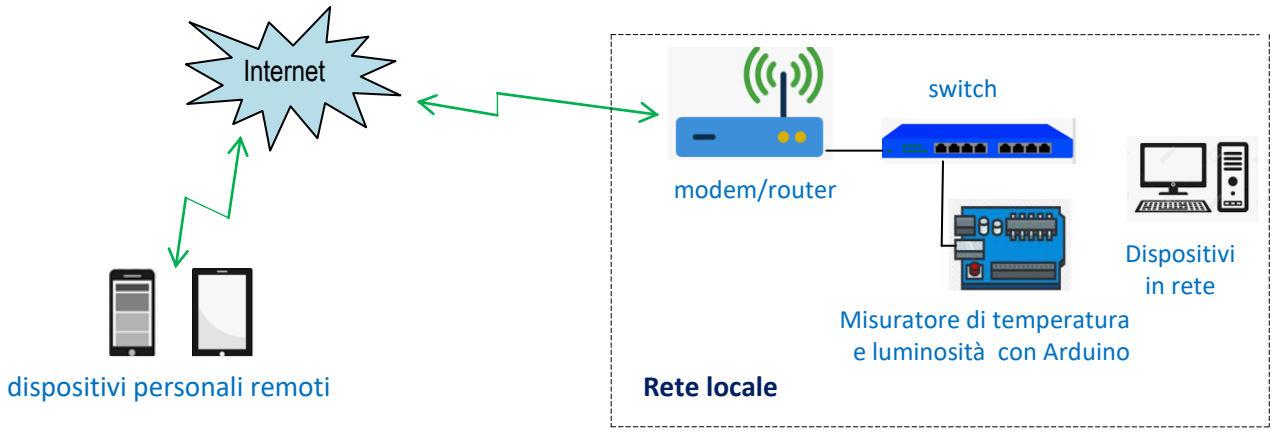
Arduino: monitoraggio remoto di temperatura e luminosità

Il progetto permette di leggere da remoto, via Internet, i valori di temperatura e luminosità misurati da una scheda Arduino; per semplicità non si prevede la possibilità di comandare Arduino da dispositivi remoti, cosa che sarà esplorata in un prossimo articolo.

E' un progetto introduttivo, semplice e didattico ma utile per affrontare una problematica, quale quella del controllo remoto, molto diffusa e con numerosi risvolti applicativi.

In questo caso, Arduino è connesso ad un router di una rete locale, e quindi a Internet, tramite una Ethernet Schield e la libreria "Ethernet Library".

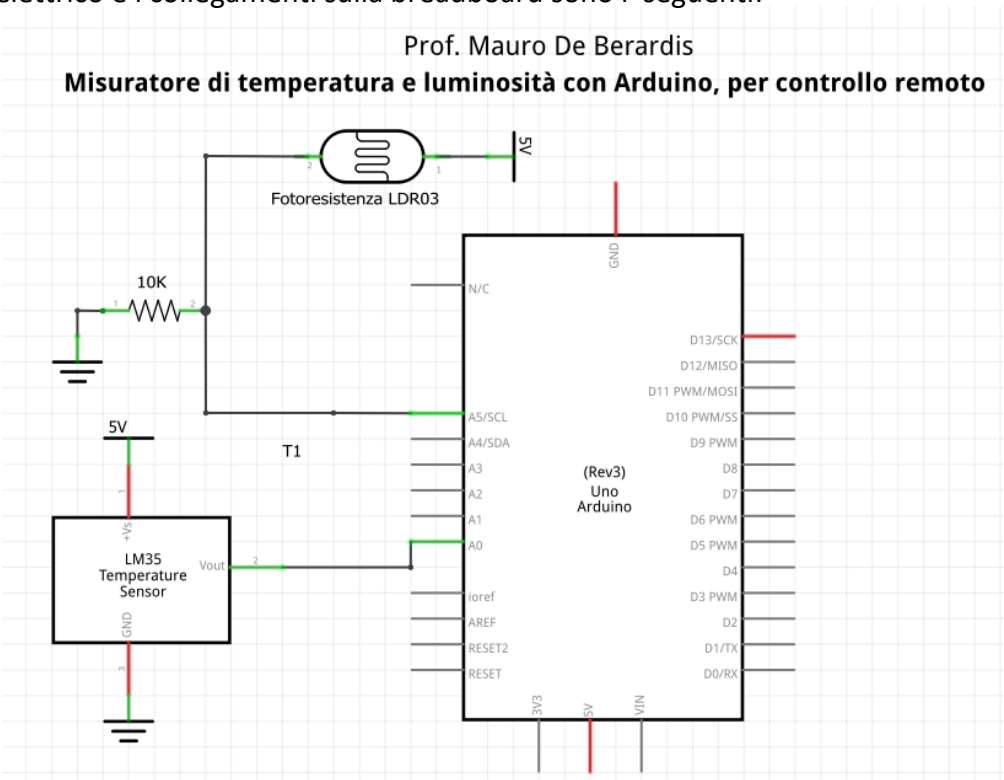
Lo schema generale dell'applicazione è il seguente:

