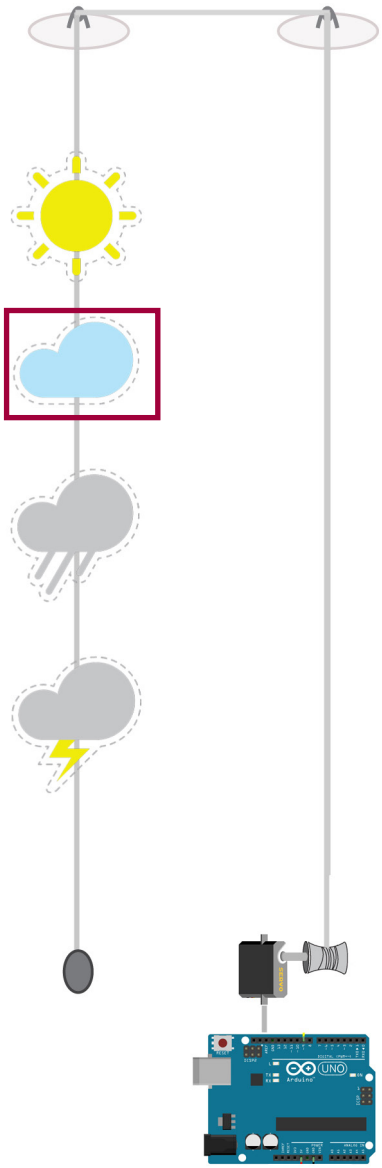
Arduino
Tutorial #11 - Sistema previsioni meteo

materiale utilizzato – Arduino ethernet o Arduino UNO con ethernet shield o Arduino UNO con WiFi shield - servo motore (o in alternativa 5 led) - ventosa - forbici - filo.

Per vedere tutti i tutorial: www.marcopucci.it/arduino/

N.B: Clicca qui per scaricare il file .zip con tutti gli sketch utilizzati per realizzare il sistema previsioni meteo.



In questo tutorial realizziamo un sistema per la visualizzazione delle condizioni meteo nella nostra città.
Per realizzare questo progetto è necessario leggere il tutorial precedente (clicca qui).
Il sistema che stiamo per costruire si collega al sito meteo www.wunderground.com **ogni ora** e scarica sulla scheda Arduino i dati riguardanti le previsioni meteo della nostra città. Questi **dati vengono analizzati** e, a seconda della situazione meteo, un motorino **muove una corda** a cui sono attaccate delle icone di cartone relative alle previsioni.
Ad esempio se nel pomeriggio è prevista pioggia, il sistema sposterà l'icona del temporale al centro di un rettangolo (un adesivo attaccato alla finestra) che rappresenta la previsione delle prossime ore. Qui sotto lo schema grafico del sistema meteo.
La struttura può essere installata su qualsiasi superficie piana come finestre o pareti.

Se non si possiede il motorino servo è possibile costruire la postazione meteo con cinque led (più avanti le istruzioni per costruirla).

Oltre alla gestione dei dati catturati dalla rete vediamo come gestire le unità di tempo (ogni ora scarichiamo i dati) e la gestione di un servo motore continuo (al contrario del servo motore visto nel tutorial n. 8, questi motorini posso ruotare senza bloccarsi a 180°) che sposterà il filo a cui sono attaccate le icone delle previsioni meteo.
Dividiamo il tutorial in tre parti: nella prima parte realizziamo un piccolo sketch in grado di creare un orologio, nel secondo muoviamo un servo motore continuo e nel terzo riprendiamo il tutorial precedente analizzando i dati scaricati da wunderground.com.
Alla fine uniamo i tre sketch e assembliamo il sistema previsioni meteo.

