Arduino e internet: utilizzare **Xively.com**

Arduino ha numerosi modi per collegarsi ad internet. In questo e nei prossimi tutorial analizziamo le varie soluzioni per uplodare dati su un sito, scaricarli, inviare tweet, etc...
Nasce in questi anni il concetto di Internet of things (internet delle cose) ovvero la possibilità di rendere interattivi e intelligenti oggetti comuni. La connessione alla rete permette di monitorare sensori a distanza e Arduino è in grado di accedere questi dati e trasformarli in una determinata azione. Uno degli esempi più noti di questi anni è una sveglia che, collegandosi con un sito che individua il traffico stradale, suona mezz'ora prima per avvisarci di un probabile ritardo.
In questo primo tutorial utilizziamo un nuovo sistema di trasferimento dati online, Xively (ex Patchube ed ex Cosm), che ci permette di uplodare i dati di un sensore collegato al nostro Arduino e di poterli scaricare con un secondo Arduino collegato allo stesso sito.

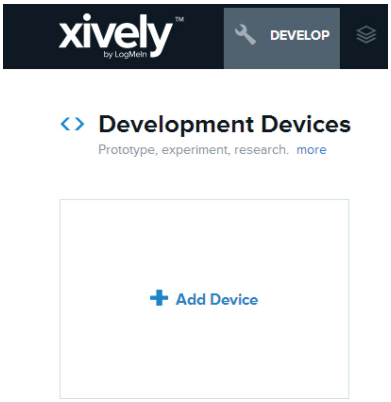
Colleghiamoci su www.xively.com

Xively è un servizio gratuito che permette di creare uno spazio in rete per la trasmissione di dati da un device all'altro, oppure di utilizzare dati di altri utenti per le nostre applicazioni smartphone, siti web, Arduino, etc..
In questo primo tutorial vediamo come registrarci, configurare il nostro Arduino per inviare su Xively i dati di un sensore, configurare un secondo Arduino per scaricare gli stessi dati dalla rete e fare accendere o spegnere un led in base ai dati che riceviamo.
A cosa serve tutto questo?
Se ipotizziamo che i due Arduino sono posizionati in due luoghi del mondo differenti e sono in grado di scambiarsi dati con un scarto di qualche secondo, le applicazioni che possiamo realizzare sono infinite.
Oltre allo scambio di dati dal sito di Xively siamo in grado di scaricare anche dati provenienti da sensori di altre persone. In questo modo è possibile fare accendere una lampada in base all'inquinamento atmosferico di Milano o in base alle radiazioni di Fukushima in Giappone.

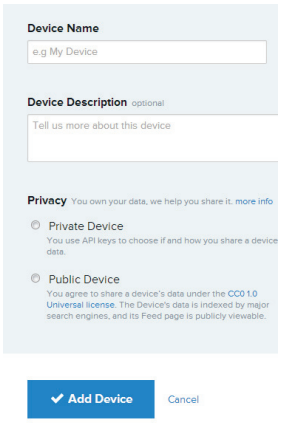
Iniziamo il tutorial.
Materiale utilizzato:
- 2 arduino ethernet o 2 arduino UNO con shield WiFi o 2 arduino UNO con shield ethernet.
- un led e qualche cavetto.

Installiamo le librerie di Xively per Arduino, per comodità ho inserito tutte le librerie necessarie al tutorial scaricabili da questo link: clicca qui.
Una volta scaricato il file zip, scompattiamo e copiamo tutte le cartelle nella cartella libraries di Arduino (per maggiorni informazioni leggere il primo tutorial).

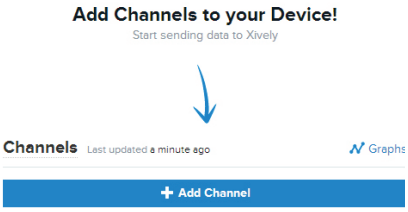
Colleghiamoci al sito e registriamoci su **SIGN UP**, in alto a destra.
Compiliamo tutti i campi, aspettiamo l'email di conferma e poi clicchiamo su **LOGIN** per effettuare l'accesso.