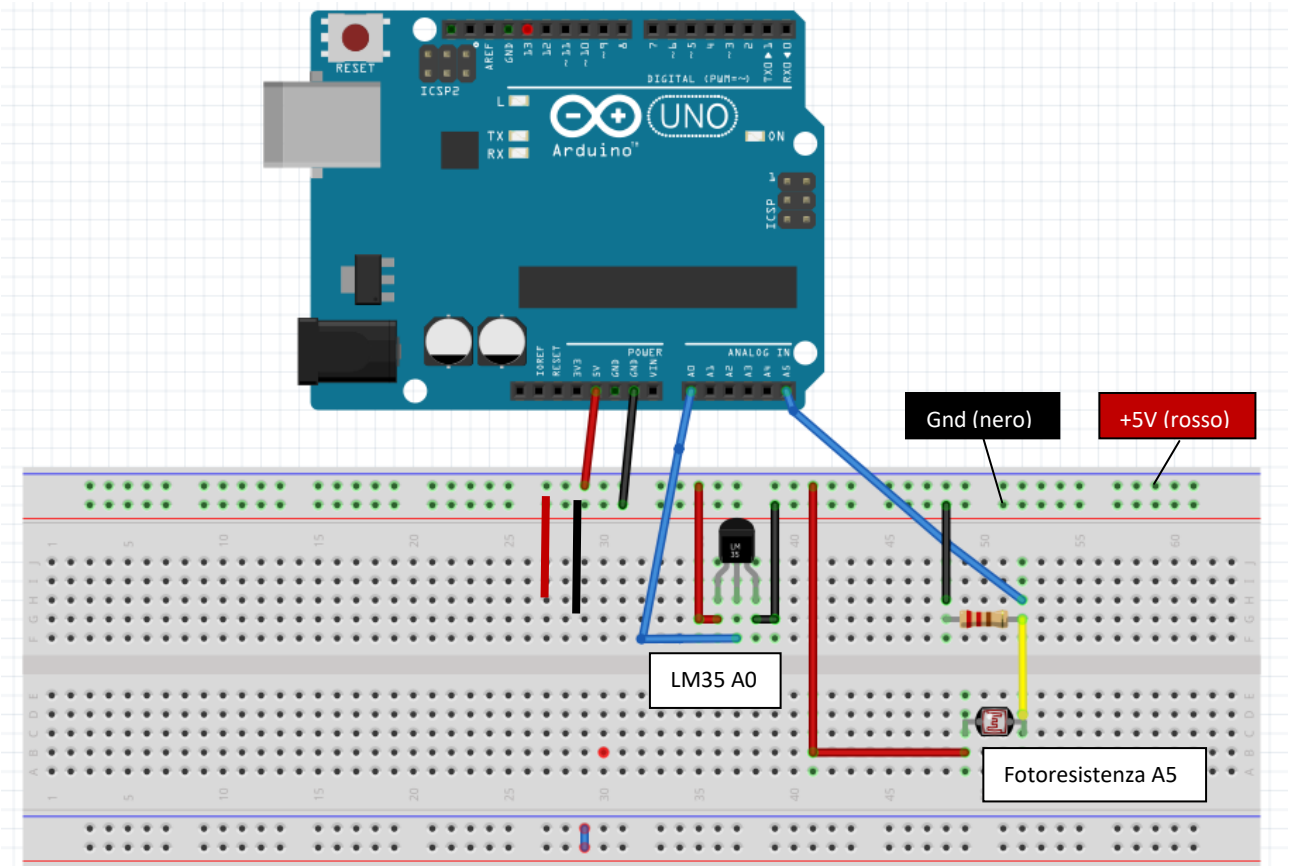


Per le competenze di Sistemi e Reti necessarie per capirne il funzionamento e per le abilità richieste per configurare alcuni dispositivi, il progetto è particolarmente adatto agli studenti del quarto/quinto anno delle specializzazioni Informatica e Telecomunicazioni degli Istituti Tecnici Tecnologici.