Orologio (Aprite lo sketch _3_orologio_if)

Per realizzare un orologio con Arduino utilizzeremo tre variabili h (ore), m (minuti) e s (secondi).
Scrivete il codice qui sotto nel programma di Arduino, verificate che sia corretto, uplodate e aprite il Serial Monitor. Ogni minuto apparirà la scritta "è passato un minuto!".

```
int m = 0;
int s = 0;
int h = 0;

void setup() {

Serial.begin(9600);

}

void loop() {

if (s == 59){
    m++;
    s = 0;
    }


    else
   {
    s++;
   }



if (m == 60){
    h+=1;
    m = 0;
  }

 delay(1000);
 Serial.println(s);
 Serial.println(m);

 if ((h==0)&&(m == 1)&&(s==0)) {
    Serial.print("è passato un minu-
to!");
    s=0;
    m=0;
}
}
```

Dichiaro le tre variabili s, m e h

La funzione loop ripete ogni secondo l'intero codice

se i secondi sono 59 allora incremento la variabile minuti (m++ vuol dire che il valore di "m" è uguale al valore precedente +1)

se "m" non è ancora uguale a 1 (non sono ancora passati 59 secondi) allora incrementa la variabile "s" di una unità.

Se "m" è uguale a 60 (è passata un'ora) allora la variabile "h" diventa il valore precedente di "h" + 1.

Questa parte di codice è il nostro orologio.

In questo punto inseriamo un controllo per fare compiere un'operazione dopo che è trascorso un minuto. Se h=0 e contemporaneamente (&&) m è uguale a 1 e i secondi sono 0 (se è passato un minuto) allora scrivi sul Serial Monitor "è passato un minuto!"
N.B: se avete copiato questo testo all'interno dello sketch e vi segnala un errore controllate le " " (cancellatele e scrivetele di nuovo).

Alla fine di questo tutorial realizzeremo il sistema che controlla i dati meteo ogni ora.
In questo caso la IF sarà: if ((h==1)&&(m == 0)&&(s==0)), e al posto di inviare sul Serial Monitor una scritta, inseriremo il codice che preleva i dati dal sito www.wunderground.com

Meteo (Aprite lo sketch _4_orologio_if_meteo)

Preleviamo i dati dal sito www.wunderground.com.
Per continuare aprite lo sketch realizzato nel tutorial precedente relativo al download dei dati meteo della nostra città (clicca qui per leggere il tutorial precedente).

Iniziamo a unire i due sketch: ogni volta che passa un minuto Arduino deve collegarsi al sito meteo e scaricare le previsioni per il giorno dopo.
Dobbiamo unire lo sketch orologio con quello del meteo.

```
int h = 0;
int m = 0;
int s = 0;

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

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED
};
//const char server[] = "api.wunderground.com";
IPAddress server(2,23,104,120); // IP 2,23,104,120
di api.wunderground.com
IPAddress ip(10,0,3,127);
EthernetClient client;
TextFinder finder( client );
String responseString;
boolean startCapture;
String readString = String(100);
char buffer_domani[31];

void setup() {
Serial.begin(9600);
}

void loop() {
  Serial.flush();
 //Serial.print("Orologio");

  if (s == 59){
    m++;
    s = 0;
  }
  else
  {
    s++;
  }

 if (m == 60){
    h+=1;
    m = 0;
  }

  if (h == 24){
    h=0;
  }

  delay(1000);
   Serial.print(s);
   Serial.print("-");
   Serial.println(m);
   if (m==1){

  if (Ethernet.begin(mac) == 0)
  {
    Serial.println("Failed to configure Ethernet
using DHCP");
    Ethernet.begin(mac,ip);
  }
```

Importiamo all'interno dello sketch dell'orologio tutte le variabili e le librerie utilizzate nello sketch del meteo

