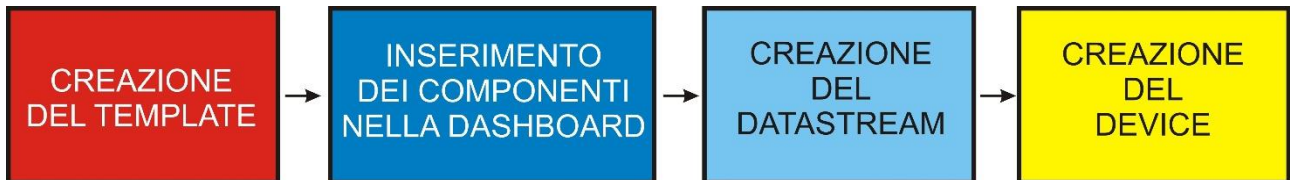


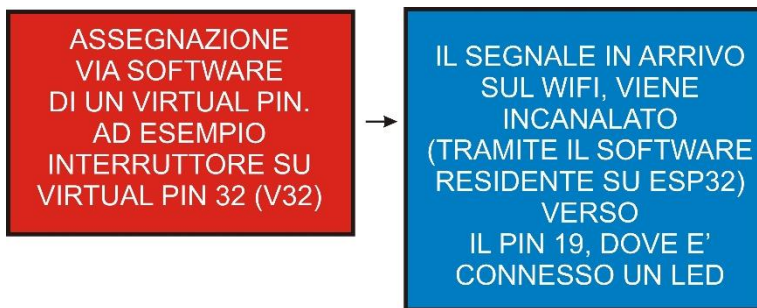
La piattaforma BLYNK

Blynk è una piattaforma che ci consente di gestire dispositivi a distanza (per esempio eccitare via web un relè remoto). Su "Laborobotica" Vol.B c'è una descrizione della sua configurazione, ma visto che è stato aggiornato il sito proprietario questa è ormai obsoleta. L'upgrade consente oggi di avere un'interfaccia utente su PC, oltre che quella su smartphone. La nuova configurazione può sembrare criptica, ma con le dovute indicazioni vedremo di rendere il tutto il più possibile semplice e lineare. Per quanto riguarda il PC, il flusso di lavoro dovrebbe essere impostato in questo modo:



Il template è come se fosse il contenitore di tutta la struttura. Viene visualizzato nella sezione appunto Templates e anche sullo schermo dello smartphone, pronto per essere selezionato.

La Web Dashboard può essere descritta come l'interfaccia utente vera e propria con dati e immagini relative al sistema.



Il Datasream è tutto quello che riguarda i collegamenti. È possibile assegnare ad un elemento, ad esempio un interruttore, un virtual pin, che non ha niente a che fare con i pin fisici dell'hardware. In sede di programmazione (sketch) sarà possibile via software assegnare questi pin all'ESP32 o ad Arduino, rendendo i collegamenti più semplici e il sistema più versatile.

La creazione del Device è la parte finale nella quale Blynk assegna le chiavi di riconoscimento consentendo all'Hardware (ESP32) la comunicazione con il software (applicazione in funzione su PC o su smartphone).

```
Click to copy Code  
#define BLYNK_TEMPLATE_ID "TMPL4fg0LAq-B"  
#define BLYNK_TEMPLATE_NAME "CONTROL"  
#define BLYNK_AUTH_TOKEN "HtPLoBb--Hqorssmqd4xeLvY0ykrWI"  
Template ID, Template Name, and AuthToken should be declared at the very top of the firmware code.  
Documentation Copy to clipboard
```

Alla fine dovremo configurare anche un'ulteriore interfaccia utente sullo smartphone.

Il progetto

Vogliamo costruire un'interfaccia utente con un interruttore che ci permetta via web di eccitare un relè a distanza.

Per cominciare bisogna portarsi su <https://blynk.cloud/dashboard/login>

Registriamoci fornendo username e password.

Il Template.

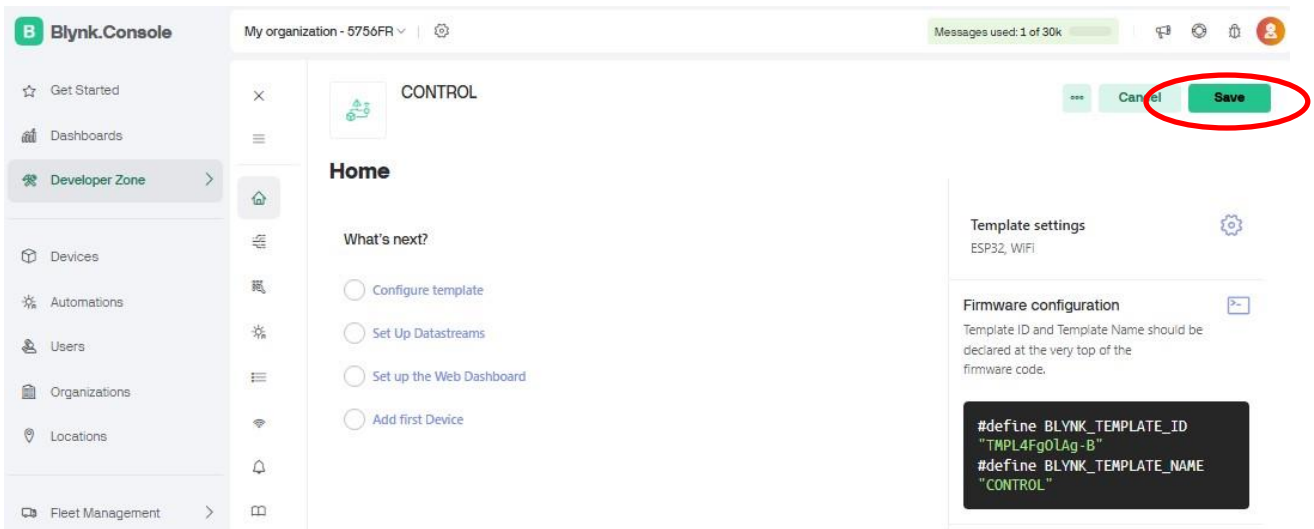
Entrare in [Developer Zone](#) > [My templates](#) click su [+ New Template](#)