Il funzionamento della scheda è il seguente: Arduino esegue le lettura degli ingressi analogici A0 e A5 collegati rispettivamente ad un sensore di temperatura e ad un sensore di luminosità.

Lo schema elettrico e i collegamenti sulla breadboard sono i seguenti:

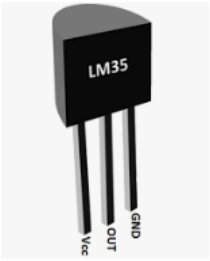




Componenti necessari

- Arduino collegato ad un PC Windows X
- Ethernet Shield con "Ethernet Library"
- Breadboard con cavetteria
- 1 circuito integrato LM35
- 1 Fotoresistenza
- 1 Resistenza da 10 KOhm

Per misurare la temperatura utilizziamo il sensore integrato LM35, un circuito poco costoso e sufficientemente preciso. La tensione di uscita è proporzionale alla temperatura e non è necessario realizzare circuiti di controllo aggiuntivi.



Guardando l'integrato di fronte, il piedino a sinistra deve essere collegato all'alimentazione 5V, quello centrale, che fornisce la tensione di uscita proporzionale alla temperatura, va collegato all'ingresso analogico A0, il piedino di destra va collegato a massa.

Quando Arduino legge con AnalogRead l'ingresso analogico A0, lo converte in digitale con un ADC (Convertitore analogico digitale) con risoluzione di 10 bit. I livelli di quantizzazione che l'ADC può fornire, sono 2^{10} =1024, ovvero sono tutte le combinazioni di 10 bit comprese tra 0000000000 e 1111111111 (in decimale da 0 a 1023).
Poiché il valore massimo del segnale analogico in ingresso è di 5 Volt e tale valore è associato al livello 1023, con una semplice proporzione il valore di tensione letto viene riportato tra 0 e 5 volt.
Infine, tenendo conto che il sensore integrato LM35 presenta un'uscita lineare in tensione uguale a 10 mV/°C, si ottiene facilmente il valore della temperatura.

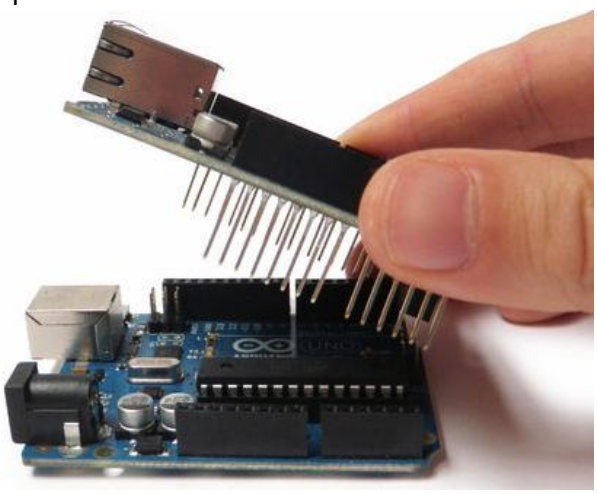
Anticipando una parte dello sketch dell'applicazione, le istruzioni per ottenere la temperatura sono le seguenti:

```
//Viene letto il valore della tensione di uscita  del sensore LM35 convertito in digitale
int ValoreLetto=analogRead(A0);
//Si ottiene la tensione   tra 0 e 5 volt utilizzando  la  proporzione 5V:1023=Vout:ValoreLetto
float Vout=5.0*ValoreLetto/1023.0;
//Si calcola la temperatura
float Temperatura =Vout*1000.0 / 10.0;
```