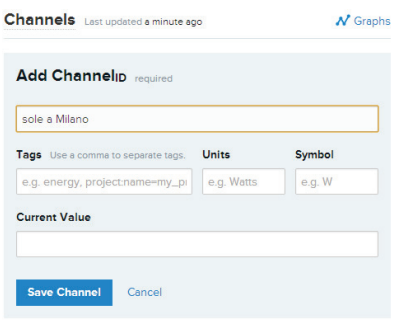
Una volta effettuato l'accesso a Xively click su **Develop** e poi **+ add Device**.
In questa prima fase stiamo registrando il nostro device che si occuperà di uplodare i dati ricevuti dal sensore collegato ad Arduino.



Inseriamo un nome (ad esempio test sensori), la descrizione e selezioniamo se i nostri dati in rete devono essere pubblici o privati.

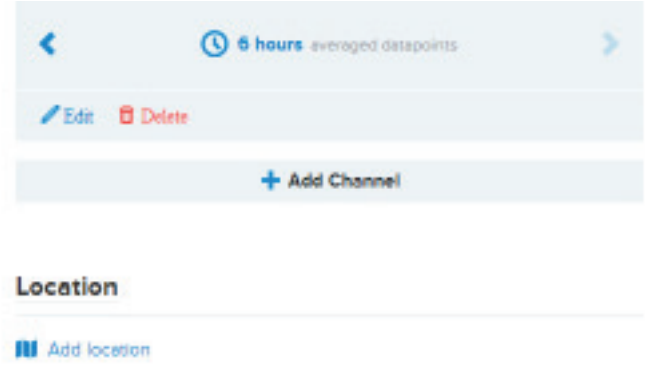


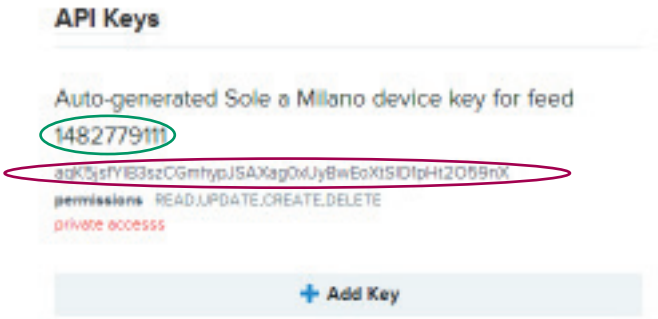
Nella schermata successiva dobbiamo aggiungere dei canali al nostro device. Un canale è il nome del nostro sensore. Es. all'interno del Device "Test sensori" possiamo inserire il canale "luce a Milano", "vento a Milano", "pressione a Milano", etc..
Per questo tutorial creiamo un canale con scritto "soleaMilano" e inserite la vostra città (non è possibile inserire nomi con spazi vuoti, caratteri particolari, etc...).
Click su + Add Channel



Inseriamo il nome e clicchiamo su **Save Channel**.

Nell'immagine qui sotto il riepilogo del canale appena creato. In evidenza i valori che dobbiamo copiare nello sketch di Arduino.





Prendiamo il nostro Arduino Ethernet o Wifi e colleghiamo un sensore al pin 2 (se abbiamo un piccolo pannello solare possiamo collegare il segnale negativo al pin GND di Arduino e il segnale positivo al pin analogico 2. In questo modo rileviamo la quantità di luce solare presente nella nostra città. Se non abbiamo nessun sensore possiamo creare un collegamento tra il segnale 3v di Arduino e il pin2. In questo modo simuliamo un sensore, nel pin2 leggeremo un valore uguale a 705, se spostiamo il collegamento con il pin 5v leggeremo un valore uguale a 1023.
Apriamo il programma di Arduino e clicchiamo su Files - Examples - Xively - **DatastreamUpload** (se stiamo utilizzando la wifi shield dobbiamo aprire lo sketch WiFiDatastreamUpload).
All'interno dello sketch dobbiamo cambiare alcuni parametri:

UPLODARE DATI


```
#include <SPI.h>
#include <Ethernet.h>
#include <HttpClient.h>
#include <Xively.h>

// MAC address for your Ethernet shield
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

// Your Xively key to let you upload data
char xivelyKey[] = "YOUR_XIVELY_API_KEY";

// Analog pin which we're monitoring (0 and 1 are used by the Ethernet shield)
int sensorPin = 2;

// Define the strings for our datastream IDs
char sensorId[] = "sensor_reading";
XivelyDatastream datastreams[] = {
  XivelyDatastream(sensorId, strlen(sensorId), DATASTREAM_FLOAT),
};
// Finally, wrap the datastreams into a feed
XivelyFeed feed(15552, datastreams, 1 /* number of datastreams */);

EthernetClient client;
XivelyClient xivelyclient(client);

void setup() {
  // put your setup code here, to run once:
```