The screenshot shows the Blynk Console interface. On the left sidebar, 'Developer Zone' is highlighted with a green circle. In the main menu, 'My Templates' is also highlighted with a green circle. The main content area displays a message: 'Start by creating your first template' followed by a brief explanation: 'Template is a digital model of a physical object. It is used in Blynk platform as a template to be assigned to devices.' Below this text, a green button labeled '+ New Template' is circled in red.

Dare il nome al template e specificare il tipo di hardware usato click su [Done](#)

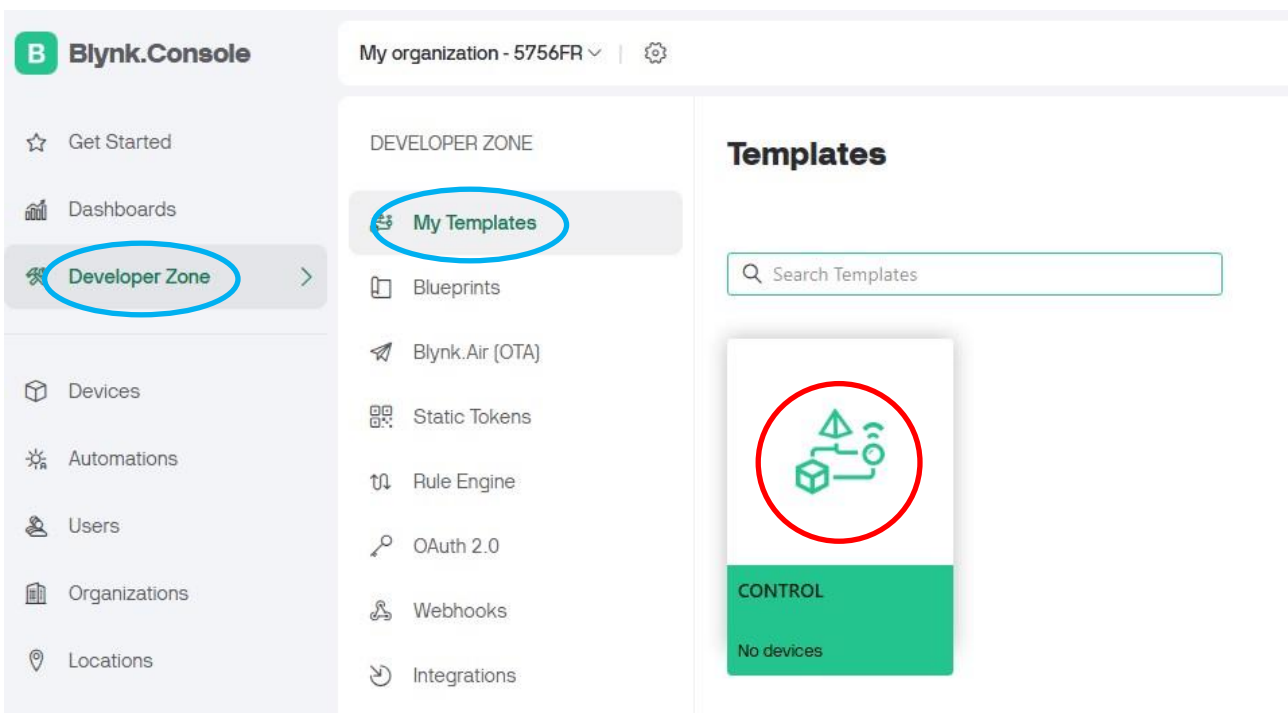
The screenshot shows the 'Create New Template' form. The 'NAME' field contains 'CONTROL' and is circled in blue. The 'HARDWARE' dropdown menu is set to 'ESP32' and is also circled in blue. The 'CONNECTION TYPE' dropdown menu is set to 'WiFi'. The 'DESCRIPTION' field is empty. At the bottom right, there are two buttons: 'Cancel' and 'Done', with 'Done' circled in red.

Ci si troverà nella console di programmazione. Sulla destra parte dei codici da inserire nello sketch di programmazione che renderanno univoco il template. [Click su Save](#)