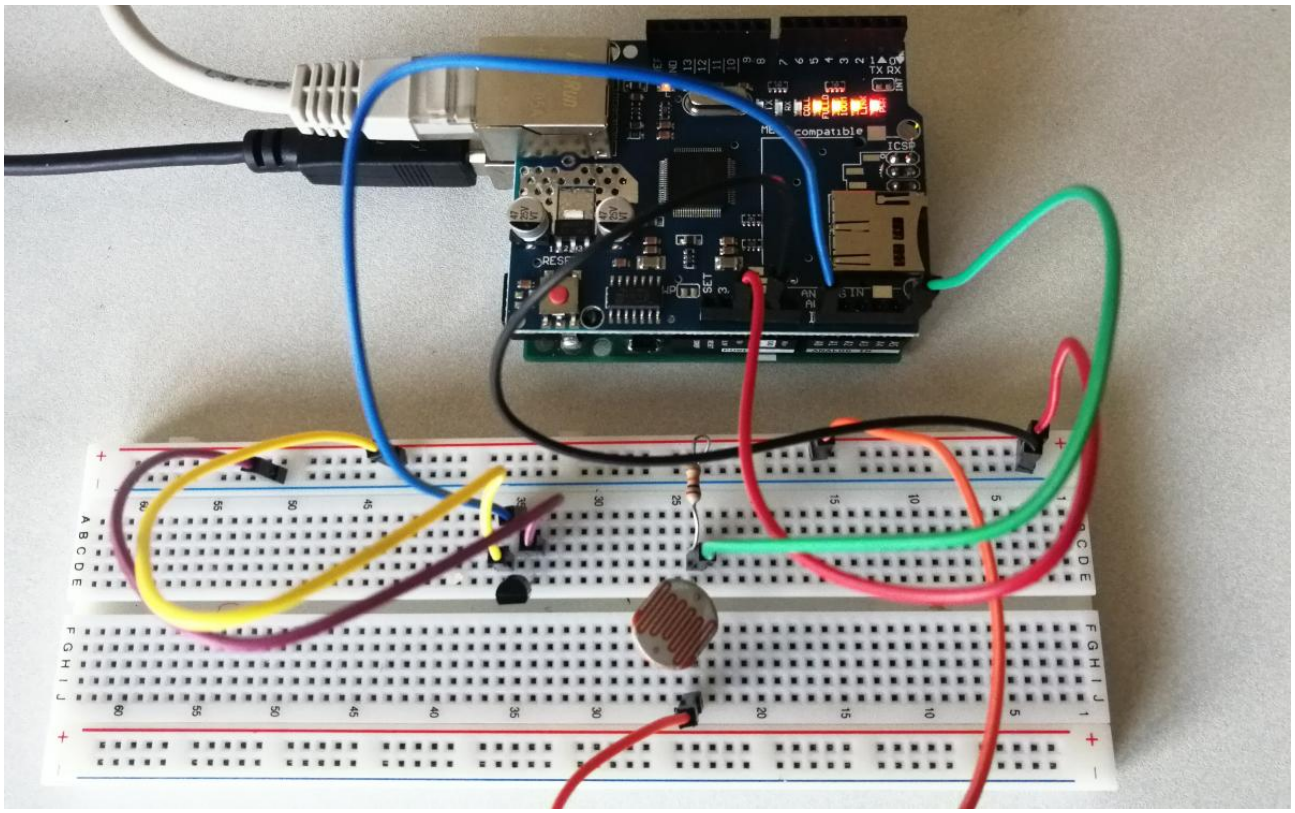
Per misurare la luminosità si utilizza una fotoresistenza, un dispositivo elettronico la cui resistenza varia in maniera inversamente proporzionale alla quantità di luce che lo colpisce.
Al buio il valore della resistenza è di qualche MOhm, in piena luce il valore scende a qualche KOhm.
Per la misura utilizziamo un partitore di tensione, con la fotoresistenza e una resistenza da 10 kOhm, collegato all'ingresso analogico A5.
In questo caso non ci preoccupiamo di convertire i valori forniti dal'ADC di Arduino in lumen o lux (lumen/mq) in quanto, al contrario che nel caso della temperatura, generalmente non abbiamo familiarità con queste unità di misura. Pertano assumiamo direttamente i valori letti con AnalogRead sul piedino A5 e convertiti dall'ADC a 10 bit, tenendo conto che tali valori tendono a 1023 nel caso di luce piena e prolungata e a 0 nel caso di buio totale e prolungato.

Circuito di prova

Dopo aver inserito Ethernet Shield in Arduino



realizziamo, utilizzando i piedini A0, A5, 5V e Gnd di Arduino "replicati" su Ethernet Shield, i collegamenti previsti per realizzare il misuratore di temperatura e luminosità. Colleghiamo quindi Arduino al PC (cavo USB nero) ed Ethernet Shield allo switch di rete (cavo di rete bianco)



L'Ethernet Schield, tramite il cavo di rete, connette Arduino alla rete locale, e quindi al router della nostra abitazione, consentendo ad un qualsiasi dispositivo (pc, tablet o smartphone) connesso ad Internet di leggere in una pagina web la temperatura e la luminosità rilevate da Arduino. Infatti, come vedremo tra poco, lo sketch utilizzato implementa in Arduino un web server con cui è possibile scambiare informazioni via protocollo http sulla porta 80.

