


Boite aux lettres connectée

Boîte aux lettres connectée à Twitter avec Arduino.

 Difficulté **Difficile**

 Durée **10 heure(s)**

 Catégories **Électronique, Énergie, Mobilier, Maison**

 Coût **40 EUR (€)**

Sommaire

Introduction

Étape 1 - Principe de fonctionnement

Étape 2 - Fabrication de la structure principale

Étape 3 - Mise en place des leds

Étape 4 - Câblage électronique

Étape 5 - Gestion de l'alimentation

Étape 6 - Utiliser Twitter avec un ESP8266

Étape 7 - Programmation de l'ESP8266

Étape 8 - Conclusion

Notes et références

Commentaires

Introduction

Rien de plus énervant que de devoir aller ouvrir sa boîte aux lettres toutes les 5 minutes lorsque l'on attend avec impatience son nouveau PC ou le dernier composant permettant de finir un projet. Voici la solution !! Une boîte aux lettres connectée vous avertissant sur Twitter lorsque le facteur passe.

C'est face à cette frustration que j'ai eu l'idée de construire un système m'avertissant quant il y a du courrier dans ma boîte aux lettres.

Matériaux

Pour l'électronique:

- un ESP8266 ESP-07 et son support
- une RTC DS3231
- un régulateur 3,3V , LM1117 (ou équivalent) avec deux condensateurs de 10 µF
- un mosfet IRF520
- une diode 1N4007
- une photorésistance
- une résistance 1 KOhms et une résistance 330 Ohms
- une batterie au plomb 6V 4Ah
- un contrôleur de charge charge solaire
- deux panneaux solaires 12V 3W
- un câble U.FL vers RP-SMA de 1m environ
- une antenne wifi RP-SMA
- Ruban de leds blanches 12V
- un élévateur de tension
- un interrupteur à bascule
- des connecteurs mâles/femelles pour circuit imprimé
- un bornier
- une plaque d'essai
- du fil électrique
- de la soudure

Pour la structure:

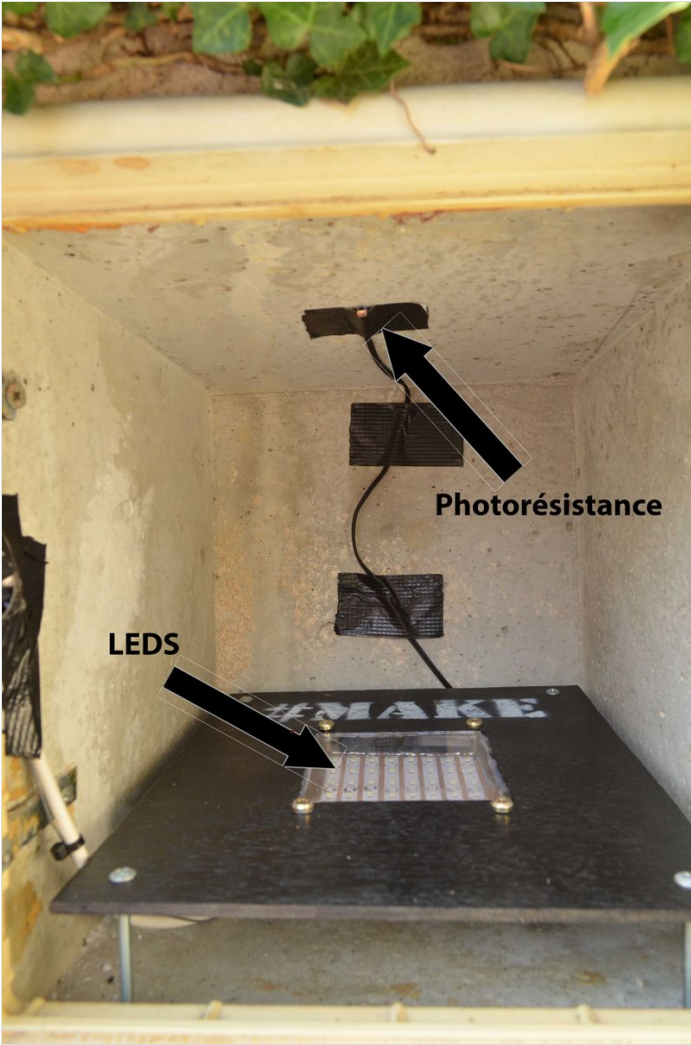
- une planche de contreplaqué (environ 400 * 300 * 40 mm)
- 4 boulons M4 60mm
- 4 boulons M4 30mm
- de la peinture
- une boîte de dérivation 80 * 80 mm
- de l'adhésif double face
- une plaque de plexiglas (environ 350 * 170 * 4 mm)
- des serflex