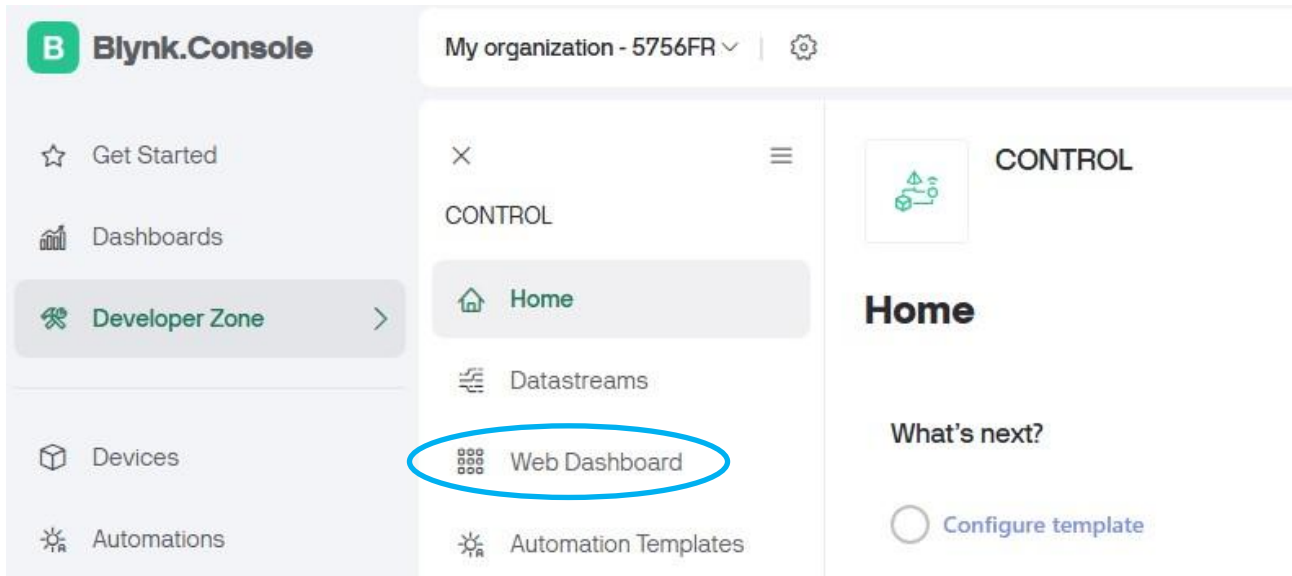
Se andremo in [Developer Zone > My Templates](#) vedremo il template CONTROL senza componenti (No devices)

[Click sul template](#)

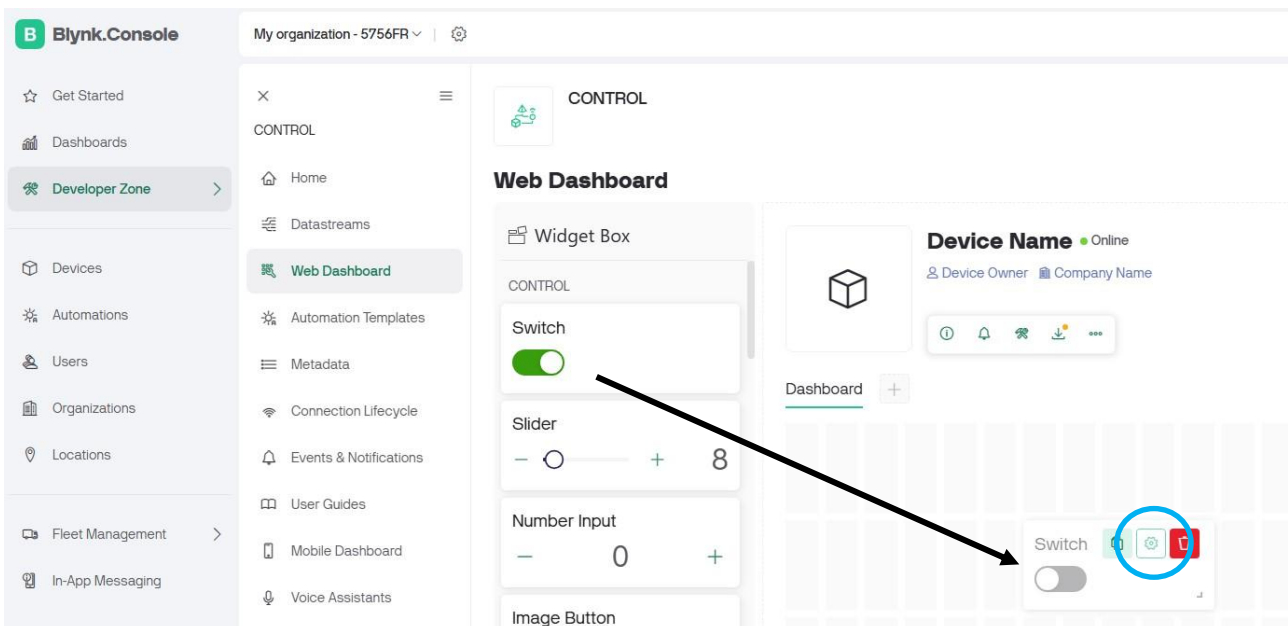


La Web Dashboard

Ora dovremo inserire i componenti nella Web Dashboard: click su [Web Dashboard](#) e poi su Edit in alto a destra.



Potremo ora inserire i componenti: drag & drop con lo switch, posizionamolo nello stage di lavoro Poi click sull'ingranaggio per la configurazione.



Il Datastream.

Ora possiamo creare il Datastream. Diamo il nome al componente e poi scegliamo Virtual Pin e assegnamo V32.

Switch Settings

TITLE (OPTIONAL)

Switch

Datastream

You have no datastreams to select

+ Create Datastream

Virtual Pin

Enum

Digital Pin

Create, Save poi ancora save sulla Web Dashboard. Abbiamo creato un'interfaccia utente con un interruttore

Switch Settings

TITLE (OPTIONAL)

CONTROL

Datastream

Virtual Pin Datastream

General Expose to Automations

NAME	ALIAS	
<input type="text" value="CONTROL"/>	<input type="text" value="CONTROL"/>	
PIN	DATA TYPE	
<input type="text" value="V32"/>	<input type="text" value="Integer"/>	
UNITS		
<input type="text" value="None"/>		
MIN	MAX	DEFAULT VALUE
<input type="text" value="0"/>	<input type="text" value="1"/>	<input type="text" value="0"/>
<input type="checkbox"/> Enable history data		
<input type="checkbox"/> ADVANCED SETTINGS		
<input type="button" value="Cancel"/>		<input type="button" value="Create"/>

CONTROL



Cancel

Save

Il Device.