Un po' di teoria

Prima di andare avanti è opportuno ricordare i principi di funzionamento sui quali si basa la suite di protocolli TCP/IP (Transport Control Protocol/Internet Protocol) che è alla base del funzionamento della stragrande maggioranza delle reti pubbliche e private. In un network basato sul TCP/IP, ciascun sistema è identificato in modo univoco da un indirizzo IP, costituito da quattro byte del tipo, in decimale, 192.168.1.100 e da un numero di porta di comunicazione. Tramite IP e numero di porta comunica con altri sistemi scambiando messaggi sotto forma di pacchetti. L'indirizzo IP, analogamente ad un numero di telefono, garantisce la possibilità di instaurare la comunicazione mentre la porta non è altro che un numero che serve a differenziare il servizio di rete, cioè l'applicazione usata per la comunicazione stessa (ad esempio, il servizio http, che ci permette di navigare sul web, ha tipicamente un numero di porta uguale ad 80, il servizio ftp, che permette di scaricare i files, ha un numero di porta uguale a 21 ecc.).

Gli indirizzi IP possono essere privati oppure pubblici.

Gli IP privati sono utilizzabili soltanto all'interno di reti locali e sono replicabili quante volte si vuole su altre reti locali. In genere tali indirizzi fanno parte delle seguenti serie di indirizzi riservati e che non possono essere instradati su Internet:

- 10.0.0.0 ÷ 10.255.255.255
- 172.16.0.0 ÷ 172.31.255.255
- 192.168.0.0 ÷ 192.168.255.255

Gli indirizzi pubblici sono invece non replicabili (non ne possono esitere due uguali) e consentono di accedere ad Internet.

Ogni dispositivo di una rete locale è contraddistinto da un indirizzo IP privato (ad esempio 192.168.1.10) ed è collegato, via cavo o via wifi, ad un router con IP appartenente alla stessa rete locale (generalmente l'IP di default del router è 192.168.1.1).

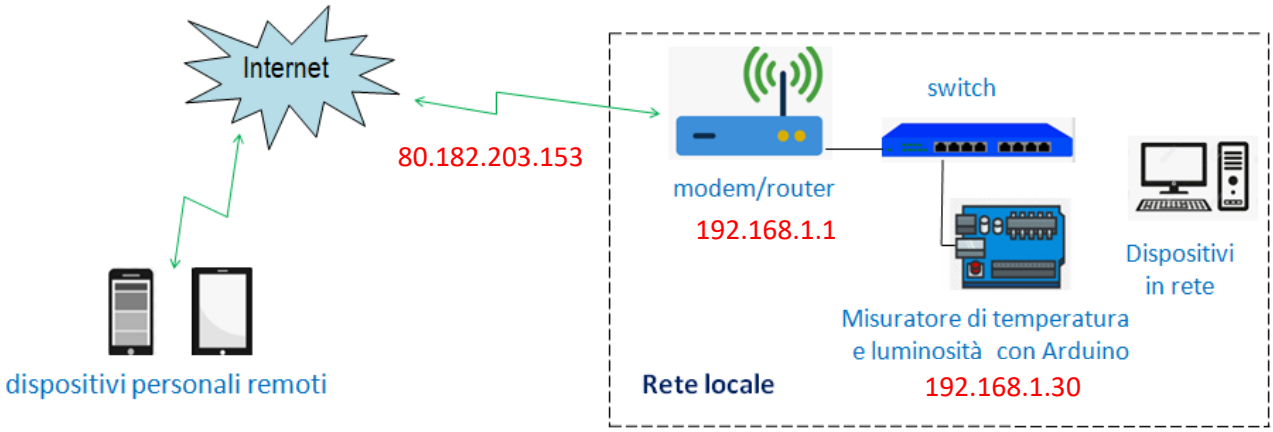
Per connettersi ad Internet Il router ha bisogno di un indirizzo pubblico, indirizzo che viene fornito dal Provider dei servizi che fornisce la connettività Internet. Tale indirizzo può essere determinato in vari modi, il più semplice è quello di collegarci ad uno dei tanti siti, ad esempio www.mioip.it, che forniscono il numero Ip pubblico a cui il router dell rete locale da cui si accede risulta connesso.
Accedendo ora, mioip.it mi dice:

