Arduino – cereri HTTP de tip GET si POST

Despre HTTP.

HTTP este un protocol de comunicare intre un client si un server web. In cazul acestui protocol comunicatia incepe de la client, dar sunt si situatii in care server-ul doreste sa initieze comunicatia cu clientul, cum vei descoperi in continuare.

In urma unei cereri HTTP prin care un client initiaza o comunicatie cu un server, acesta din urma proceseaza cererea si trimite un raspuns inapoi clientului (dar sunt situatii in care nu este obligatoriu sa trimita raspuns, ci o proceseaza si atat).

HTTP, alaturi de alte protocoale este foarte cunoscut deoarece opereaza la nivelul 7 (nivelul Aplicatie) din modelul OSI.

Pentru a afla mai multe detalii despre acest protocol sau despre modelul OSI acceseaza link-urile de mai jos. Sunt foarte utile deoarece te ajuta sa iti aprofundezi cunostintele despre retelistica, tipurile de protocoale ce opereaza in spatele unei conexiuni de internet, tipurile de retele ce se pot construi, s.a.m.d.

http://support.microsoft.com/kb/103884

http://www.webopedia.com/quick_ref/OSI_Layers.asp

http://en.wikipedia.org/wiki/OSI_model

http://compnetworking.about.com/cs/designosimodel/g/bldef_osi.htm

GET si POST ?

Pana aici ai inteles ca un client initiaza o comunicatie HTTP cu un server web. Exista 2 tipuri de cereri pe care un client le poate lansa catre un server si anume: GET si POST.

La ce te ajuta pe tine aceste cereri ? Cererile GET si POST te vor ajuta pe tine sa publici informatii pe Internet, mai exact sa programezi o placa Arduino sa citeasca „ceva“ si sa faca vizibila aceasta informatie pe Internet.

Ca si exemplu, sa ne gandim la o statie meteo construita cu o placa Arduino. Statia meteo este alcatuita dintr-o multitudine de senzori ce sunt responsabili cu masurarea: temperaturii, presiunii, umiditatii, vitezei/directiei vantului, cantitatii de ploaie, indicelui de confort termic. Asta este doar un exemplu, dar sunt foarte multe situatii la care trebuie sa publici informatiile pe internet.

Urmatoarea intrebarea care se poate naste este: cum public aceste informatii pe internet si cum pot sa le vizualizez, ce pot sa fac cu ele sau cum le stochez ?

Cererile GET si POST sunt doar o parte a raspunsului. Aceste cereri te ajuta sa initiezi comunicatia intre statia meteo si server-ul web. Mai departe server-ul web proceseaza informatiile, le publica pe o pagina web si le salveaza intr-o baza de date.

Server-ul web proceseaza cererile printr-o pagina scrisa in PHP. Pagina PHP initiaza conexiunea cu baza de date, unde printr-un alt protocol salveaza informatiile procesare prin GET sau POST. Mai departe, o alta pagina initiaza o conexiune cu baza de date, preia informatiile si le afiseaza sub forma unui grafic.

Tocmai am descris, sumar, un serviciu de procesare a datelor online (mai este cunoscut sub serviciu Cloud).

Totusi sunt 2 cereri si trebuie sa alegi doar una, dar inainte de a face lucrul asta trebuie sa vezi din ce este compusa o cerere:

- in primul rand resursa solicitata de client. In cazul statiei meteo, resursa este o cale catre un fisier PHP, fisierul care proceseaza cererea. (exemplu: /meteo/preiadate.php)
- tipul cererii: GET sau POST.
- Headere HTTP: sunt informatii standard transmite de client catre server. Practic aceste informatii ajuta la identificarea clientului.
- Informatiile statiei meteo (temperatura, presiune, umiditate, s.a.m.d).