Dobbiamo ora creare un nuovo Device: Devices > **+ New Device**


The screenshot shows the Blynk.Console dashboard. On the left is a sidebar with navigation options: Get Started, Dashboards, Developer Zone, **Devices** (highlighted), Automations, Users, Organizations, Locations, Fleet Management, and In-App Messaging. The main content area displays the heading "All of your devices will be here." followed by the text "You can activate new devices by using your Blynk.Console app for IOS or Android". Below this are two buttons: "Download for IOS" and "Download for Android". At the bottom center, a green button with a plus sign and the text "+ New Device" is circled in red.

Click su **From Template**

New Device

Choose a way to create new device

Four cards are displayed in a row, each with a title and an icon: "Create New" with a plus sign, "From template" with a red circle around an icon of a cube and a signal tower, "Scan QR code" with a QR code icon, and "Manual entry" with a keyboard icon.

 Point on the cards to see instructions

Cancel

New Device

Create new device by filling in the form below

TEMPLATE

CONTROL

DEVICE NAME

CONTROL 7 / 50

Cancel

Create

Diamo il nome e poi **Create**

Attenzione che in alto a destra compariranno le credenziali da inserire nello sketch per avere un collegamento univoco con il nostro hardware.

The screenshot shows the Blynk Console interface. On the left is a navigation menu with options like 'Get Started', 'Dashboards', 'Developer Zone', 'Devices', 'Automations', 'Users', 'Organizations', and 'Locations'. The main area displays a device named 'CONTROL' which is 'Offline'. Below the device name are various control icons and a toggle switch. A notification window titled 'New Device Created!' is open on the right, showing the following code:

```
#define BLYNK_TEMPLATE_ID "TMPL4Fg0LAg-B"  
#define BLYNK_TEMPLATE_NAME "CONTROL"  
#define BLYNK_AUTH_TOKEN "MîPLoBb--MQorssmwqd4xeLvy0ykrHI"
```

Below the code, it says: 'Template ID, Template Name, and AuthToken should be declared at the very top of the firmware code.' There are buttons for 'Documentation' and 'Copy to clipboard'.

This is a close-up of the notification window. It features a 'Click to copy Code' button at the top left. The code is displayed in a dark box with green text:

```
#define BLYNK_TEMPLATE_ID "TMPL4Fg0LAg-B"  
#define BLYNK_TEMPLATE_NAME "CONTROL"  
#define BLYNK_AUTH_TOKEN "MîPLoBb--MQorssmwqd4xeLvy0ykrHI"
```

Below the code, the text reads: 'Template ID, Template Name, and AuthToken should be declared at the very top of the firmware code.' At the bottom, there are buttons for 'Documentation' and 'Copy to clipboard'.

Passiamo con il mouse sul campo delle informazioni e clicchiamo su Click to copy Code. Anremo a incollare il tutto nello sketch per l'ESP32. Per sicurezza incolliamolo anche in un file di testo e teniamolo per ogni evenienza.

Lo sketch.

Ecco lo sketch da inserire sull'ESP32. Ricordiamoci di premere il pulsante BOOT sulla scheda quando compare la scritta **Connecting....**

```
#define BLYNK_PRINT Serial //la seriale di BLYNK per il passaggio dei dati

#define BLYNK_TEMPLATE_ID "TMPL4Y0SUsZ0x"
#define BLYNK_TEMPLATE_NAME "CONTROL"
#define BLYNK_AUTH_TOKEN "IlruihS0PoQ7whTImdTUvzJCh6lZj0L"
```

Il codice generato dalla creazione del device che ci permette il collegamento univoco tra l'ESP32 e la dashboard di comando.

```
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
```

Le librerie per il corretto funzionamento di BLYNK.

```
char ssid[] = "Vodafone-*****";
char pass[] = "*****";
```

La rete WiFi e la password.

```
int rel=32; //il relè è connesso al pin 32 dell'ESP32
```

```
BLYNK_WRITE(V32)
{
  int value = param.asInt();
  if (value == 1)
  {
    digitalWrite(rel, HIGH);
  }
  else
  {
    digitalWrite(rel, LOW);
  }
}
```

La sub BLYNK_WRITE, con parametro V32 viene richiamata ad ogni cambio di stato dell'interruttore posizionato nella dashboard e collegato al pin virtuale V32.

È in questo punto che avviene il collegamento tra il pin V32 e il pin rel. Difatti viene ricevuto lo stato dell'interruttore, e tramite un blocco if else, viene messo a stato alto o basso il pin rel, consentendo l'attivazione/disattivazione del relè.

```
void setup()
{
  pinMode(rel,OUTPUT); //si configura in uscita il pin del relè

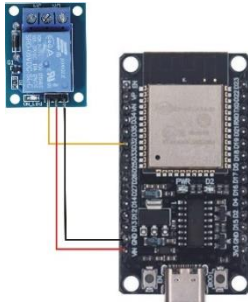
  Serial.begin(115200);

  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass); //si avvia BLYNK
}
void loop()
{
  Blynk.run();
}
```

Nel void loop è presente solo la funzione Blynk.run(): il software penserà a tutto il resto.

Lo schema elettrico.

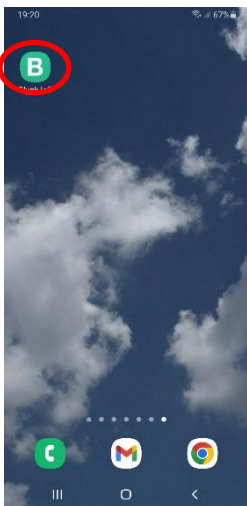
Trasferiamo lo sketch sull'ESP32. Ricordiamoci alla fine del caricamento di staccare/riattaccare l'alimentazione della scheda per effettuare un hard reset.