L'orologio inizia a contare

Quando è passato un minuto copiamo tutto lo sketch relativo alla connessione sul sito wunderground.com e la ricerca delle previsioni meteo.

```
delay(500);
  Serial.print("My IP Address is: ");
  Serial.println(Ethernet.localIP());
  Serial.println("Connecting...");

  if (client.connect(server, 80))
  {
    Serial.println("Connected!");
     String richiesta = "GET http://api.wun-
derground.com/api/f8ac4a22f841d562/conditions/
forecast/q/zmw:00000.1.16080.xml HTTP/1.0";

    String chiusura_connessione = "Connection:
close";

    client.println(richiesta);
    client.println(chiusura_connessione);
    client.println();

    responseString = "";
    startCapture = false;
  }
  else
  {
   Serial.println("Connection failed!");
  }
  if (client.available())
  {
    char c = client.read();
    if(c == '{')
      startCapture=true;

    if(startCapture)
      responseString += c;
  }

    if (client.connected())
  {

   Serial.print("tempo domani:  ");
   if ( finder.find("<period>2</period>") && finder.
find ("icon")){
   finder.getString(">","<",buffer_
domani,sizeof(buffer_domani));
   char *tempo_domani =buffer_domani;
   Serial.println(tempo_domani);


  }


    client.stop();
    client.flush();


   m=0;
   }

   }

   }
```

Stampiamo sul SerialMonitor le previsioni meteo del giorno dopo

Azzero la variabile minuti. In questo modo l'orologio riparte da zero e, passato un minuto, si ricollega al sito meteo

Meteo con funzioni - (Aprite lo sketch _5_orologio_if_meteo_funzioni)

Le funzioni sono una parte fondamentale in un linguaggio di programmazione. Aiutano a tenere più pulito il codice mettendo un po' di ordine al suo interno.
Ad esempio utilizzando il codice precedente possiamo effettuare le seguenti modifiche:

```
void loop() {
Serial.flush();

  if (s == 59){
    m++;
    s = 0;
  }
  else
  {
    s++;
  }

 if (m == 60){
    h+=1;
    m = 0;
  }

  if (h == 24){
    h=0;
  }
   delay(1000);
   Serial.print(s);
   Serial.print("-");
   Serial.println(m);

   if (m==1){
   connessione();
   lettura_meteo();

   m=0;
   }
   }


  void connessione(){
    if (Ethernet.begin(mac) == 0)
  {
    Serial.println("Failed to configure Ethernet using
DHCP");
    Ethernet.begin(mac,ip);
  }
    delay(500);
  Serial.print("My IP Address is: ");
  Serial.println(Ethernet.localIP());
  Serial.println("Connecting...");
  if (client.connect(server, 80))
  {
    Serial.println("Connected!");
    String richiesta = "GET http://api.wunderground.
com/api/f8ac4a22f841d562/conditions/forecast/q/
zmw:00000.1.16080.xml HTTP/1.0";
    String chiusura_connessione = "Connection: close";
    client.println(richiesta);
    client.println(chiusura_connessione);
    client.println();
    responseString = "";
    startCapture = false;
  }
  else
  {
   Serial.println("Connection failed!");
  }
  }
```

La prima parte di codice, quella relativa alle variabili, è identica a quella precedente.

