



Open-Bouton



Clément

[VIEW IN BROWSER](#)

updated 12. 4. 2022 | published 17. 12. 2021

Summary

Open-Bouton If French is not your main language, we recommend you to check the English Version Explications Ce...

[Hobby & Makers](#) > [Electronics](#)

Tags: [3dprinting](#) [arduino](#) [esp8266](#) [button](#) [3dprinter](#)
[raspberrypi](#) [opensource](#) [rotaryencoder](#) [discord](#) [fablab](#)
[bot](#) [github](#) [bouton](#) [labouest](#) [openlab](#)

Open-Bouton

If French is not your main language, we recommend you to check the [English Version](#)

Explications

Ce programme repose sur un Arduino ESP 8266 et un Raspberry Pi connectés via DNS

Un bouton fait maison est également à imprimer en 3D et à assembler

Dans le dossier Raspberry figure le code Python pour le Raspberry :

- bot.py : Fichier Python principal, analyse la reception UDP et gère l'envoi des messages vers Discord
- udp_receiver.py : Fichier Python qui gère la reception UDP et retransmet les messages à bot.py
- config.py : Fichier de configuration de la partie Raspberry

data.txt : Fichier tampon dans lequel sont stockées les commandes UDP avant d'être traitées Dans le dossier Arduino figure le code C++ pour l'ESP :

main.ino : Fichier C++ principal, il gère le lien entre les composants ainsi que la communication UDP

- rotary.ino : Fichier C++ qui contrôle la reception du signal transmis par le bouton rotatif
- neoled.ino : Fichier C++ qui contrôle l'allumage des LEDs

Les [].h : Déclaration des classes utilisées dans les fichiers .ino correspondants Dans le dossier Montage figurent les fichiers pour l'impression 3D du bouton ainsi que le diagramme de l'électronique

Installation

Prérequis

- Un [Arduino ESP 8266](#)
- Un [Raspberry Pi](#)
- Un encodeur [KY-040](#)
- Une [bande de leds]

De quoi imprimer en 3D le bouton ### Partie Discord

Il va vous falloir [créer votre serveur Discord](#) puis [créer votre bot](#) (Arrêtez vous au point 7 sur ce dernier tutoriel)

N'oubliez pas de conserver l'identifiant de votre Bot et de [récupérer les identifiants du serveur et du channel](#) sur lequel vous voulez envoyer les messages

Partie Raspberry

En premier lieu, il vous faudra installer Raspbian sur le raspberry, un tutoriel à ce sujet est disponible [ici](#)

- Il faut ensuite vérifier que Python 3 soit installé sur le raspberry, pour cela tapez python --version puis python3 --version dans un terminal et vérifiez bien qu'une de ces deux commandes renvoie un python 3.X

- Ensuite, téléchargez le contenu du dossier Raspberry et envoyez-le sur le Raspberry (des tutoriels sont accessibles facilement sur Internet)
- Placez tous les fichiers dans /home/pi/udp_esp_button (vous devrez créer le dossier)
- Vous pouvez alors modifier le fichier config.py, les instructions sont dans le fichier lui-même
- Dans le fichier start.sh, vous allez pouvoir enlever le # devant la ligne qui renvoyais Python 3.X précédemment
- Ensuite pour devrez **faire un serveur DNS** sur votre Raspberry
- Accordez le droit à pi de lancer le bot avec `cd /home/pi/udp_esp_button puis chmod 754 start.sh`

Pour lancer la machine, ouvrez le dossier /home/pi/udp_esp_button dans le terminal et tapez `./start.sh #####` Lancer le script au démarrage

Si votre Raspberry redémarre, vous devrez relancer le script vous-même, pour pallier à cela, vous pouvez le faire se lancer au démarrage comme cela :

Tapez `sudo nano /etc/rc.local` dans un terminal

- Avant la dernière ligne (exit 0), tapez `bash /home/pi/udp_esp_button/start.sh`

Sauvegardez et fermez le fichier ##### Partie Arduino

Installez la librairie **WifiManager**

- Téléchargez le dossier Arduino et ouvrez le fichier main/main.ino qu'il contient dans l'**IDE Arduino**.

Dans ce fichier, changez si vous le voulez

- Le port de communication UDP à la ligne 16
 - L'IP du Raspberry Pi à la ligne 20

Après avoir vérifié que la modèle de carte est bien défini, uploadez le fichier dans l'ESP, il détectera et uploadera tout seul chacun des fichiers annexes.

Partie Electronique

Branchez la bande de LEDs et l'encodeur comme indiquée sur l'image ci-dessous :

Une alimentation de 5V est nécessaire au bon fonctionnement du système, n'importe quelle alimentation micro-usb type chargeur de portable suffira

(le système total a été testé sur 5V à environ 500mA) ##### Photo du système final

Utilisation

Allumage du bot

Pour allumer le bot, vous pouvez faire `./home/pi/udp_esp_button/start.sh`

Pour éteindre le bot, envoyez par UDP ceci : stopbot

License

La **license** appliquée à ce projet est une License GNU Public License v3
Cette license autorise le partage, la modification et la redistribution du projet sous ces conditions :

- Le code source de votre projet doit être disponible gratuitement
 - Votre projet doit comporter la license GPLv3 ainsi qu'une mention de copyright en notre faveur
 - Les modifications apportées doivent être clairement énoncées
- Toutes les spécifications de cette license sont écrites en anglais dans le fichier éponyme. En cas de problème

En cas de problème, n'hésitez pas à nous contacter par mail :

- aubin.sionville@orange.fr

cle.chec@laposte.net Ou sur notre **serveur Discord**-

Nous soutenir

Adresse BTC : 3Hkvtn6uYy27X76buSNepsQcZ77692cLMm

Category: Electronics

Model files

fond.stl





dessus.stl



bouton_v3.stl

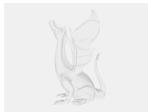
Other files



license-gplv3-blue_95404.svg



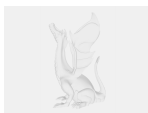
made-with-c-plus-plus_95404.svg



made-with-python_95404.svg



license-gplv3-blue_95404.svg



made-with-c-plus-plus_95404.svg



made-with-python_95404.svg

[Find source .stl files on Thingiverse.com](#)

License ©

This work is licensed under a
GNU



General Public License v2.0

- ✘ | Sharing without ATTRIBUTION
- ✓ | Remix Culture allowed
- ✓ | Commercial Use
- ✓ | Meets Open Definition
- i | Share under the same license