Ca sa intelegi cat mai usor trebuie sa te gandesti la un exemplu. Revenind la statia meteo, aceasta este programata sa publice datele printr-un server. Statia meteo initiaza cereri la intervale regulate de timp (spre exemplu, o data pe minut sau o data la 10 minute) prin GET sau POST. Practic statia meteo specifica serverului urmatoarele lucruri: pagina php care proceseaza temperatura, umiditatea, etc; ii spune serverului la ce tip de cerere sa se astepte, ii da de asemenea detalii despre propria identitate (cine sunt eu, cu ce host vreau sa comunic) si in final ii transfera informatiile (temperatura, umiditate, etc).

Daca statia meteo a realizat acest lucru si server-ul web preia corect datele, poti sa spui ca ai reusit sa publici cu succes datele pe un server web.

Cum arata o cerere POST scrisa in cod Arduino ?

Iata mai jos cum o placa Arduino poate sa initializeze o cerere POST cu un server web. Asa cum am explicat mai sus despre structura cererii HTTP, se poate observa acelasi lucru si in structura cereri POST.

Mai exact, daca conexiunea cu serverul s-a realizat cu succes se va executa toata structura de cod cuprinsa intre acolade. Clientul ii spune serverului la ce tip de cerere sa se astepte urmata de adresa paginii php, apoi ii transmite mai multe detalii despre el insasi (adresa host-ului, tipul agentului: Arduino, ce ar trebui sa faca server-ul dupa ce primeste datele adica sa inchida conexiunea, formatul headerului http, dimensiunea informatiilor si in final informatia utila).

```
if (client.connect(server,80)) { // daca s-a conectat cu succes
    Serial.println("connected"); // afiseaza pe Monitorul Serial
    client.println("POST /pagethattakesdata.php HTTP/1.1"); //
transmite tipul cererii si adresa paginii care proceseaza cererea
    client.println("Host: 192.168.2.1"); // adresa host-ului
    client.println("User-Agent: Arduino/1.0"); // detalii despre
client
    client.println("Connection: close"); // inchide conexiunea
    client.println("Content-Type: application/x-www-form-
urlencoded"); // formatul headerului http
    client.print("Content-Length: "); // dimensiunea informatiilor
    client.println(sizeof(date));
    client.println();
    client.println(String(date)); // informatiile in sine sub forma
de String
    client.println();
    client.flush(); // inchide conexiunea
    client.stop();
}
```

Cum proceseaza o pagina PHP cererea de mai sus ?

```
<?php
$file = 'date.txt';
$current = file_get_contents($file);
if (isset($_POST['data'])) {
  $t = $_POST['data'];
  $current .= $t;
  file_put_contents($file, $current);
} else {
  $t = 0;
  $current .= $t;
  file_put_contents($file, $current);
}
?>
```

Ce se intampla cu codul php de mai sus ? La fiecare cerere lansata de statia meteo (placa Arduino) codul este executat si realizeaza urmatorul lucru: deschide un fisier text (a doua linie), preia continutul (a treia linie), verifica daca exista o cerere POST care incepe cu 'data' (a patra linie), daca exista atunci codul salveaza toata informatia intr-o variabila iar mai apoi variabila este stocata in continutul fisierului si apoi rescris in fisierul text (liniile 5, 6 si 7).

Daca exista cereri dar care nu sunt identificate cu 'data', atunci codul scrie un 0 in fisierul text. Acel 0 poate fi interpretat ca si eroare, daca este cazul.