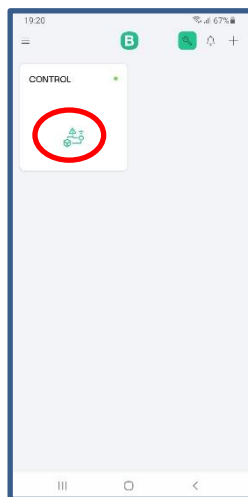
Ecco i componenti e i collegamenti. Useremo una breakout board con un relè e la scheda ESP32 WROOM.

Lo smartphone

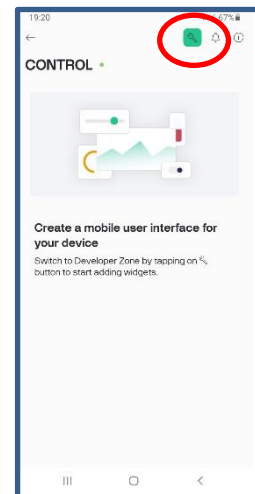
Per avere un controllo del nostro sistema anche sul cellulare dobbiamo costruire anche qui una dashboard. Innanzitutto da Play Store scarichiamo l'applicazione BLYNK e installiamola.



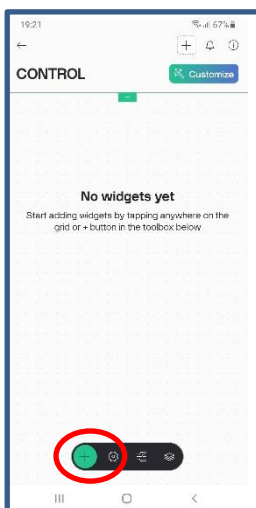
Tap sull'icona di BLYNK



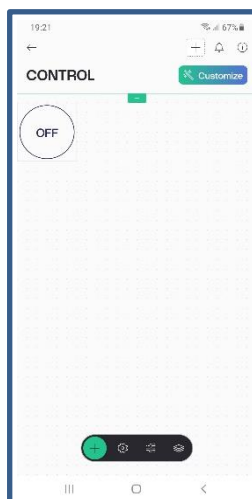
Comparirà il nostro template
Tap sull'icona



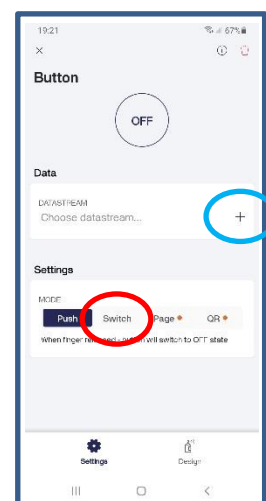
Tap sulla chiave fissa



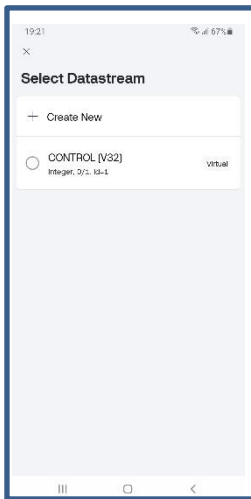
Tap sul + per aggiungere
elementi.



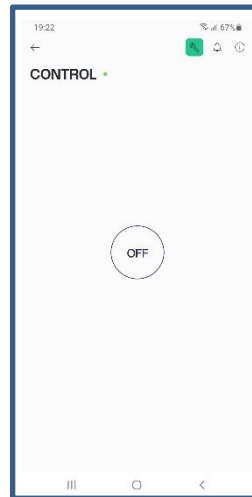
Dalla lista tap sul pulsante



Tap sul pulsante per configurarlo
[Tap sul + datastream](#)



Selezionare V32

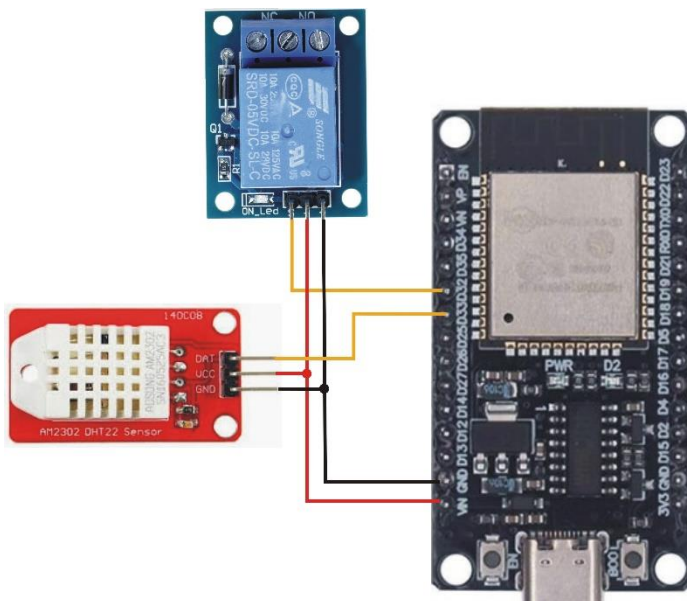


Ed ecco come si presenta l'interfaccia utente sul cellulare

Ovunque dovessimo trovarci, se presente la copertura internet, ogni qualvolta premeremo il pulsante, il relè si ecciterà.

Controllo di temperatura e umidità.

Vogliamo ora progettare un controllo di temperatura e umidità utilizzando un sensore DHT22 (ampiamente descritto su "Laborobotica" Vol.2).



Ecco i collegamenti del sistema. Il DHT22, già montato su una breakout board viene connesso con l'uscita dati al pin 33 dell'ESP32. La gestione di questo componente verrà affidata a una speciale libreria che inseriremo nello sketch.