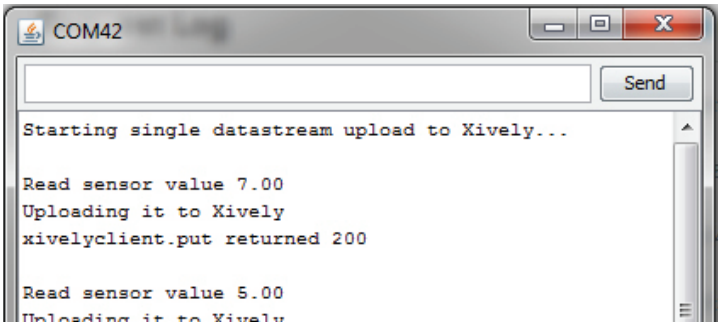
char xivelyKey[] = "YOUR_XIVELY_API_KEY";
Inseriamo la Api Key generata sul sito di xively (nella pagina precedente segnata con il colore viola

char sensorId[] = "sensor_reading";
Inseriamo il nome del canale

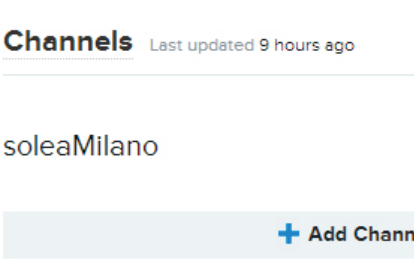
XivelyFeed feed(15552, datastreams, 1 /* number of datastreams */);
Inseriamo il numero Feed. (nella pagina precedente segnato in blu)

```
// Your Xively key to let you upload data
char xivelyKey[] = "aqK5jsfYIB3szCGmhypJSAXagOxUyBwEoXtSlDlpHt2059nX";

// Analog pin which we're monitoring (0 and 1 are used by the Ethernet shield)
int sensorPin = 2;

// Define the strings for our datastream IDs
char sensorId[] = "soleaMilano";
XivelyDatastream datastreams[] = {
  XivelyDatastream(sensorId, strlen(sensorId), DATASTREAM_FLOAT),
};
// Finally, wrap the datastreams into a feed
XivelyFeed feed(1482779111, datastreams, 1 /* number of datastreams */);
```

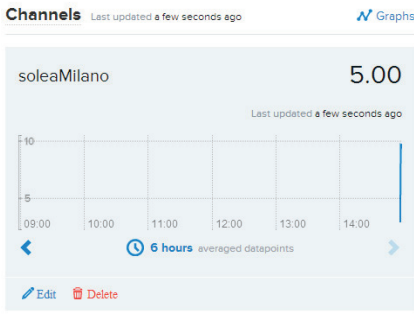
A sinistra il codice modificato con i miei dati.
Possiamo ora uplodare lo sketch.
(* per uplodare lo sketch su una scheda Ardunio Ethernet dobbiamo utilizzare un componente esterno perchè la scheda non ha l'ingresso USB.
(More info: http://schianorobotics.altervista.org/Ethernet_Shield.pdf)
Colleghiamo Arduino alla presa Lan di internet.



Apriamo il serial monitor di Arduino per verificare se stiamo caricando i dati del sensore.
La scritta **Read sensor value 7.00** si riferisce alla quantità di luce rilevata dal pannello solare (se al posto del sensore avete utilizzato un filo collegato a 3v il valore sarà circa 675).
Uploading it to Xively il valore sta per essere uplodato su xively.



Torniamo su sito di Xively e clicchiamo sul nome del nostro canale.
Se tutto è andato bene nella nuova schermata appariranno gli stessi dati presenti sul monitor di Arduino.



Ogni tre secondi (o qualche secondo in più a seconda della velocità della rete internet) la pagina si aggiorna in automatico caricando i dati del sensore di Arduino.
Per continuare con il tutorial scolleghiamo il nostro Arduino dal nostro computer ma non dal cavo Lan. Colleghiamolo con una alimentazione esterna, batterie, o collegandolo semplicemente a un secondo computer. I dati continueranno ad essere uplodati. Ora possiamo collegare e programmare il secondo Arduino al nostro computer e a un secondo cavo Lan.