Outils

- un fer à souder
- une scie
- tournevis
- une perceuse
- pistolet à colle chaude
- un multimètre
- une équerre
- un régllet
- matériel de protection
- convertisseur usb série


Étape 1 - Principe de fonctionnement

Le courrier est détecté grâce à des LEDs et une photorésistance. Lorsque le courrier est déposé il tombe sur les LEDs disposées au fond, ce qui modifie la luminosité à l'intérieur, changement qui est détecté par la photorésistance. L'ESP 8266 détecte ainsi la présence de courrier. Il se connecte alors au réseau wifi puis publie un message sur twitter informant du passage du facteur ou du livreur. Le tout fonctionne de manière autonome grâce à des panneaux solaire et une batterie




Étape 2 - Fabrication de la structure principale

Commencez par découper la plaque de contreplaqué au dimension de votre boîte aux lettres.

 Prenez des dimension légèrement inférieur à celle de la boîte aux lettres pour pouvoir sortir et remettre facilement le système

Puis en son centre découpez un carré de 100*100 mm

 Il peut être intéressant de peindre le bois pour qui résiste dans le temps

Découpez un carré de 100*100 mm dans du plexiglas.

Insérez le dans la plaque en bois. Pour le maintenir en place utilisez de la colle chaude ou de la colle plexiglas.

Enfin percez dans les coins, 4 trous de diamètre 4 puis montez y un boulon de 60mm. Ces derniers servent de pieds pour maintenir la structure en hauteur.

