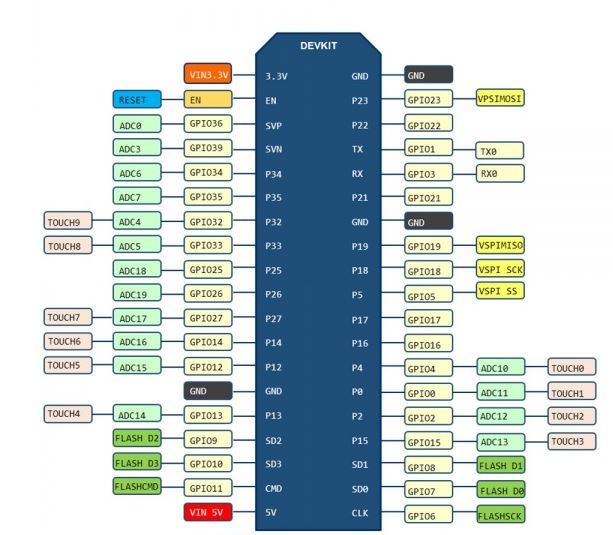
**ESP32 et BLYNK avec l'IDE ARDUINO :**

Dans cette partie nous allons voir comment faire pour que l'ESP32 soit capable de comuniquer avec une application via l'outil BLYNK.



**Présentation des broches de l'ESP32 :**

Ce schema nous est important car il nous faut absolument connaitre les types de chacune des brôches dont nous dispons afin de pouvoir au mieu utilisé l'outil BLYNK.

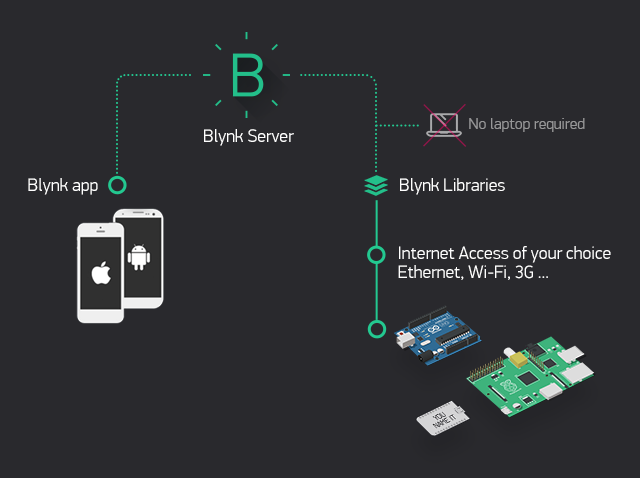
**BLYNK c'est quoi ?**

Blynk est un outil nous permettant de crée très facilement une application mobile qui contrôlera notre ESP32, il peut contrôler un hardware à distance ( comme l'ESP32) via le Wifi ou bluetooth, il peut afficher des données de capteur, les stocker et même transmettre des valeurs au système.

Blynk est composé de trois éléments :

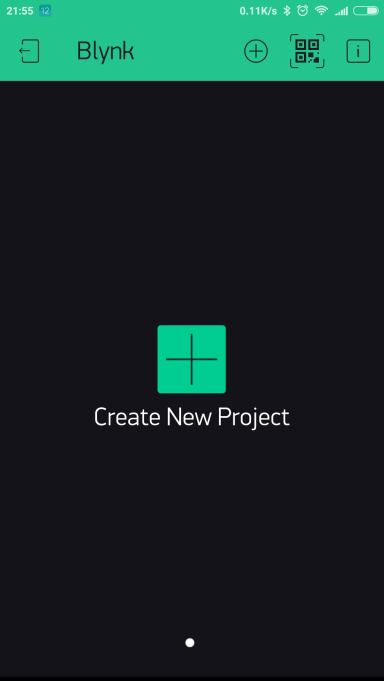
-L'application qui permet de crée une interfaces homme machine pour notre projet.

