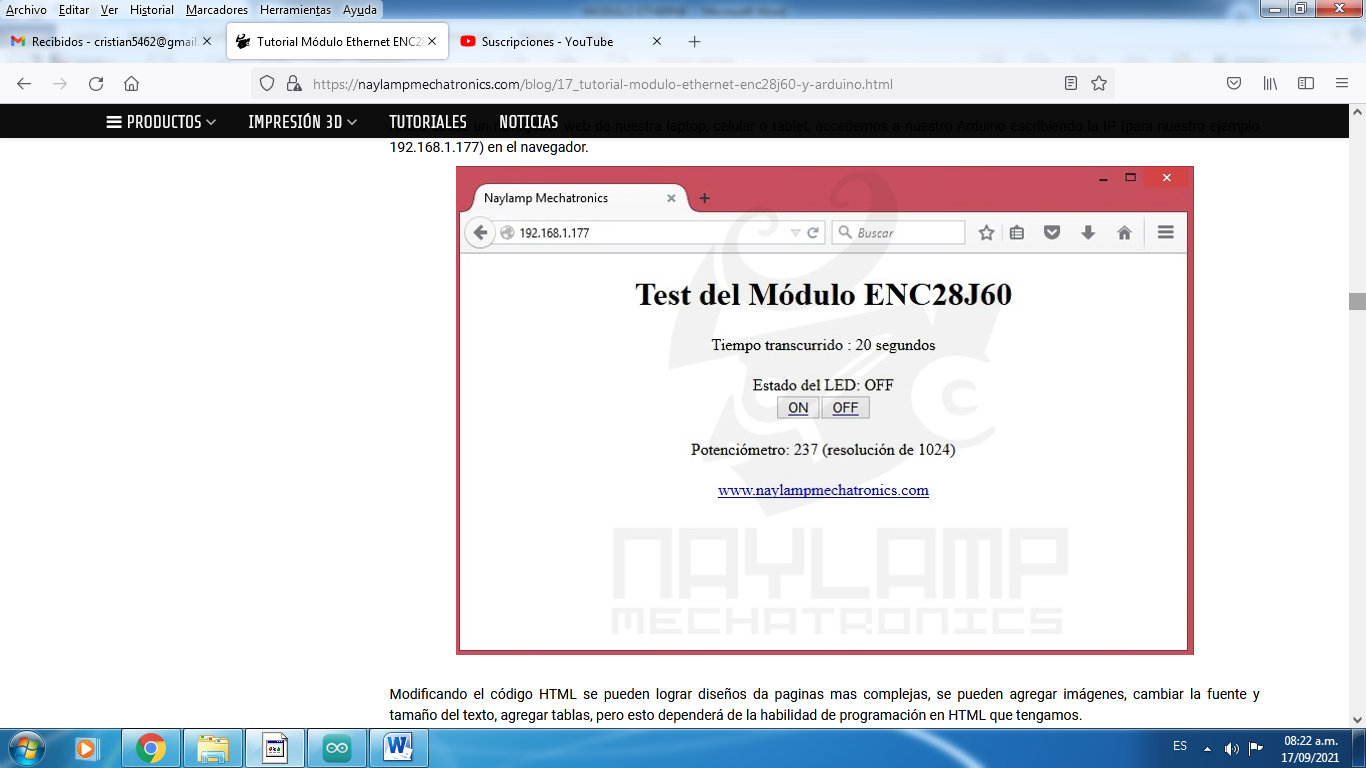
**Serial**.println("Comando ON recivido");

      digitalWrite(ledPin, HIGH);

      EstadoLed = "ON";

    }

En este caso estamos comparando si se ha recibido un dato con el método GET de la siguiente forma ?status=ON , esto sucede si en el navegador se presiona el botón ON (ver el código HTML  del botón en la función anterior)   
Una vez escrito el sketch lo cargamos a nuestro Arduino.  
Solo faltaría conectar el módulo ENC28J60 con un cable Ethernet a aun modem, router, Switch o a la PC.  
Ahora desde un navegador web de nuestra laptop, celular o tablet, accedemos a nuestro Arduino escribiendo la IP (para nuestro ejemplo 192.168.1.177) en el navegador.



Modificando el código HTML se pueden lograr diseños da paginas mas complejas, se pueden agregar imágenes, cambiar la fuente y tamaño del texto, agregar tablas, pero esto dependerá de la habilidad de programación en HTML que tengamos.