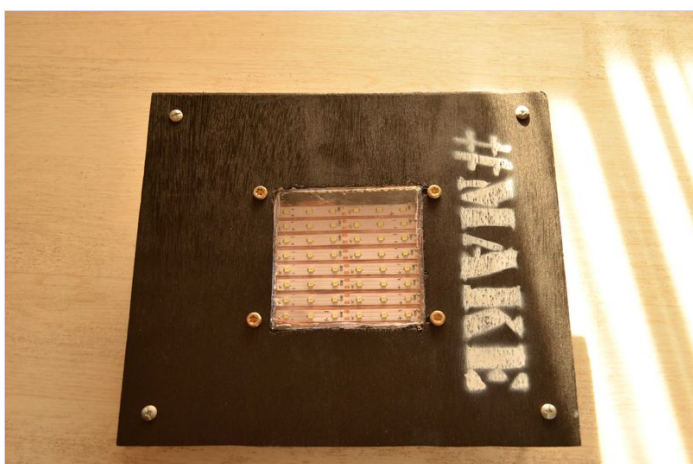
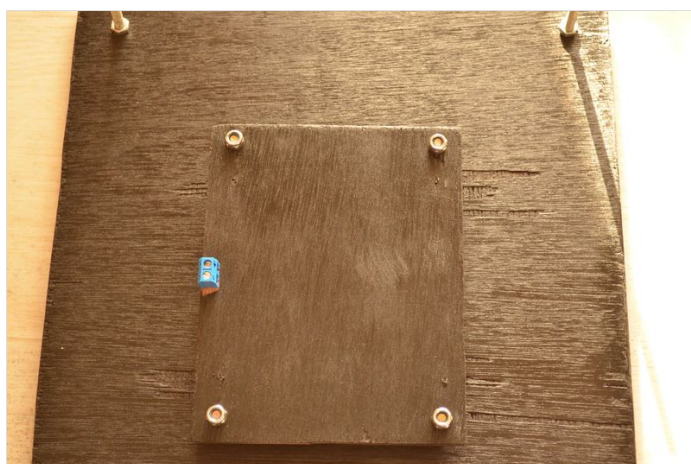
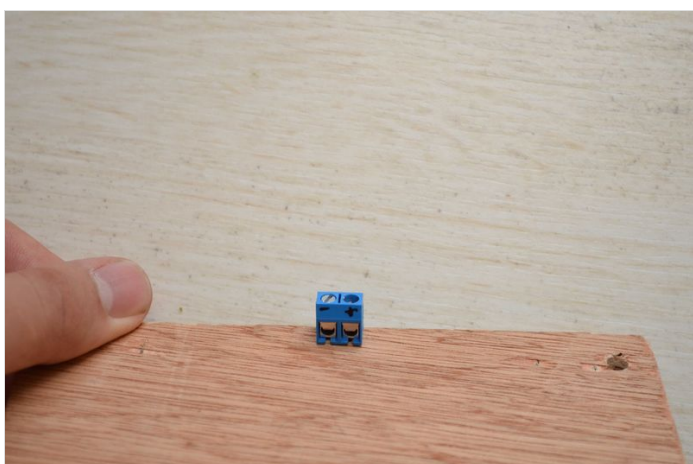
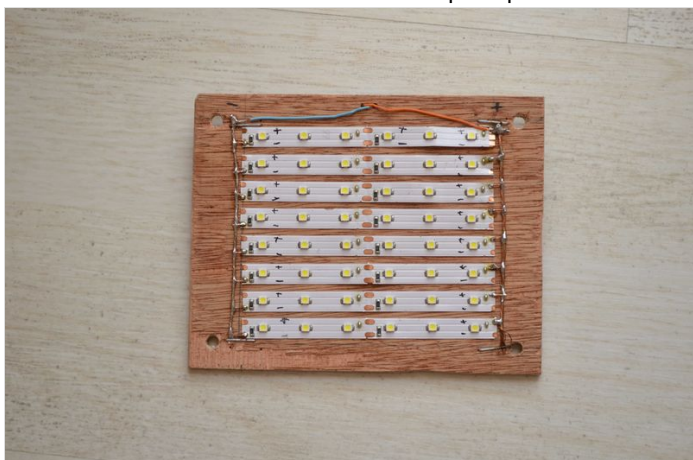


Étape 3 - Mise en place des leds

Pour avoir un système de LEDs suffisamment puissant pour que les variations de luminosité soient détectés je me suis inspiré des panneaux à LEDs DIY utilisés en vidéo (Cf vidéos d'Experimentboy ou DIY Perks)

Pour se faire, commencez par découper une plaque de contreplaqué de 140*140mm. Puis collez y 8/10 bandes de leds. Ensuite reliez ensemble d'un coté tous les pôles "+" et de l'autre tous les pôles "-". Et ensuite placez un bornier de l'autre coté de la plaque pour l'alimentation.

Pour finir assemblez les LEDs à la structure principale à l'aide de 4 boulons M4 30mm



Étape 4 - Câblage électronique

Comme dis précédemment le système repose sur un ESP8266 qui sera programmé à l'aide de l'IDE Arduino. Pour alimenter ce dernier il sera nécessaire d'utiliser un LM1117 couplé à deux condensateurs de 10 μ F pour réguler la tension à 3,3V.

