Monitor consumi elettrici con Fishino: "*Fish and KiloWatt*"

Introduzione

Il progetto permette di realizzare un monitor dei consumi elettrici che consente di:

1. leggere in tempo reale la potenza istantanea consumata; dati aggiornati ogni 5 secondi.
2. registrare su file in formato CSV, il log dei consumi orari, giornalieri e mensili
3. visualizzare in tempo reale, l'andamento dei consumi su grafico aggiornato ogni 5 secondi (modalità *Live View*) *
4. scaricare da browser i file di log
5. visualizzare su browser, i grafici dei consumi registrati sui file di log *
6. essere avvisati del superamento di una soglia prestabilita di consumo istantaneo, tramite APP Pushetta (disponibile gratuitamente sia per dispositivi Android, che IOS) *
7. verificare il superamento di una soglia prestabilita da browser, grazie al cambiamento di colore dello sfondo.
8. Ripetizione su uscita digitale 2, di un impulso in corrispondenza di ogni watt consumato.

* Le funzionalità 3,5 e 6, sono disponibili solo in modalità wifi

Con il progetto proposto, si potrà quindi mettere sotto osservazione i consumi elettrici dei singoli elettrodomestici conoscendone in ogni momento l'effettiva potenza assorbita.
Si potrà inoltre, analizzare lo storico dei consumi per individuare, ed eventualmente correggere, l'utilizzo dei vari elettrodomestici e l'incidenza in *stand-by.*

Il principio di funzionamento, si basa sul fatto che a ogni flash del led superiore del contatore elettrico, corrisponde il consumo di un watt di potenza attiva
Se riusciamo a far riconoscere i flash a Fishino, sarà sufficiente calcolare il tempo che intercorre fra due flash, per poter risalire alla potenza istantanea consumata in quell'istante.

Approfondimento sul principio di funzionamento

Il principio di funzionamento, si basa sul fatto che a ogni flash del led superiore del contatore elettrico, corrisponde il consumo di un watt di potenza attiva
Se riusciamo a far riconoscere i flash a Fishino, sarà sufficiente calcolare il tempo che intercorre fra due flash, per poter risalire alla potenza istantanea consumata in quell'istante.

Se ad esempio rilevassimo un flash ogni due secondi, avremo 30 flash al minuto, che corrispondono a 60 X 30 = 1800 flash in un'ora, che equivalgono a un consumo di 1,8 kWattora
Se il contatore lampeggiasse più velocemente, ad esempio un flash al secondo, avremo 60 flash al minuto pari a 3600 flash in un ora, che equivalgono a un consumo di 3,6 kWattora

Utilizzando la funzione millis(), siamo in grado di rilevare il numero di millisecondi da quando la scheda Fishino ha iniziato l'esecuzione del programma. Se salviamo in una variabile il risultato della funzione millis() calcolata in corrispondenza di due flash consecutivi, possiamo calcolare per differenza i millisecondi intercorsi per il consumo di un watt. Nel programma, la variabile si chiama Gftinterval e la formula per calcolare i Wh è;

(3600 X 1000) : Gftinterval

```
GFtinterval = (GLtflash - GLtoldflash) ;
GLtoldflash = GLtflash;
GFPotIstantanea = 3600/GFtinterval*1000; //Calcolo la potenza istantanea
```

Se ho un flash ogni due secondi, Gftinterval sarà uguale a 2000 ms
Applicando la formula si ottiene (3600 X 1000) : 2000 = 1800 = 1,8 KW/h

Per poter rilevare un flash, realizziamo un semplice partitore di tensione collegando la fotoresistenza ai +5V, e chiudendo il circuito a massa tramite la resistenza da 47 Kohm. Il punto centrale del partitore verrà applicato all'ingresso analogico A0 e con la funzione analogRead(A0) potremo monitorare la variazione di tensione provocata dal flash del led.