-Un serveur, c'est lui qui s'occupe de la communications entre le smartphone et l'ESP32

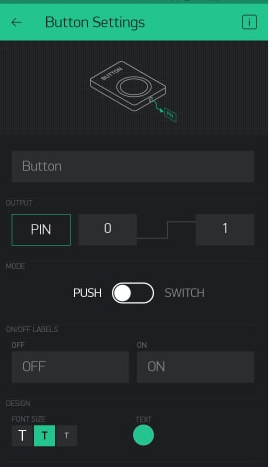
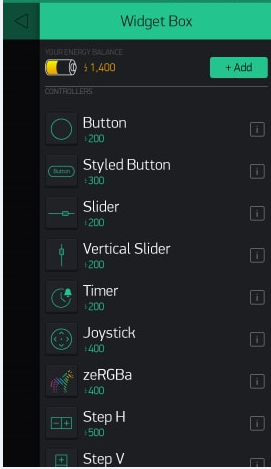
-Une Blibliothèque Blynk, présente dans l'IDE d'arduino qui permet d'activer la communication avec le serveu et traité toutes les commandes entrantes et sortantes.

**Fonctionnement de BLYNK**

Il faut tout d'abord crée un compte Blynk, une fois le compte on peut débuté coté application a crée notre interface graphique que l'on souhaite présenté a l'utilisateur.



Une fois le projet crée il suffit d'ajouter des éléments a l'écran, par exemple ici on va crée un bouton.



La case PIN permet de configuré l'endroit où va agir le bouton sur l'ESP32, les valeurs 0 et 1 sont les deux valeur dans les deux états du bouton ici, en position basse on envoi la valeur 0 et en position haute on envoi la valeur 1, on peut réglé le type de bouton si on veut un bouton poussoir ou un interrupteur.

Une fois notre interface graphique paramétré il faut récupéré le jetons de connexion a l'application, on peut le recevoir par mail ou le trouver directement dans les paramètre de notre projet.

Maintenant il faut crée la connexion entre l'ESP32 et notre projet, pour ce faire on utilise l'IDE Arduino avec la librairie de BLYNK.

|  |
| --- |
| #include <BlynkSimpleEsp32.h> |

on ajoute aussi les librairie pour la connexion WIFI :

|  |
| --- |
| #include <WiFi.h>  #include <WiFiClient.h> |

on initialise nos variable pour crée la connexion :

|  |
| --- |
| char auth[] = "b20ddc1e0efb4833b30ce8fd57fe5721"; // Jeton d'accès pour l'esp32  char ssid[] = "projet"; // Le nom du réseau sur lequel le smartphone et esp32 sont connecté  char pass[] = "123456"; // le mot de passe du réseau |

dans la partie setup on établie la connexion :

|  |
| --- |
| Serial.begin(115200);  delay(10);  Serial.print("Connecting to ");  Serial.println(ssid);  WiFi.begin(ssid, pass);  int wifi\_ctr = 0;  while (WiFi.status() != WL\_CONNECTED) {  delay(500);  Serial.print(".");  }  Serial.println("WiFi connected");  Blynk.begin(pass, ssid, pass); |

maintenant que le lien entre l'ESP32 et l'application est crée, on peut agir sur les broches de l'ESP32 via notre interface graphique précédemment crée.

Par exemple si on a relié une led au bouton poussoir il suffit de faire :

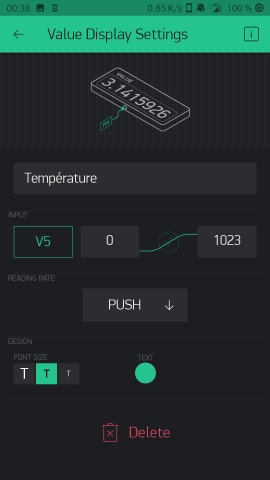
|  |
| --- |
| Blynk.run(); |

Dans notre projet nous devons interagir avec différent capteur et contrôleurs en simultané, chacun des capteur et contrôleurs a une variable associer, par exemple le capteur de température aura une variable "temps" qu'il faudra affiché sur l'application on devra donc envoyer la variable temps de l'esp32 a l'application et l'application doit savoir récupéré cette valeur, pour géré ce flux on va utilisé des broches virtuel.

Au niveau de l'IDE Arduino il faut donc après avoir récupéré notre data l'envoyer a l'application de la manière suivante :

|  |
| --- |
| BLYNK\_READ(V5)  {  Blynk.virtualWrite(5, temps);  } |

V5 correspond a la broche virtuel 5, on envoi donc notre data temps via la fonction virtualWritre présente dans la librairie de BLYNK.

Une fois envoyer pour l'afficher sur l'application il suffit d'ajouter un afficheur relier au port virtuel 5 de la manière suivante :

Voici un exemple complet pour la lecture et l'affichage d'une température récupéré sur l'ESP32 et afficher sur l'application :