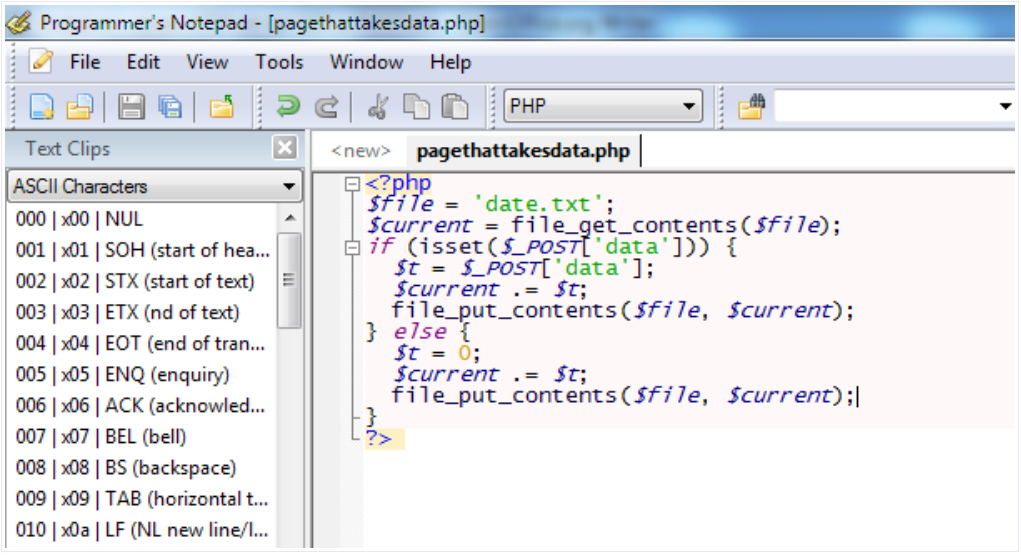
Cum testez codul de mai sus ?

Primul lucru pe care trebuie sa il faci este sa instalezi un server Apache, un compilator PHP, o si o baza de date MySQL. Pentru simplitate iti recomand utilitarul XAMPP, pe care il poti descarca si instala de la adresa de mai jos:

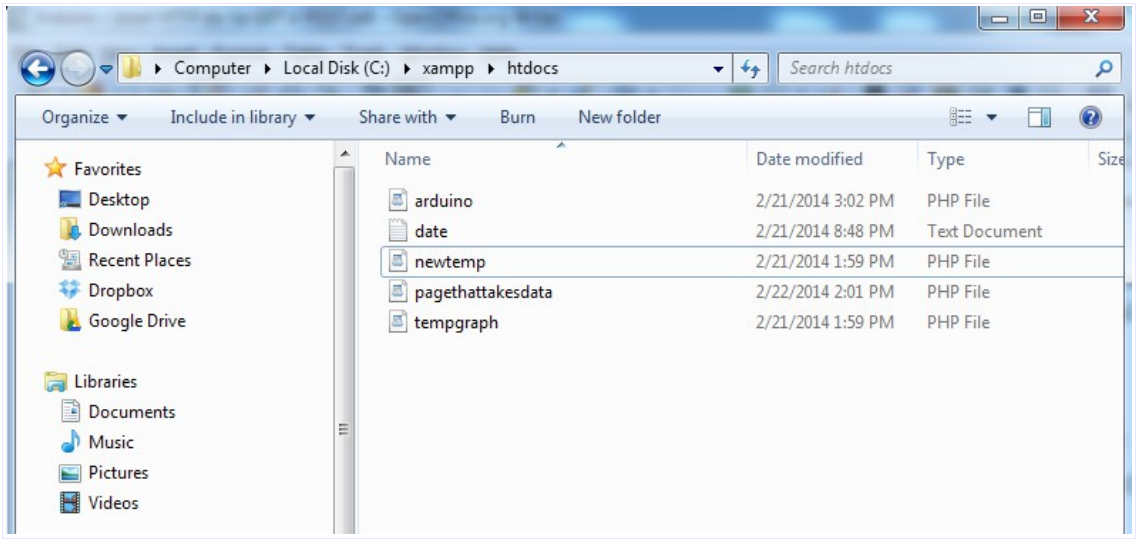
http://www.apachefriends.org/ro/index.html

Ai la dispozitie varianta pentru Windows, Linux si Apple.

Descarca utilitarul Notepad++ si creeaza o pagina PHP. Copiaza codul sursa de mai sus in pagina PHP.