DOWNLOAD DATI

Scarichiamo ora i dati che stiamo uplodando con il nostro primo Arduino.
Nel caso che i due Arduino si trovino collegati alla stessa rete dobbiamo ricordarci di modificare il MAC address della seconda scheda Arduino per non farla andare in conflitto con la prima. Nel caso i due Arduino sono collegati in due reti differenti non dobbiamo modificare niente.
Apriamo il programma di Arduino e clicchiamo su Files - Examples - Xively - **DatastreamDownload** (se stiamo utilizzando la wifi shield dobbiamo aprire lo sketch WiFiDatastreamDownload).

```
// MAC address for your Ethernet shield
byte mac[] = { 0xAA, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
int led6 =6;
// Your Xively key to let you upload data
char xivelyKey[] = "aqK5jsfYIB3szCGmhypJSAXag0xUyBwEoXtSlDlpHt2059nX";

// Define the string for our datastream ID
char temperatureId[] = "soleaMilano";

XivelyDatastream datastreams[] = {
  XivelyDatastream(temperatureId, strlen(temperatureId), DATASTREAM_FLOAT),
};
// Finally, wrap the datastreams into a feed
XivelyFeed feed(1482779111, datastreams, 1 /* number of datastreams */);

EthernetClient client;
XivelyClient xivelyclient(client);
```

Qui sotto le modifiche che dobbiamo apportare al codice.

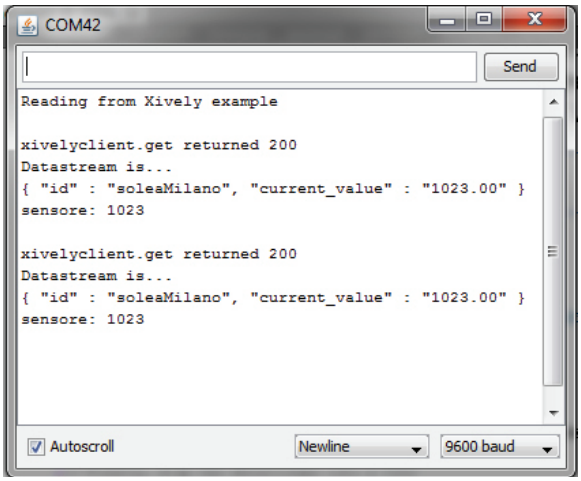
byte mac[]= dobbiamo inserire un nuovo indirizzo per differenziarlo da quello utilizzato nell'Arduino che uploda i dati). Inseriamo questo indirizzo:
0xAA, 0xAD, 0xBE, 0xEF, 0xFE, 0xED

char xivelyKey[] = "YOUR_XIVELY_API_KEY";
Inseriamo la Api Key generata sul sito di xively

char sensorId[] = "sensor_reading";
Inseriamo il nome del canale

XivelyFeed feed(15552, datastreams, 1 /* number of datastreams */);
Inseriamo il numero Feed.

A questo punto abbiamo un Arduino collegato a un sensore che sta uplodando dati su Xively. Il nostro secondo Arduino lo colleghiamo al secondo cavo Lan e uplodiamo il codice modificato.
Se apriamo il Serial Monitor di Arduino possiamo verficare se la trasmissione dati sta funzionando. Nel monitor devono apparire gli stessi dati presenti su Xively.



Se tutto è andato bene dopo la scritta "sensore:" (oppure "Temperature:" basta modificare la scritta nel codice) appare il valore del sensore collegato al primo Arduino.
Possiamo fare delle prove coprendo il nostro pannello solare (oppure spostando il cavetto collegato al Pin2 da 5v a 3v oppure mettendolo su GND) e vedere in quanto tempo il dato viene trasferito su Xively e poi sul monitor del secondo Arduino.

Qui a sinistra le due schermate del monitor di Ardunio e il sito di Xively che si scambiano i dati.