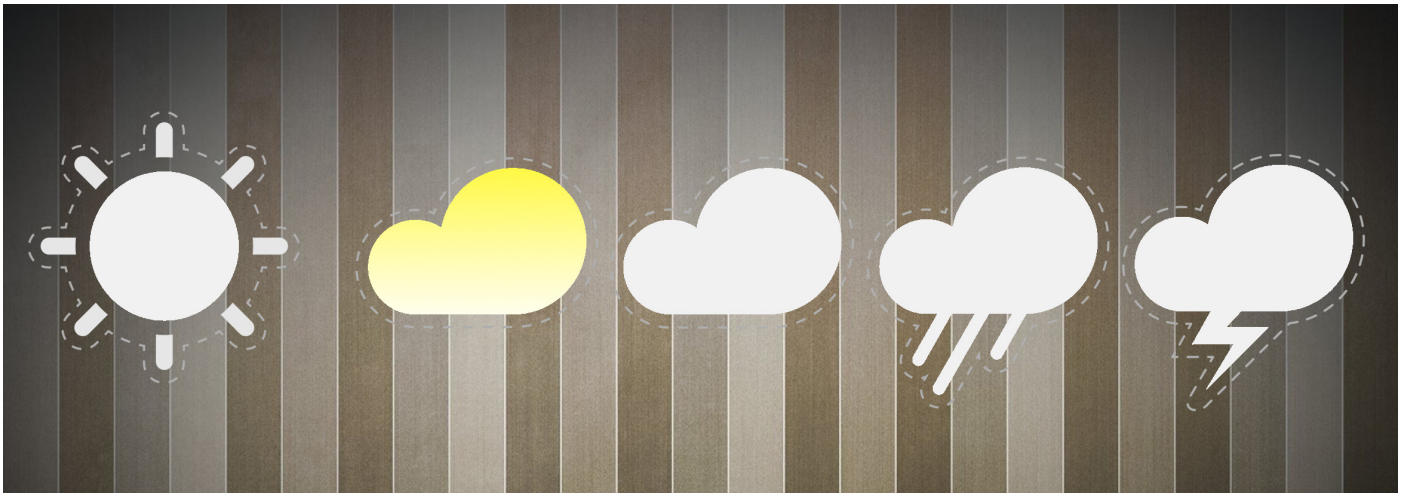
Quando viene superato un minuto vengono richiamate due funzioni: connessione() e lettura_meteo().
Queste due funzioni sono presenti alla fine del codice. Al loro interno c'è lo stesso codice dello sketch precedente ma è stato diviso per creare più ordine nella programmazione. Infatti ora quando la IF dei minuti supera il valore 1 richiama prima la funzione connessione() che si collega al sito meteo e subito dopo la funzione lettura_meteo() che scarica e visualizza i dati delle previsioni meteo.
Se provate a caricarla su Arduino il risultato è identico a quello di prima.

```
void lettura_meteo(){
 if (client.available())
  {
    char c = client.read();
    if(c == '{')
      startCapture=true;

    if(startCapture)
      responseString += c;
  }

    if (client.connected())
  {

    Serial.print("tempo domani:  ");
    if ( finder.find("<period>2</period>") && finder.
find ("icon")){
    finder.getString(">","<",buffer_
domani,sizeof(buffer_domani));
    char *tempo_domani =buffer_domani;
    Serial.println(tempo_domani);

    }

    client.stop();
    client.flush();

  }
  }
    responseString = "";
    startCapture = false;
  }
  else
  {
   Serial.println("Connection failed!");
  }
  }

     Serial.println(tempo_domani);


  }

    client.stop();
    client.flush();

   }

   }
```

Questa è la funziona lettuta_meteo().
Al suo interno c'è lo stesso codice utilizzato prima per scaricare e stampare i dati del sito meteo.

Sistema previsioni meteo con led - (Aprite lo sketch _6_meteo_con_led)

Siamo pronti per terminare il primo prototipo di sistema rilevamento meteo utilizzando cinque led.
Il sistema funziona così: cinque led si accendono in base alle previsioni ricevute dal sito wunderground.com.
Se è bel tempo si accende il primo led, se è nuvoloso il secondo, se piove il terzo, se c'è un temporale il quarto, se nevica il quinto.
Davanti ai cinque led è posizionata una striscia di cartone con le 5 icone meteo ritagliate e coperte con della carta semitrasparente. Quando un led si accende illumina l'icona corrispondente.
Qui sotto lo schema grafico dell'installazione. Collegate 5 led ad Arduino e inseriteli all'interno di una scatola con le cinque icone ritagliate.



Sotto le sagome, all'interno del cartone, sono posizionati i cinque led.
Clicca qui per scaricare le sagome delle icone meteo.