Il tuo indirizzo IP è:

80.182.203.153

Torniamo sul progetto

Per quanto detto, la configurazione degli IP pubblici e privati di cui tener conto, è la seguente:



Il modem/router (Ip privato 192.168.1.1) si collega ad Internet con l'IP pubblico 80.182.203.153 e, grazie agli indirizzi IP privati della rete locale, instrada i vari pacchetti di dati verso Arduino e gli altri dispositivi in rete (PC desktop, notebook, stampanti di rete, ecc.)

Il circuito misuratore di temperatura e luminosità realizzato con Arduino ed Ethernet Schield si collega alla rete locale, e quindi al router, con l'IP privato 192.168.1.30.

Pertanto, per leggere da remoto i valori di temperatura e luminosità misurati da Arduino, dobbiamo prima raggiungere l'indirizzo pubblico del router e poi essere instradati verso l'indirizzo privato di Arduino.

Si presentano due problemi:

a) **Dobbiamo conoscere l'indirizzo IP pubblico del router**
Purtroppo tale indirizzo non sempre è fisso; anzi, per le utenze domestiche generalmente è dinamico e può variare con una certa frequenza. Il sito mioip.it oggi ci dice che l'IP pubblico della nostra rete è 80.182.203.153, domani tale indirizzo potrebbe essere un altro.
La soluzione migliore è ovviamente quella di attivare un IP fisso sul nostro router, ma spesso questa strada è costosa e impraticabile.
In alternativa possiamo utilizzare il servizio di DNS dinamico che diversi siti forniscono gratuitamente. Io utilizzo il servizio gratuito di **www.noip.com** che mi ha fornito un alias del tipo **xxxx.ddns.net**: a tale alias viene associato l'IP pubblico del router. L'aggiornamento dinamico di tale indirizzo viene effettuato da un app fornita da www.noip.com, presente sul mio server di rete e che si avvia come servizio Windows, Pertanto, se sul mio cellulare nella barra degli indirizzi del browser, digito xxxx.ddns.net, vengo reindirizzato verso l'IP pubblico che in quel momento risulta attivo nel mio router.

b) **Dobbiamo configurare il router in modo che,** una volta raggiunto da remoto attraverso l'IP pubblico**, stabilisca una comunicazione con la Ethernet Schield attraverso la porta 80.**
Per fare questo occorre impostare un "port forwarding", ovvero una nuova regola di routing per far si che ogni volta che da remoto giunge una richiesta http sulla porta 80, questa venga reindirizzata all'IP privato (nel nostro caso 192.168.1.30) a cui è associato il misuratore di temperatura e luminosità. Come fare?

Facendo riferimento al modem/router della mia rete (le modalità operative variano da modello a modello ma generalmente le opzioni che ho utilizzato sono presenti in tutti i modelli commerciali):

- nella sezione "Rete locale" aggiungo un IP statico associato al MAC address che utilizzerò nello sketch di Arduino. Ethernet Shield è una scheda basata sul chip intelligente W5100 che non ha un MAC Address preassegnato e può essere configurato con un indirizzo di 6 cifre esadecimali qualsiasi, purchè diverso dai MAC Address di altri dispositivi della LAN.Ad esempio scegliamo l'indirizzo 80:80:80:80:80:80.

- nella sezione "servizi Wan", imposto la seguente regola di routing:

	Nome	Protocollo	Porta WAN	Porta LAN	Indirizzo IP remoto	Indirizzo MAC remoto
ON	HTTP server	TCP/UDP	80	80	192.168.1.30	

che instrada le richieste HTTP verso l'IP privato associato alla scheda Arduino.