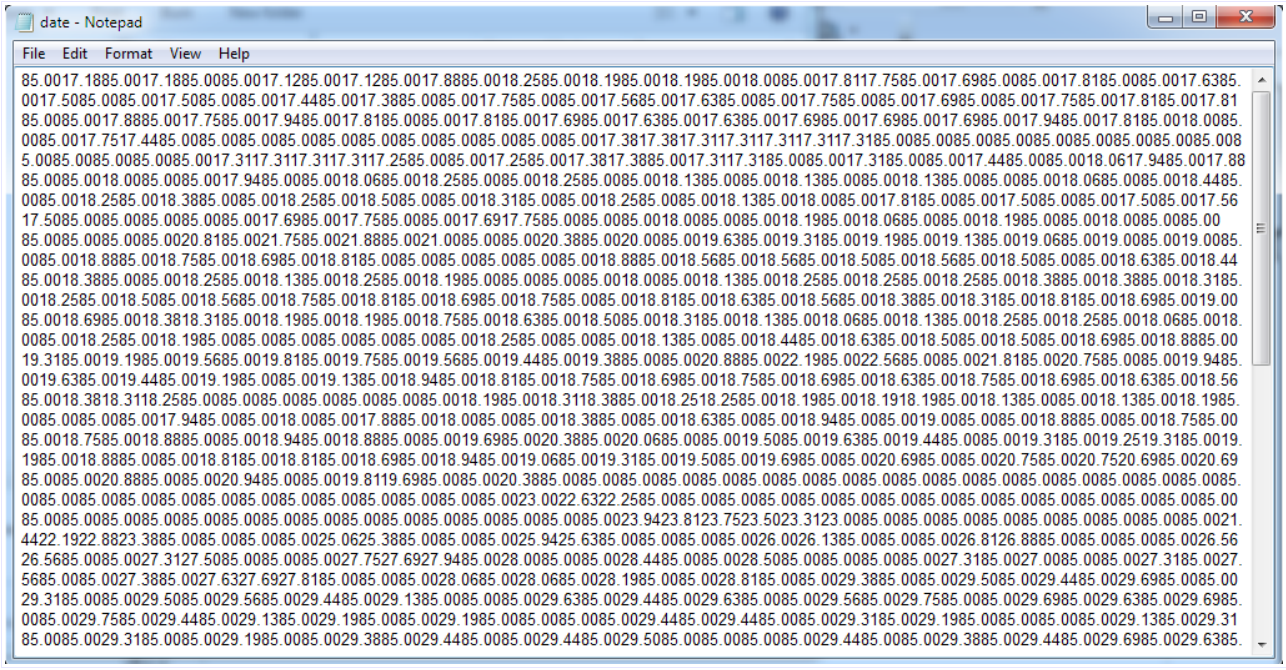


Pagina PHP nou creata trebuie salvata la urmatoarea adresa unde ai instalat utilitarul xampp, adica /xampp/htdocs/.



Daca server-ul este pornit si configurat corect, acesta poate primi cereri de la placa Arduino. Asa cum am spus mai sus, tot ce face pagina php care serveste cererile este sa le salveze intr-un fisier text.

Daca codul de mai sus functioneaza cu succes, fisierul text va arata sub urmatoarea forma:



Tot ce vezi acolo este o succesiune de valori salvate pe spatiul serverului, valori generate de o placa Arduino avand un shield Wifi conectat in pinii acesteia si un senzor de temperatura care la anumite intervale de timp genereaza si o eroare (valorile 85 prezente in fisier).

Ce poti realiza tu in continuare ?

Afla cum poti salva aceste valori intr-o baza de date ! De asemenea afla cum poti sa generezi un grafic pe baza unui fisier text continand informatii cu privire la temperatura, umiditate, presiune.

Poti sa iti creezi un sistem dinamic de prelucrare a datelor studiind link-urile de mai jos:

http://blog.protoneer.co.nz/arduino-http-post-requests/

http://www.codetorment.com/2009/11/12/arduino-temperature-and-light-data-logging-and-chartplotting-webmonitor/

http://forum.arduino.cc/index.php?topic=155218.0

https://github.com/ericbenwa/POST-Arduino-Data-Wireless