Per finire questo primo sistema meteo dobbiamo verificare il dato che stiamo scaricando dalla rete.
Se mandiamo in stampa l'ultimo sketch vediamo che nel Serial Monitor, terminato il minuto di countdown, appare la scritta:

Connected!
tempo domani: clear

La variabile **tempo_domani** contiene questa informazione.
Le varie opzioni che possiamo scaricare dal sito wunderground.com sono:

SOLE: clear - partlysunny - nt_clear - nt_partlysunny

NUVOLE: cloudY - mostlycloudY - nt_cloudY - nt_mostlycloudY

PIOGGIA: chancesleet - chancerain - sleet - rain - nt_chancesleet- nt_chancerain - nt_sleet - nt_rain -

TEMPORALE: chancetstorms - nt_chancetstorms

NEVE: chanceflurries - nt_chanceflurries

N.B: il nome preceduto da nt_ si riferisce alle previsione notturne. Per vedere tutte le previsioni disponibili clicca qui:
http://www.wunderground.com/weather/api/d/docs?d=resources/icon-sets

Per fare accendere il led relativo alla previsione meteo dobbiamo controllare se la variabile tempo_domani è uguale a una delle opzioni meteo.

if ((strcmp(tempo_domani,”clear”)==0)
{
...accendiamo il led sotto l’icona del sole
}

Il comando **strcmp** (string compare) verifica se la variabile tempo_domani è uguale alla parola messa tra le due virgolette (in questo caso clear).

L’icona del Sole deve essere illuminata anche se il dato scaricato è uguale alle parole “partlysunny”, “nt_clear” e “nt_partlysunny”.
In questo caso il controllo IF deve essere scritto in questo modo:

if ((strcmp(tempo_domani,”clear”)==0) || (strcmp(tempo_domani,”nt_clear”)==0) || (strcmp(tempo_domani,”partlysunny”)==0) || (strcmp(tempo_domani,”nt_partlysunny”)==0))
{

}

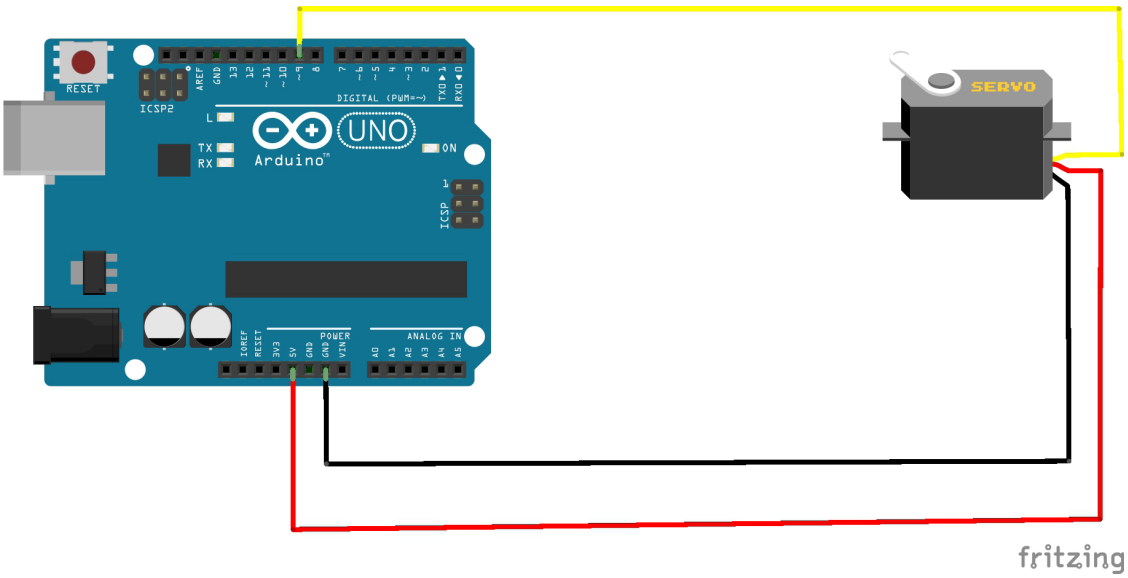
Il simbolo || in questo codice significa **OR**. All’interno della If vuol dire se il dato scaricato è uguale a “clear” **OPPURE** è uguale a “nt_clear” **OPPURE** è uguale a “partlysunny” **OPPURE** è uguale a “nt_partlysunny” **ALLORA** accendi il led sotto l’icona del sole.
L’operazione logica OR rende la condizione della IF vera se almeno uno dei vari controlli risulta essere vero.