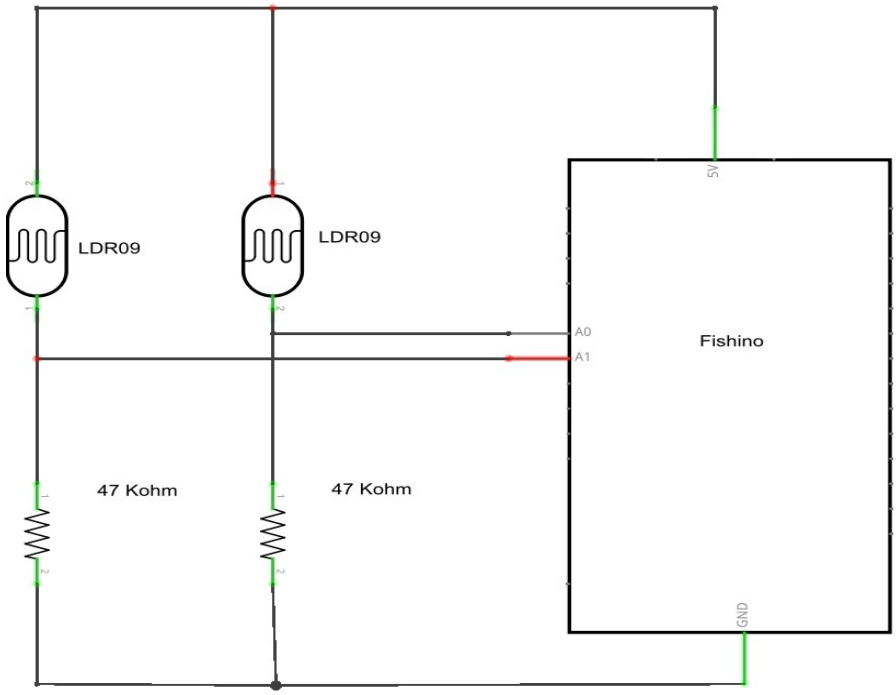
Nel mio caso, il contatore è posizionato in un garage dove sono presenti delle finestre che danno su una strada. Il valore letto con analogRead(A0) è quindi influenzato sia dall'accensione o meno dell'illuminazione del garage, sia dal variare durante il corso del giorno della quantità di luce che filtra dalle finestre nonché dal passaggio di auto con i fari accesi.
Inoltre, sul contatore è presente anche un secondo led che lampeggia ogni volta che si è consumato un watt di potenza reattiva. La potenza reattiva non entra nel calcolo della bolletta, e pertanto dobbiamo ignorare tale lettura. Questo flash, però, si propaga tramite il plexiglas che ricopre il contatore e può interferire con la lettura del led della potenza attiva.
Per risolvere questi problemi, ho aggiunto una seconda fotoresistenza da posizionare in corrispondenza del led inferiore e, invece di rilevare un flash sulla base della lettura di un valore assoluto, opero una lettura differenziale fra le due fotoresistenze.
In pratica, tutte le condizioni di disturbo sopra elencate si manifestano in egual misura sulle misurazioni delle due fotoresistenze. Facendo la differenza dei valori rilevati, otterrò un valore positivo se sto rilevando un flash dal led superiore, negativo se sta lampeggiano il led inferiore, e un valore prossimo a zero se non ho nessun flash.
Con questo semplice accorgimento, riduciamo drasticamente le possibilità di errore nel rilevare un flash.

Il circuito elettrico

Per realizzare il progetto, sono sufficienti due fotoresistenze, e due resistenze.
In particolare, l'apparato si compone di due fotoresistenze (LDR09), e due resistenze da 47 Khom.
Le due fotoresistenze devono essere collegate ai +5V, e devono chiudersi a massa tramite una resistenza da 47 Khom.
Il punto centrale del partitore riferito al led superiore (potenza attiva), viene applicato all'ingresso analogico A0.
Il punto centrale del partitore riferito al led inferiore (potenza reattiva), viene applicato all'ingresso analogico A1.
Le fotoresistenze dovranno essere posizionate in corrispondenza dei due led de contatore e fissate, ad esempio, con del semplice nastro adesivo.
Per il mio prototipo, ho inserito le due fotoresistenze in due tubicini neri che poi ho alloggiato in una vecchia lampada con braccio snodabile. In questo modo ho potuto posizionare l'apparato in maniera molto accurata senza intaccare in nessun modo il contatore.
Dal momento che i due led del contatore non saranno più visibili, Fishino farà da ripetitore per quanto riguarda il solo led superiore (quello relativo alla potenza attiva). Durante il campionamento di un flash, Fishino porterà al valore *HIGH* l'uscite digitale 2 alla quale sarà possibile collegare un led e una resistenza da 180 Ohm, per ripristinare l'effetto visivo.

La realizzazione è molto semplice e non è previsto PCB.