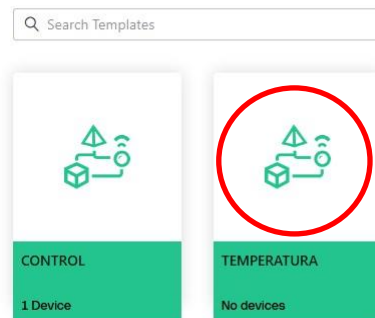
Il Template.

Entriamo in Developer Zone >My templates click su + New Template

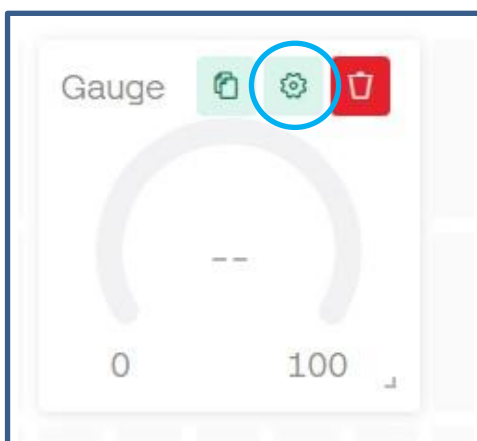
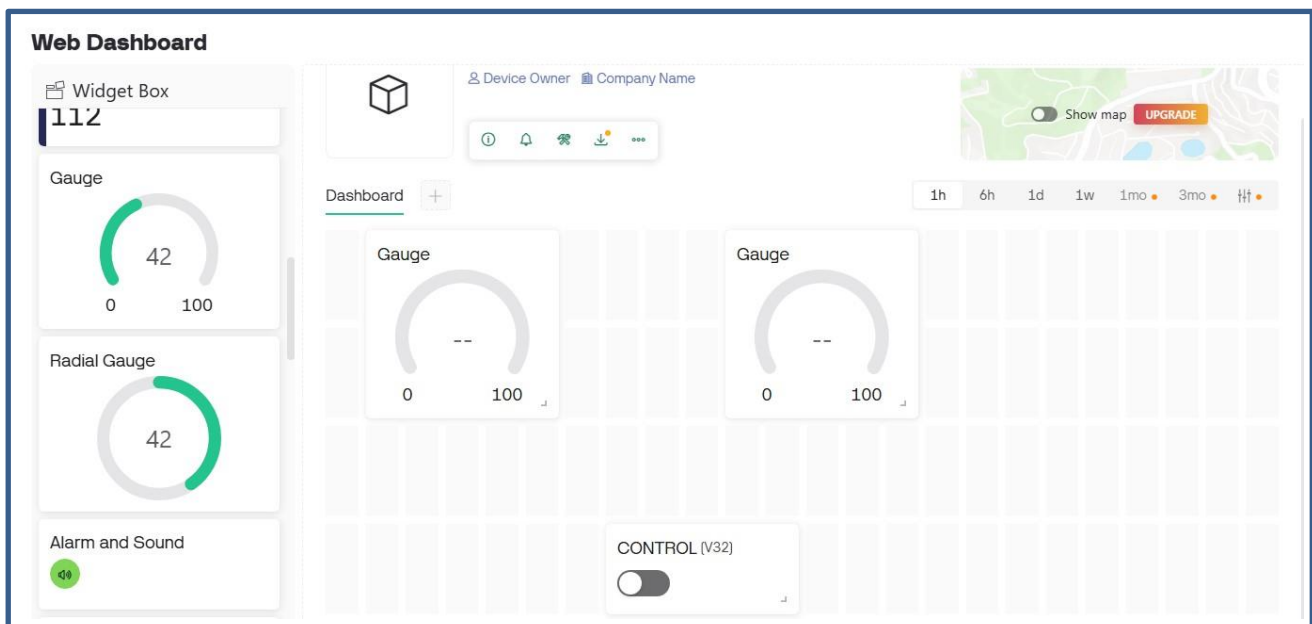
Chiamiamo il template "TEMPERATURA".

Adesso click sul template "TEMPERATURA" per creare la dashboard.

Templates



Click su Web Dashboard, poi sul tasto verde Edit in alto a destra. Trasciniamo i componenti. Utilizzeremo un interruttore e due "Gauge": uno per la temperatura e uno per l'umidità.



Passando con il mouse sopra l'immagine si potrà vedere l'ingranaggio di configurazione. [Click sull'ingranaggio.](#)

Ci troveremo nella configurazione del gauge. Assegnamo il nome TEMPERATURE poi click su [Create Datastream > Virtual Pin](#)

Gauge Settings

TITLE (OPTIONAL)
TEMPERATURE

Datastream

Choose Datastream

Gauge Settings

TITLE (OPTIONAL)
TEMPERATURE

Datastream

Virtual Pin Datastream

General Expose to Automations

NAME: TEMPERATURE ALIAS: TEMPERATURE

PIN: V33 DATA TYPE: Double

UNITS: Celsius, °C

MIN: -10 MAX: 60 DECIMALS: #.## DEFAULT VALUE: 0

Enable history data

Il nome è TEMPERATURE

Il virtual pin è il V33, il formato dei dati è double.

L'unità di misura son i gradi Celsius.

Il valore minimo è -10, il massimo è 60, il tutto con due decimali.

Dopo aver configurato Click su [Create e Save](#)

La stessa cosa faremo sul Gauge dell'umidità:

Name > Umidity Units > % (è l'umidità) Type > double Min > 0 Max > 100
Pin > V34 (Il pin è il V34, in modo da poter indirizzare in maniera univoca i dati verso i widget della dashboard)
V32 > interruttore V33 > indicatore temperatura V34 > indicatore umidità

Ora creiamo il device: Devices > +New Device > From Template > Scegliamo il template Temperature > Create
Compariranno in alto a destra i dati per lo sketch. Copiamoli.