Qui sotto la parte di codice dei controlli IF sul dato scaricato.

```
if((strcmp(tempo_domani,”clear”)==0)|| (strcmp(tempo_domani,”nt_clear”)==0)|| (strcmp(tempo_
domani,”partlysunny”)==0)|| (strcmp(tempo_domani,”nt_partlysunny”)==0)){
     digitalWrite(led_sole,1);
     digitalWrite(led_nuvoloso,0);
     digitalWrite(led_pioggia,0);
     digitalWrite(led_temporale,0);
     digitalWrite(led_neve,0);
     delay(2000);
     }

     //se il tempo è  nuvoloso
     if((strcmp(tempo_domani,”cloudy”)==0)|| (strcmp(tempo_domani,”nt_cloudy”)==0)|| (strcmp(tempo_
domani,”partlycloudy”)==0)|| (strcmp(tempo_    domani,”nt_partlycloudy”)==0)||(strcmp(tempo_
domani,”mostlycloudy”)==0)|| (strcmp(tempo_domani,”nt_mostlycloudy”)==0)){
     digitalWrite(led_sole,0);
     digitalWrite(led_nuvoloso,1);
     digitalWrite(led_pioggia,0);
     digitalWrite(led_temporale,0);
     digitalWrite(led_neve,0);
     }

     //se piove
     if((strcmp(tempo_domani,”chancesleet”)==0)|| (strcmp(tempo_domani,”nt_chancesleet”)==0)||
(strcmp(tempo_domani,”chancerain”)==0)|| (strcmp(tempo_domani,”nt_chancerain”)==0)){
     digitalWrite(led_sole,0);
     digitalWrite(led_nuvoloso,0);
     digitalWrite(led_pioggia,1);
     digitalWrite(led_temporale,0);
     digitalWrite(led_neve,0);
    }
    ........
```

Sistema previsioni meteo con servo motore continuo

Abbiamo già visto nel tutorial “Costruiamo un Danbo interattivo” (clicca qui per leggerlo) L’utilizzo dei servo motori a 180°. In questo tutorial studiamo invece i servo motori continui in grado di proseguire la loro corsa senza doversi fermare a 180°. La differenza tra i due sta proprio in questa piccola particolarità: mentre quelli a 180° sono dei motorini precisi, che riescono a tornare al grado indicato, quelli continui non possono fermarsi a una determinata posizione ma si muovono avanti o indietro in modo continuo.
Questi ultimi vengono ad esempio utilizzati in robotica per le ruote dei robot in modo da poter gestire la direzione della ruota e la loro velocità.
Qui sotto lo schema elettronico per collegarlo alla scheda Arduino.



I motorini 360° si collegano esattamente come quelli bloccati a 180°. Oltre all’alimentazione Vcc (5V) e la massa (GND) colleghiamo il cavo giallo al pin 9 di Arduino (il pin9 gestirà i comandi del motorino).

```
#include <Servo.h>                Importiamo la libreria per la gestione dei servo
Servo myservo;                    Dichiariamo una variabile per comandare il motorino

void setup()
{
 Serial.begin( 9600 );
 myservo.attach(9);               Il motorino sarà collegato al pin digitale 9
}

void loop()
{
 myservo.write(0);                Il motorino va indietro per 1 secondo
 delay(1000);
 myservo.write(90);               Il motorino si ferma per 1 secondo
 delay(2000);
 myservo.write(180);              Il motorino va avanti per un secondo
 delay(1000);
 myservo.write(90);               Il motorino si ferma per un secondo
 delay(2000);
}
```