Per quanto riguarda l'alimentazione, non ci sono particolari indicazioni in quanto non vi sono carichi esterni da alimentare, o shield aggiuntivi.
Non è richiesto alcun sistema di mantenimento dell'alimentazione in caso di stacco del contatore, in quanto in una tale situazione, non vi sarebbero consumi da monitorare.



Descrizione del software.

Sezione dichiarativa

La sezione dichiarativa presenta le variabili e le funzioni di supporto in questo ordine:

- Gestione modulo wifi (con la possibilità di definire la modalità *standalone* o *normal* come da esempio FishinoWiFiWebServer

- Gestione dell'interfaccia web server

- Gestione della lettura potenza istantanea

  La variabile *GISoglia*, contiene il valore differenziale che indica al programma l'accensione del led superiore. Il valore di default, è stato rilevato in maniera sperimentale ed è impostato a 90. Eventualmente può essere variato.
  La variabile *GIContaFlash*, indica il numero di campionamenti consecutivi con superamento della soglia differenziale, necessari a rilevare un flash valido. Il valore di default, è stato rilevato in maniera sperimentale ed è impostato a 150. Eventualmente può essere variato.

  La funzione CheckFlash() si fa carico di campionare la tensione presente sui due partitori, calcolarne la differenza e confrontarla con il valore *GISoglia.*
  Vengono poi contate le volte consecutive che si supera tale soglia, e se si raggiunge il valore indicato in *GIContaFlash*, si assume di aver rilevato un flash valido.
  In questo caso la funzione restituisce al programma principale il valore rilevato dalla chiamata millis(), altrimenti restituisce 0.

- Gestione delle notifiche tramite Pushetta (con compilazione condizionale in quanto valido solo in modalità *normal*)

  E' possibile configurare la variabile *GLLimite*, impostando il limite in watt superato il quale viene inviata la notifica Pushetta. Il default è 3000, che corrisponde a 3 kW.

  Troviamo poi la definizione dell'API Key associato al proprio account Pushetta, e la definizione del Channel sul quale far confluire la notifica.

  La *Content-Length* è impostata a 74 considerando la lunghezza del messaggio che apparirà sulla notifica Pushetta. Se si volesse variare il messaggio “Attenzione soglia superata”, occorrerà ricalcolare tale dato con la formula 48 + lunghezza messaggio.

  Per ulteriori dettagli, fate riferimento all'articolo su Elettronica In n. 194 di aprile 2015e alla documentazione online (http://www.pushetta.com/)

- Gestione dell'orario e dei nomi file di log.

  Non ho utilizzato la libreria RTClib; dovendo solo leggere l'orario, mi sono interfacciato direttamente ai registri dell'RTC risparmiando molta memoria di programma.

- Il naming dei file di log è il seguente:

  char GLFname_m[] = "aamm.csv"; //File di log mensile. Ad ex. 1604.csv per il mese di aprile 2016
  char GLFname_g[] = "aammgg.csv"; //File di log giornaliero. Ad ex. 160425 per il log del 25 aprile 2016
  char GLFname_h[] = "aammgghh.csv"; //File di log orario. Ad ex. 16042513 per il log dei consumi dalle 13:00 alle 13:59 del 25 aprile 2016

Sezione setup()

La sezione *setup(),* inizializza il modulo wifi, e le varie interfacce: SERIAL, SPI, SD e WIRE.
Procede poi alla lettura dell'ora e all'impostazione delle relative variabili. Vengono anche impostati i nomi dei 3 file di log.
In ultimo, viene attivata l'interfaccia web.

Ciclo principale loop()

Il loop principale svolge in sequenza le seguenti funzioni:

1. Verifica cambio di minuto, ora o giorno.
   In corrispondenza dello scadere di un minuto, ora o giorno, vengono registrate le letture di consumo e successivamente aggiornati i nomi dei file di log,

```
if (  getDateTime()) {
  if ( GLm != GLmOld ) {
  // scaduto un minuto, scrivi su file GLFname_h
  // il numero di watt (GLwm ) consumati nell'ultimo minuto ( GLmOld )
  dataFile = SD.open(GLFname_h, FILE_WRITE);
  if (dataFile) {
    dataFile.print(GLmOld);
    dataFile.print(",");
    dataFile.println(GLwm);
    dataFile.close();
  }
 // azzero i watt consumati nel minuto
  GLwm = 0;
  GLmOld = GLm;
  }
```