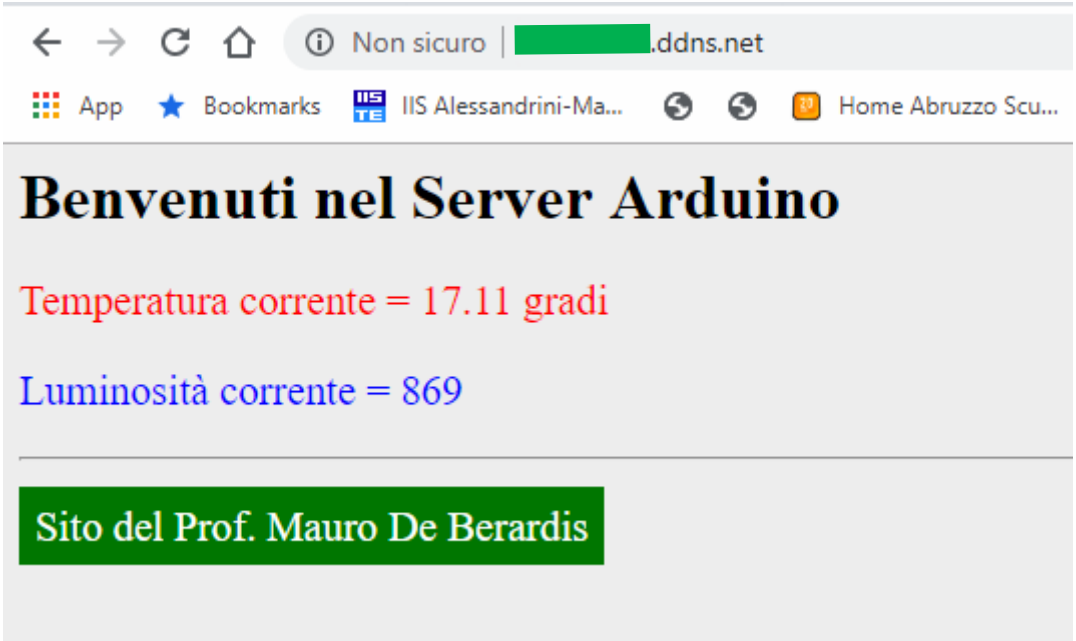
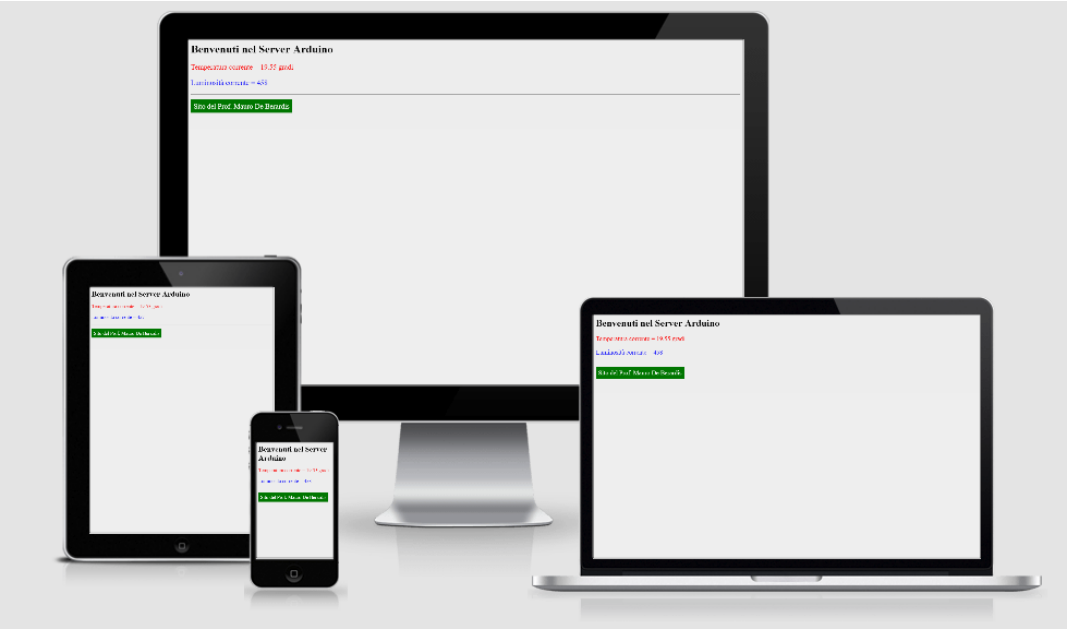
Corre l'obbligo a questo punto precisare che se la rete non è ben protetta sia internamente che esternamente, il port forwarding espone a problemi di sicurezza informatica. Tralasciando altre considerazioni e tenendo conto che il progetto è puramente didattico, consiglio di disabilitare la porta 80, aperta sull'IP 192.168.1.30, una volta testato il progetto.