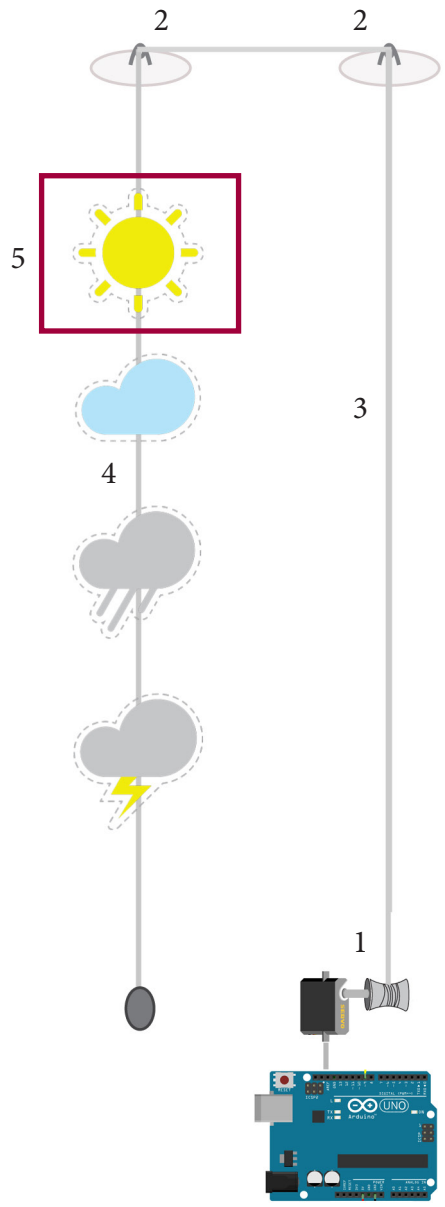
Servo motore continuo con funzioni - Aprite lo sketch _7_motorini_funzioni

Vediamo anche in questo caso l'utilizzo delle funzioni.

```
#include <Servo.h>

Servo myservo;

void setup()
{
   Serial.begin( 9600 );
   myservo.attach(9);
}

void loop()
{
  avanti();
  fermo();
  dietro();
  fermo();
}

void avanti(){
myservo.write(0);
delay(1000);
}

void fermo(){
myservo.write(90);
delay(1000);
}

void dietro(){
myservo.write(180);
delay(1000);
}
```

All'interno della funzione loop() non appare più il comando che fa muovere il motorino avanti o indietro. Al suo posto ci sono tre funzioni (avanti, dietro e fermo).
Queste funzioni quando vengono lette portano direttamente alla fine del codice dove sono dichiarate.
La loro comodità sta nel fatto che quando voglio far andare avanti il motorino non devo riscrivere tutto il codice ma devo semplicemente richiamare la funzione avanti();

Una funziona deve essere scritta in questo modo: void + nome della funzione + (){ tutto il codice che appare tra le due parentesi graffe fa parte della funzione }



Costruiamo il sistema meteo con il servo motore.

1 - Incolliamo un rocchetto di filo al nostro motorino.

2 - Attacchiamo due ventose a un vetro oppure due chiodi in un muro.

3 - Facciamo scorrere il filo dal rocchetto alle ventose o ai chiodi.

4 - Stampiamo, ritagliamo e incolliamo al filo le icone del meteo (clicca qui per scaricare il file)

- Le icone devono essere incollate a una distanza di circa 4 cm. Prima di incollarle facciamo questa prova:

5 - Attacchiamo sul vetro o sul muro un adesivo che indica il punto di lettura delle nostre previsioni meteo (nello schema a sinistra il rettangolo viola rappresenta il punto di lettura).