2. Campionamento del led superiore del contatore per rilevare il consumo in watt, e calcolare la potenza istantanea.

```
// campiono il led della potenza attiva
 GLtflash = CheckFlash();
 // se rilevato il flash, procedo con il calcolo della potenza istantanea
 if ( GLtflash > 0 )
 {
  GFtinterval = (GLtflash - GLtoldflash) ;
  GLtoldflash = GLtflash;
  GFPotIstantanea = 3600/GFtinterval*1000; //Calcolo la potenza istantanea
  //ho consumato un watt, aggiorno i contatori di consumo
  GLwm++;
```

```
    GLwh++;
    GLwg++;
  }
```

3. Gestione delle chiamate web
   Lo scheletro di questa sezione, deriva da FishinoWiFiWebServer.
   Sono state gestite le seguenti risorse:

   “index” - per servire la pagina iniziale index.htm. La pagina viene creata dinamicamente e la compilazione condizionale esclude dalla risposta i riferimenti non ammessi in modalità standalone.

```
#ifndef STANDALONE_MODE
        client << F("<a  href=\"liveview.htm\">Live View</a><BR>");
        client << F("<a  href=\"chfile.htm\">Grafici Storici</a><BR>");
#endif
```

   “*FileName”* - per l'invio dei file di log CSV. Sia per lo scarica da browser, che per la creazione dinamica dei grafici.

   “leggivalore” - per l'invio in formato json, della potenza istantanea. Questa risorsa è utilizzata nella modalità *Live View.*

   “liveview” - per servire la pagina htm corrispondente alla modalità *Live View: liveview.htm*

   “chfile” - per servire la pagina htm corrispondente alla modalità Grafici Storici*: chfile.htm*

   “logfile” - per servire la pagina htm corrispondente alla modalità Log Storici*: logfile.htm*

4. Gestione notifica Pushetta
   se ho rilevato il primo superamento della soglia, oppure se è passato più di un minuto dall'ultimo superamento della soglia, attivo la notifica.
   Anche in questo caso, ci si avvale della compilazione condizionale per escludere o meno tale chiamata.

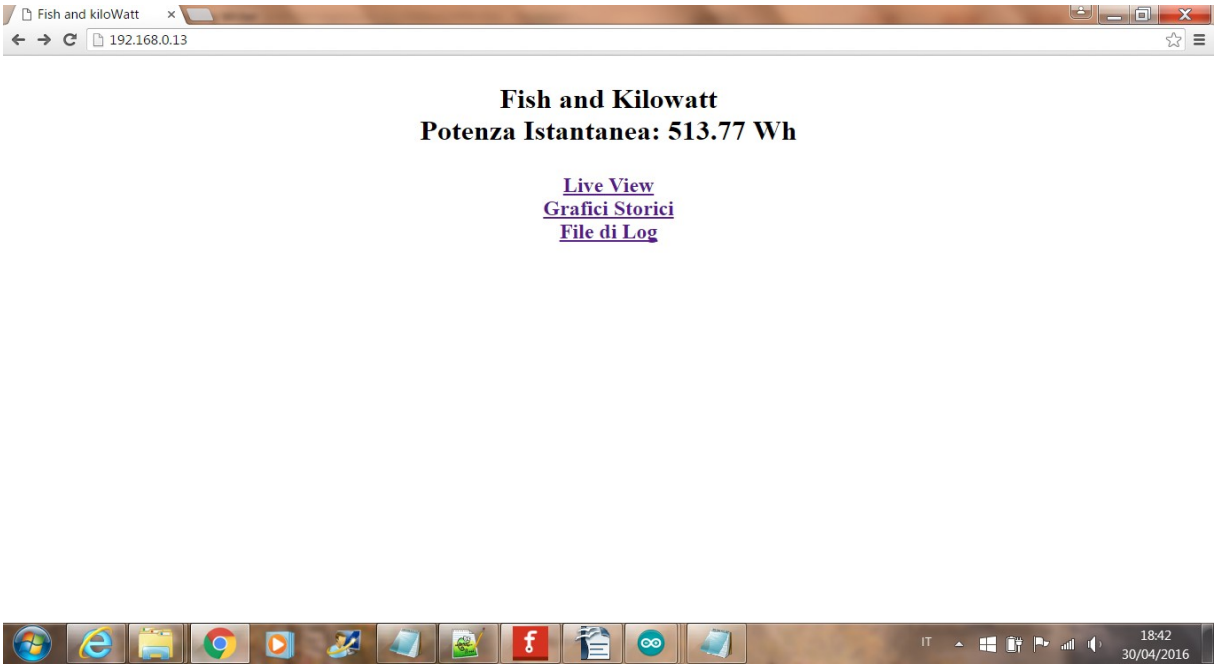
Le pagine HTML.