	Nome	Protocollo	Porta WAN	Porta LAN	Indirizzo IP remoto	Indirizzo MAC remoto
OFF	HTTP server	TCP/UDP	80	80	192.168.1.30	

In alternativa, se si vuole testare il progetto in rete locale, evitando di intervenire sul modem/router, è sufficiente digitare sulla barra degli indirizzi del browser l'IP assegnato ad Arduino sullo sketch (nel nostro caso 192.168.1.30)

Sketch di Arduino

```
//  Misuratore di temperatura e luminosità  con Arduino ed Ethernet Shield
//  Mauro De Berardis
//
// SPI.h, Ethernet.h sono due  librerie necessarie per realizzare il progetto
// Con la  libreria Ethernet.h  è possibile far diventare Arduino un mini server web

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

/*
 * Ethernet Shield è una scheda  basata sul chip intelligente   W5100 che non ha un MAC Address
 * preassegnato e può essere configurato con un indirizzo di 6 cifre esadecimali qualsiasi, purchè
 * diverso dai MAC Address di altri dispositivi della LAN.
 * Scegliamo l'indirizzo 80:80:80:80:80:80  utilizzato per configurare un IP statico sul router.
 */
byte macaddress[] = {0x80, 0x80, 0x80, 0x80, 0x80, 0x80 };
/*
 * 192.168.1.30 è l'IP privato della scheda Arduino
 */
byte ipprivato[]={192,168,1,30};
/*
 * 192.168.1.1 è l'IP privato del router(il gateway che permette ai dispositivi della rete
 ] di accedere ad Internet)
 */
byte gateway[]={192,168,1,1};
byte subnetmask[] = { 255, 255, 255, 0 }; //subnet mask
/*
 *  La porta di default del protocollo HTTP è la porta 80
 */
EthernetServer Server(80);
String readStringa;
void setup() {
  Serial.begin(9600);
  Ethernet.begin(macaddress, ipprivato,gateway,subnetmask);
  Server.begin();
}

void loop()
{

  // Lettura luminosità---------------------------------------------
  // Arduino legge il valore della tensione del partitore Fotoresistenza-10K
  // convertito in digitale dall'ADC a 10bit
  int Luminosita=analogRead(A5);
 // Lettura temperatura---------------------------------------------
 //Arduino legge il valore della tensione di uscita  del sensore LM35 convertito in digitale
  int ValoreLetto=analogRead(A0);
 //Si ottiene la tensione  normalizzata  tra 0 e 5 volt con la  proporzione 5V:1023=Vout:ValoreLetto
  float Vout=5.0*ValoreLetto/1023.0 ;
  float Temperatura =Vout*1000.0 / 10.0; //Si calcola la temperatura
  // creo una connessione al client
  EthernetClient client = Server.available();
  if (client) {
  while (client.connected()) {
    if (client.available()) {
      char c = client.read();
       readStringa.concat(c);// memorizzo ogni carattere della stringa
        if (c == '\n' )//Se la richiesta HTTP è stata completata
        {
          // per il debug può essere utile stampare la stringa sul monitor seriale
          // Serial.print(readStringa);
        client.println("HTTP/1.1 200 OK....."); // costruzione pagina html
        client.println("Content-Type: text/html");
        client.println();
        // Testata pagina
        client.println("<html><head><title>ARDUINO Controllo Led via WEB</title>");
        client.println("<meta http-equiv='Content-Type' content='text/html; charset=iso-8859-1' >");
        client.println("<link rel='stylesheet' type='text/css' href='http://www.maurodeberardis.it/Codici
PHP/mycssarduino.css'/>");
        // la riga precedente che ho usato per migliorare la pagina web, si può togliere
        client.println("</head>");
```

```
        // body
         client.println("<body>");
         client.println("<h1>Benvenuti nel Server Arduino</h1>");
         client.println("<p style=';color:red'>Temperatura corrente = ");
         client.print(Temperatura);
         client.println(" gradi <br /></p>");
         client.println("<p style='color:blue'>Luminosit&agrave; corrente = ");
         client.print(Luminosita);
         client.println("<hr />");
         client.println("<p><a target='_blank' href='http://www.maurodeberardis.it'>Sito del Prof. Mauro D
e Berardis</a></p>");
         client.println("</body></html>");

         //readStringa="";//pulizia della stringa per la successiva  lettura

          client.stop();//stop al client

        }
    }
  }
}

}
```

Test del progetto