Posizioniamo la prima icona del sole al centro dell'adesivo di lettura.
Carichiamo lo sketch di Arduino con il seguente codice all'interno del loop()

```
avanti();
avanti();
fermo();
```

Ogni movimento di icona è formato da due avanti()

Ogni volta che il filo si sposta e si ferma scolleghiamo Arduino, attacchiamo l'icona delle nuvole al centro dell'adesivo. Colleghiamo nuovamente Arduino, il filo si sposta di nuovo e attacchiamo la terza icona...e così via.
In questo modo abbiamo tarato le distanze delle varie icone.

Lo sketch finale farà muovere avanti o indietro le icone in base alle previsioni scaricate. Per gestire il movimento del filo dobbiamo sapere esattamente la posizione dell'icona al centro dell'adesivo.
Es: se la previsione è di temporale e quella precedente è di nuvoloso il sistema farà salire il filo di due unità.
Se dopo il temporale è previsto bel tempo allora il sistema si muoverà di 3 unità in basso.

Per conoscere la posizione dell'icona al centro dell'adesivo utilizziamo una variabile di appoggio che chiamiamo **statoMeteo**.
statoMeteo=0 - icona sole al centro dell'adesivo
statoMeteo=1 - icona nuvoloso
statoMeteo=2 - icona pioggia
statoMeteo=3 - icona temporale
statoMeteo=4 . icona neve

Aprite lo sketch _8_meteo_con_motorini

Qui sotto una parte dello sketch.
In questo caso l'icona del sole è al centro dell'adesivo e il meteo scaricato prevede altro bel tempo.

```
if ((strcmp(tempo_domani,"clear")==0)||
(strcmp(tempo_domani,"nt_clear")==0)||
(strcmp(tempo_domani,"partlysunny")==0)||
(strcmp(tempo_domani,"nt_partlysunny")==0)){
```

Se le previsioni di domani sono di bel tempo allora controllo la posizione delle icone per poter muovere la nuova previsione.

```
 if (statoMeteo==0){
         fermo();
         }

         if (statoMeteo==1){
         dietro();
         dietro();
          fermo();
         }
         if (statoMeteo==2){
         dietro();
         dietro();
         dietro();
         dietro();
          fermo();
         }
         if (statoMeteo==3){
         dietro();
         dietro();
         dietro();
         dietro();
         dietro();
         dietro();
          fermo();
         }
        if (statoMeteo==4){
         dietro();
         dietro();
         dietro();
         dietro();
         dietro();
         dietro();
         dietro();
         dietro();
         dietro();
          fermo();
         }
         statoMeteo=0;
     }
```

Se statoMeteo==0 vuol dire che al centro dell'adesivo c'è già l'icona del sole e quindi non si deve muovere niente

Se ==1 allora l'icona al centro è quella con le nuvole. Il sistema deve fare due passi indietro

Se ==3 l'icona al centro dell'adesivo è temporale. Il sistema deve scendere di tre posizioni per far tornare l'icona del sole al centro.

Questa installazione è servita per ripassare lo sketch sul download di dati dalla rete (tutorial 10), la gestione dei motorini continui e l'utilizzo delle funzioni.

Alcuni problemi potrebbero essere causati dalla scarsa precisione dei motorini 360° che non tornano mai allo stesso punto di partenza (a differenza di quelli da 180°).
In questi casi è bene tarare leggermente il tempo di discesa e salita del cavo, sistemare bene il rocchetto del filo evitando movimenti di filo che differiscono quando viene rilasciato o tirato.

Quando fate partire il sistema posizionate manualmente l'icona del sole al centro dell'adesivo.

Qui sotto alcune immagini del sistema meteo.
Sul sito http://www.marcopucci.it/tutorial-arduino-11-previsioni-meteo/ è presente un piccolo video dell'installazione.
A questo link potete scaricare tutte gli sketch utilizzati in questo tutorial.