La struttura delle pagine HTML è la seguente:

index.htm – la pagina principale viene creata dinamicamente direttamente da programma Fishino.

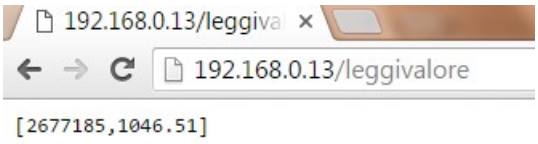
La pagina ha una clausola di refresh impostata a 5 secondi.
Al superamento della soglia prestabilita, la pagina cambia colore.

```
if (GFPotIstantanea > GLLimite)
{
 client << F("<body bgcolor=\"#ff0000\">\n");
}
```



liveview.htm - questa pagina consente di visualizzare l'andamento in tempo reale della potenza istantanea su di un grafico.
Il refresh è di 5 secondi, e si utilizza una chiamata ajax con jQuery alla risorsa *leggivalore*.

Se si interroga con il browser l'indirizzo della nostra scheda Fishino, seguito dalla stringa *leggivalore*, si ottiene la risposta in formato Json che contiene il tempo in millisecondi dall'avvio della scheda, e il valore della potenza istantanea




```
if (StrContains(HTTP_req, "leggivalore")>0) {
   SendHeaderOk();
   client << F("Content-Type: text/json\n\n");
   client << F("[") << millis()  << F(",") << GFPotIstantanea << F("]\n");
  }
```

Per la gestione dei grafici, mi sono appoggiato alle librerie highcharts (http://www.highcharts.com/)

chfile.htm – da questa pagina è possibile visualizzare in formato grafico, i log file salvati su scheda microsd.
È necessario selezionare la data, ed eventualmente anche l'ora, per generare i 3 tipi di grafici: mensile, giornaliero e orario.
Anche in questo caso ci si avvale delle librerie highcharts e a livello Fishino, si fa riferimento alla risorsa *"FileName="*

Grafico Mensile (riferito ad aprile 2016)