Lo sketch.

```
#include <DHT.h>
#define DHTPIN 33
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
```

I parametri di configurazione per il corretto funzionamento del sensore DHT22.

```
#define BLYNK_PRINT Serial
```

```
#define BLYNK_TEMPLATE_ID "TMPL4dbZ6sAeS"
#define BLYNK_TEMPLATE_NAME "TEMPERATURA"
#define BLYNK_AUTH_TOKEN "yTk4hxjd2-nabeHTwRcSIBCC4xI2116e"
```

Il codice generato dalla creazione del device che ci permette il collegamento univoco tra l'ESP32 e la dashboard di comando.

```
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

char ssid[] = "Vodafone-33867814";
char pass[] = "60n73554";
int rel=32;
```

```
BlynkTimer timer;
```

```
BLYNK_WRITE(V32)
{
    int value = param.asInt();
    if (value == 1)
    {
        digitalWrite(rel, HIGH);
    } else
    {
        digitalWrite(rel, LOW);
    }
}

void misura()
{
    float t = dht.readTemperature();
    Blynk.virtualWrite(V33, t);

    float h = dht.readHumidity();
    Blynk.virtualWrite(V34, h);
}
```

La sub BLYNK_WRITE, con parametro V32 viene richiamata ad ogni cambio di stato dell'interruttore posizionato nella dashboard e collegato al pin virtuale V32.

È in questo punto che avviene il collegamento tra il pin V32 e il pin rel. Difatti viene ricevuto lo stato dell'interruttore, e tramite un blocco if else, viene messo a stato alto o basso il pin rel, consentendo l'attivazione/disattivazione del relè.

La sub misura(), che effettua le rilevazioni di temperatura e umidità viene richiamata ogni 60 secondi dalla funzione `timer.setInterval(60000L, misura);`

Con la funzione `Blynk.virtualWrite();` Scriverà la temperatura su V33 e l'umidità su V34.

```

void setup()
{
  pinMode(re1,OUTPUT);
  pinMode(DHTPIN, INPUT);
  Serial.begin(115200);

  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
  dht.begin();

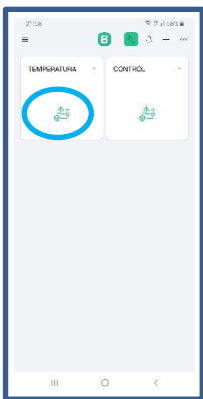
  timer.setInterval(60000L,misura);
}
void loop()
{
  Blynk.run();
  timer.run();
}

```

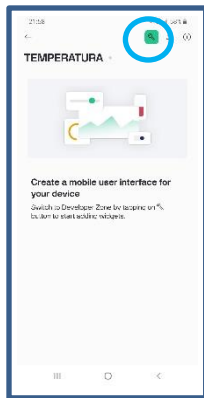
La funzione timer che richiama la sub misura().

Trasferiamo lo sketch sull'ESP32.

Ricordiamoci alla fine del caricamento di staccare/riattaccare l'alimentazione della scheda per effettuare un hard reset. Bisogna ora creare un'interfaccia utente sullo smartphone. Alcuni procedimenti sono già stati spiegati.



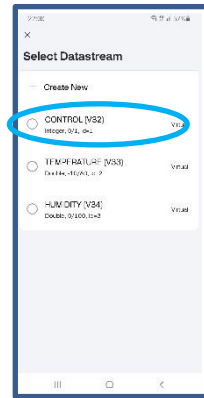
Tap su temperatura



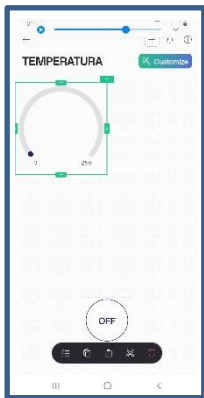
Tap sulla chiave



Tap sul +



Tap sul Button poi configurazione come switch. Datastream > Control



Ancora + > Gauge > Tap su gauge > Datastream TEMPERATURE Ripetere la procedura per il gauge dell'umidità.



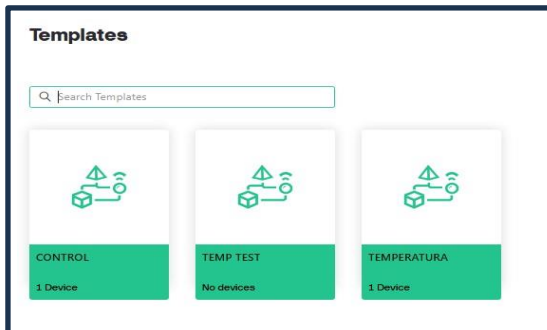
Ed ecco il risultato finale.

Potremo ora controllare via web la temperatura e l'umidità di una stanza e far partire con il pulsante Control un condizionatore o una stufa.

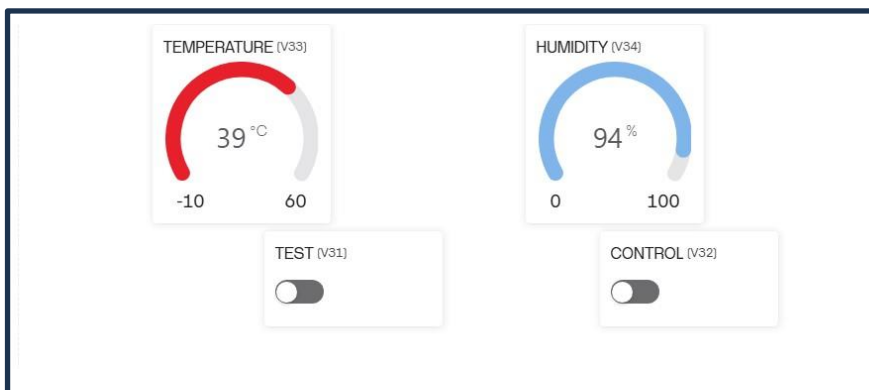
Un discorso particolare va fatto sulla funzione timer:

```
timer.setInterval(60000L,misura);
```