Per completare questo primo tutorial di Xively facciamo compiere un'azione all'Arduino che riceve i dati. Se il primo Arduino riceve dal sensore un valore più grande di 70 allora il secondo Arduino che preleva questo dato da Xively (quindi potrebbe stare dall'altra parte del mondo) accende un led.
Per realizzare questo analizziamo e modifichiamo la patch **DatastreamDownload**.

```
void loop() {
  int ret = xivelyclient.get(feed, xivelyKey);
  Serial.print("xivelyclient.get returned ");
  Serial.println(ret);

  if (ret > 0)
  {
    Serial.println("Datastream is...");
    Serial.println(feed[0]);

    Serial.print("Temperature is: ");
    Serial.println(feed[0].getFloat());
  }

  Serial.println();
  delay(15000UL);
}
```

Se la connessione è andata a buon fine stampo sul monitor di Arduino "Datastream is..." e subito dopo il valore del sensore "Temperature is:".

Da questo codice possiamo capire che il valore presente su Xively che vogliamo scaricare è presente all'interno della variabile **feed[0].getFloat()**

Creiamo due nuove variabili, la prima serve per accendere un led e la chiamiamo **led6**, la seconda serve per leggere il valore del sensore e la chiamiamo **sensore**.

```
// MAC address for your Ethernet shield
byte mac[] = { 0xAA, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

int led6 =6;
int sensore=0;
```

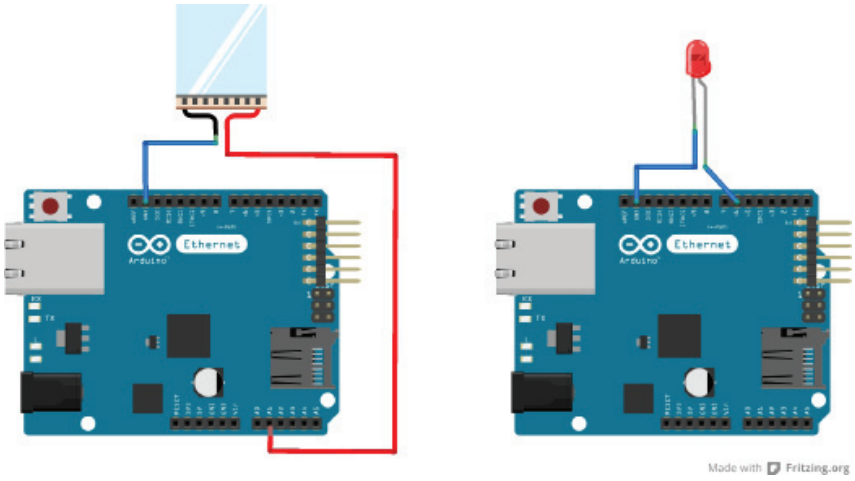
int led6 =6;
int sensore=0;

Nella funzione loop inseriamo una lettura della variabile sensore e poi facciamo un controllo su questa. Se il valore è più grande di 70 allora accendi il led, altrimenti spegni il led.
Il codice a fianco è il codice modificato che permette di gestire il valore del sensore proveniente dal primo Arduino.

```
void loop() {
  int ret = xivelyclient.get(feed, xivelyKey);
  Serial.print("xivelyclient.get returned ");
  Serial.println(ret);

  if (ret > 0)
  {
    Serial.println("Datastream is...");
    Serial.println(feed[0]);

   Serial.print("sensore: ");
    sensore=(feed[0].getFloat());
   // Serial.println(feed[0].getFloat());
   Serial.println(sensore);
    if (sensore >70){
    digitalWrite(led6,1);
        }
        else{
         digitalWrite(led6,0);
        }
  }

  Serial.println();

}
```

Con questo semplice esempio siamo in grado di far comunicare due Arduino a grandi distanza tramite una connessione internet (Lan o Wifi). Qui sotto lo schema elettrico dei due componenti.

Nota bene: la libreria di Xively che permette questa comunicazione di dati ha un bug al suo interno. Se scarichiamo la versione originale dal sito di Xively ci accorgiamo che nella fase di download del dato, tra un Arduino e l'altro, passa più di un minuto.
Nel blog di Xively **nik8989** ha postato la soluzione. Per i più esperti potete leggere le specifiche della modifica a questo link: https://github.com/xively/xively_arduino/issues/1.
Se avete scaricato le librerie dal mio sito questo errore è stato già risolto.
Ricordatevi che se in futuro avete bisogno di installare tutte le librerie di Arduino in seguito a una formattazione e scaricate le librerie dal sito di Xively potreste avere questo problema.



Arduino 1
upload

Arduino 2
download