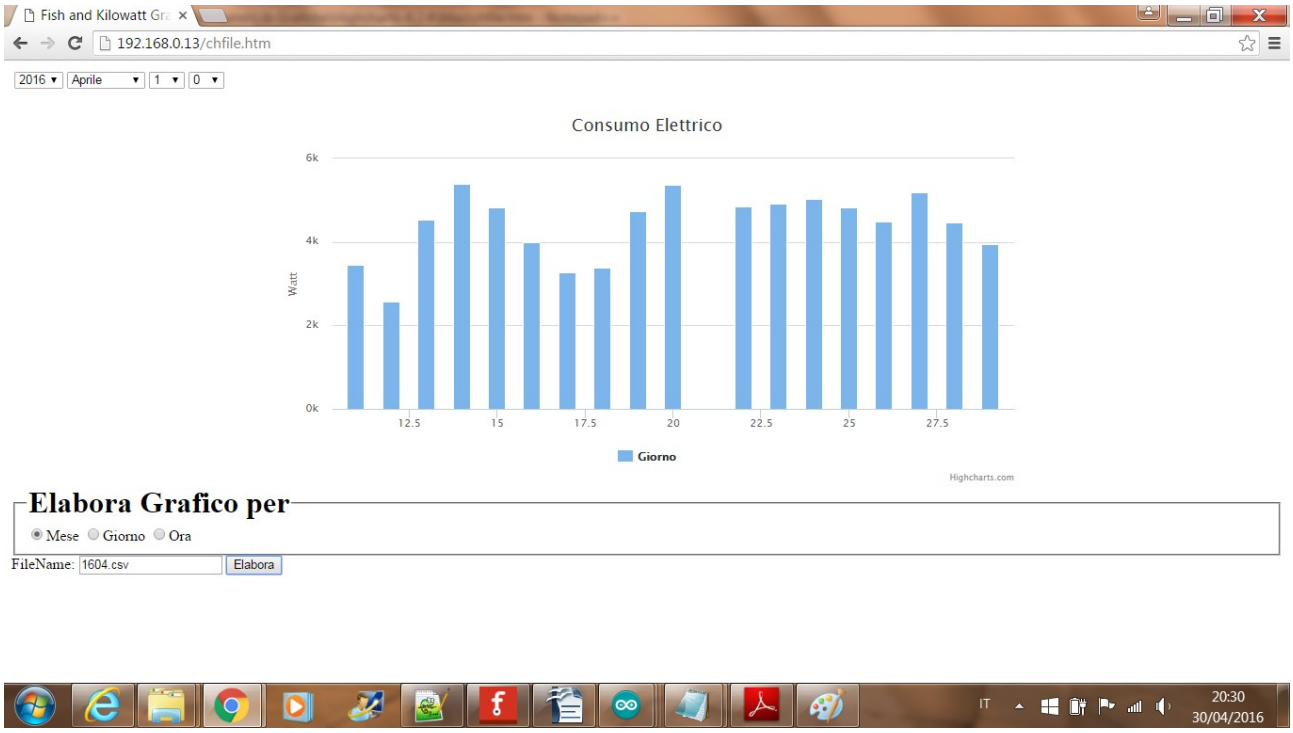


Grafico Giornaliero (riferito al 23 aprile 2016)