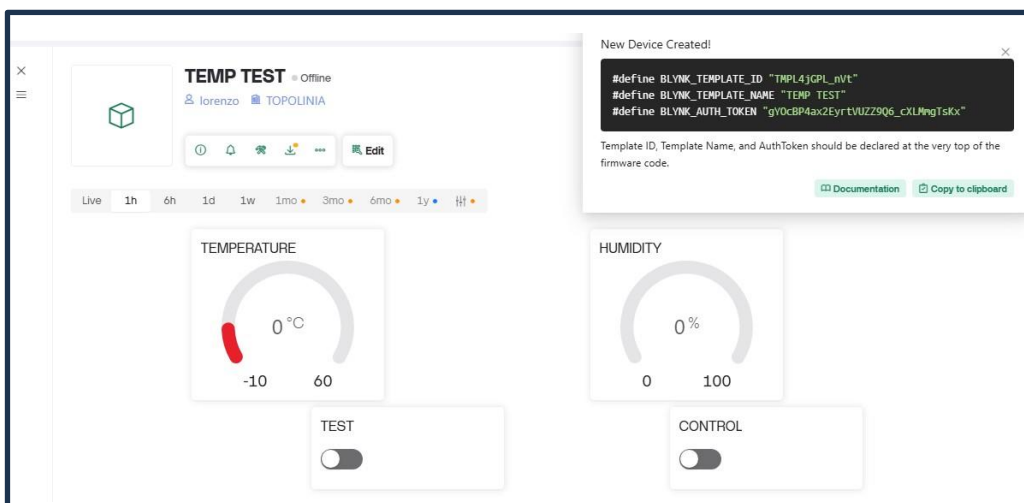
In questo caso la sub “misura()” viene richiamata ogni 60 secondi. Se dovessimo lasciare sempre acceso il dispositivo, in un mese fornirebbe 44640 dati (senza contare le attivazioni/disattivazioni del relè), che eccederebbero il massimo dei messaggi consentiti (30000), con conseguente blocco del profilo gratuito. La soluzione potrebbe essere aumentare il tempo di richiamo della sub, oppure procedere in questo modo.



Creiamo un nuovo template di nome TEMP TEST. Entriamo nel template.



Creiamo una dashboard simile alla precedente, con però uno switch in più di nome TEST, collegato al pin virtuale V31



Creiamo ora un nuovo device di nome TEMP TEST e copiamo i codici di identificazione.

Lo sketch

```
#include <DHT.h>
#define DHTPIN 33
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);

#define BLYNK_PRINT Serial

#define BLYNK_TEMPLATE_ID "TMPL4jGPL_nVt"
#define BLYNK_TEMPLATE_NAME "TEMP TEST"
#define BLYNK_AUTH_TOKEN "gY0cBP4ax2EyrntVUZZ9Q6_cXLMmgTsKx"

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

char ssid[] = "Vodafone-*****";
char pass[] = "*****";
int rel=32;

int led=26;

BlynkTimer timer;

BLYNK_WRITE(V32)
{
    int value = param.asInt();
    if (value == 1)
    {
        digitalWrite(rel, HIGH);
    }
    else
    {
        digitalWrite(rel, LOW);
    }
}

BLYNK_WRITE(V31)
{
    int value = param.asInt();
    if (value == 1)
    {
        misura();
    }
    else
    {
        //
    }
}
```

Ecco il punto in cui (grazie alla funzione BLYNK_WRITE()) azionando l'interruttore TEST (V31) verrà chiamata la sub "misura()". Per risparmiare dati spegneremo subito l'interruttore, riaccendendolo quando avremo bisogno di valori aggiornati.

```

void misura()
{
float t = dht.readTemperature();
Blynk.virtualWrite(V33, t);

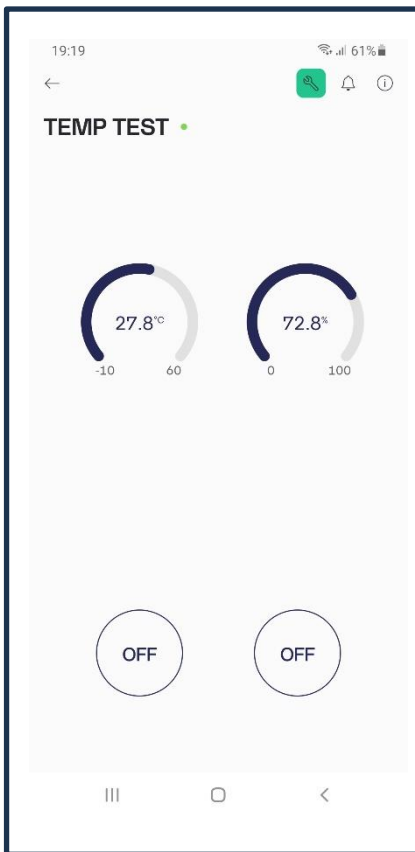
float h = dht.readHumidity();
Blynk.virtualWrite(V34, h);
delay(1000);
}

void setup()
{
pinMode(rel,OUTPUT);
pinMode(led,OUTPUT);
pinMode(DHTPIN, INPUT);
Serial.begin(115200);

Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
dht.begin();
}

void loop()
{
Blynk.run();
}

```



Il risultato finale visto sullo smartphone. I due indicatori sono fissi; si muoveranno, misurando i parametri ambientali solo alla pressione dell'interruttore a destra. L'interruttore di sinistra attiverà il relè.