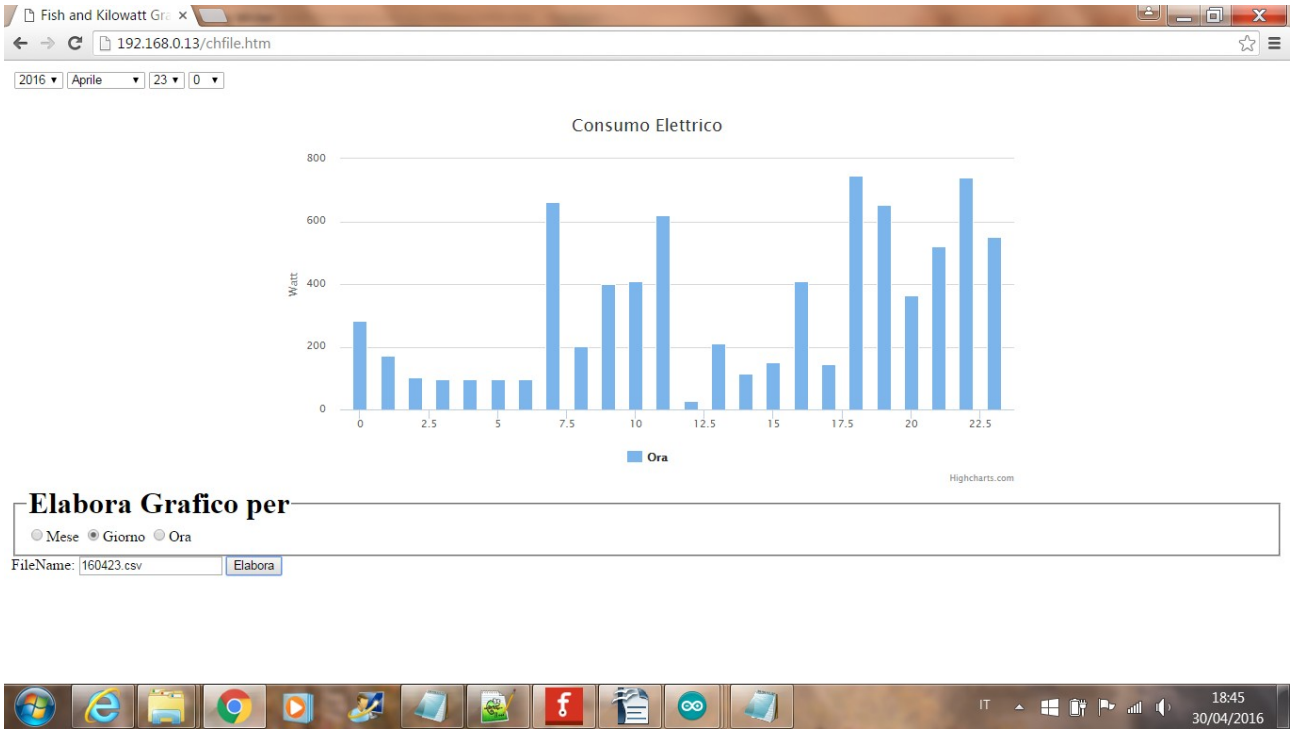
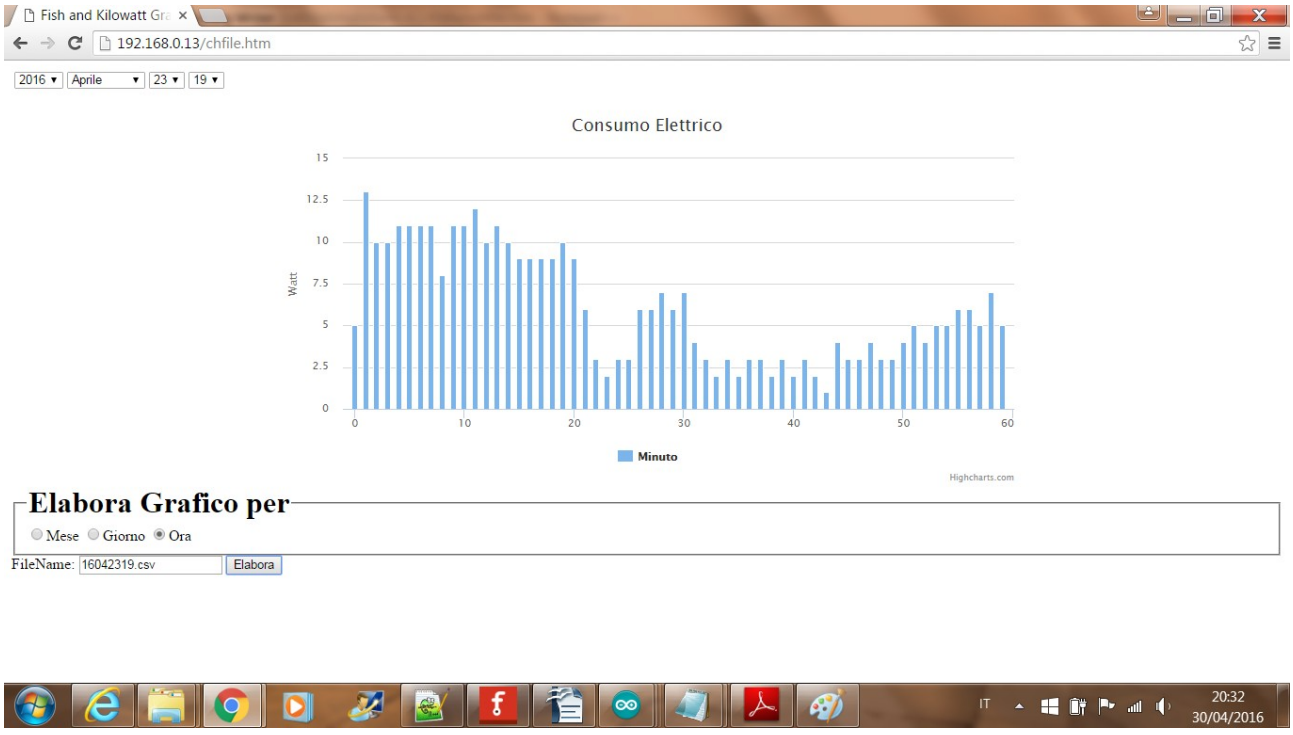


Grafico Orario (riferito ai consumi dalle 19:00 alle 19:59 del 23 aprile 2016)



logfile.htm – da questa pagina è possibile scaricare i log file salvati su scheda microsd.
L'interfaccia è simile a quella già illustrata per i grafici.

Ad esempio, con i parametri qui indicati:

possiamo scaricare il file CSV giornaliero del 24 aprile 2016

Altre considerazioni sul software

Dovendo ottimizzare al massimo l'occupazione del software, ho tolto quasi tutte le segnalazioni di debug su seriale.
Prima di caricare lo sketch, consiglio quindi di verificare la corretta configurazione dei singoli elementi:

1. rete (wifi o standalone)
   utilizzando lo sketch di esempio FishinoWiFiWebServer

2. scheda microsd
   utilizzando gli esempi della libreria SD

3. orologio RTC
   utilizzando gli esempi della libreria RTClib. Ricordarsi di impostare l'ora con l'esempio ds13078