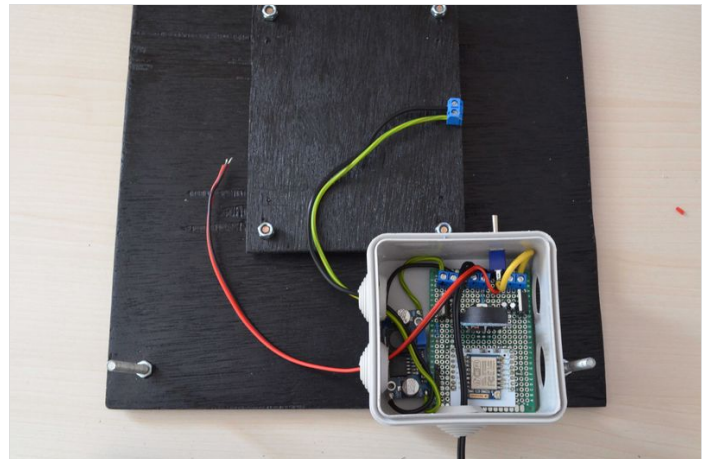
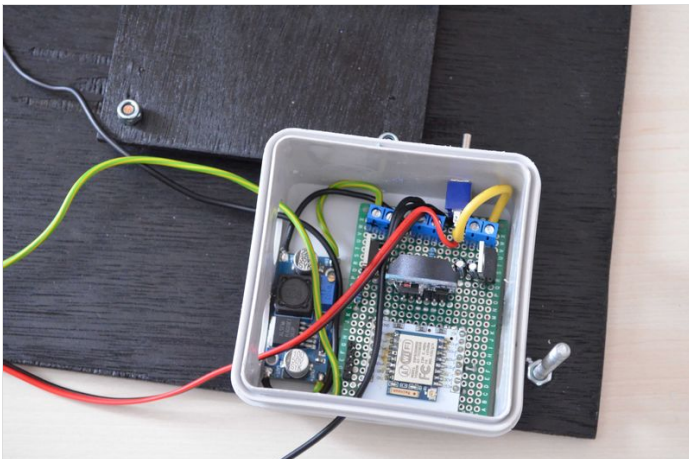
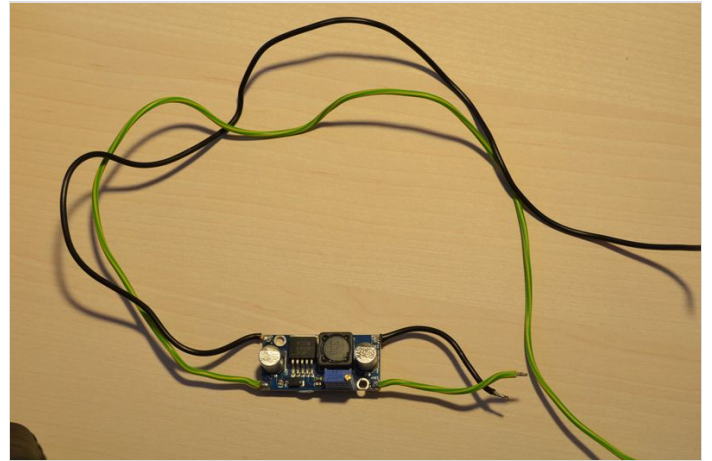
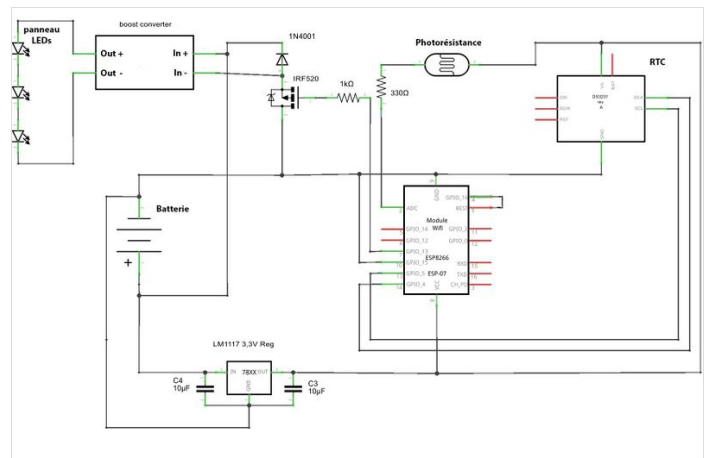
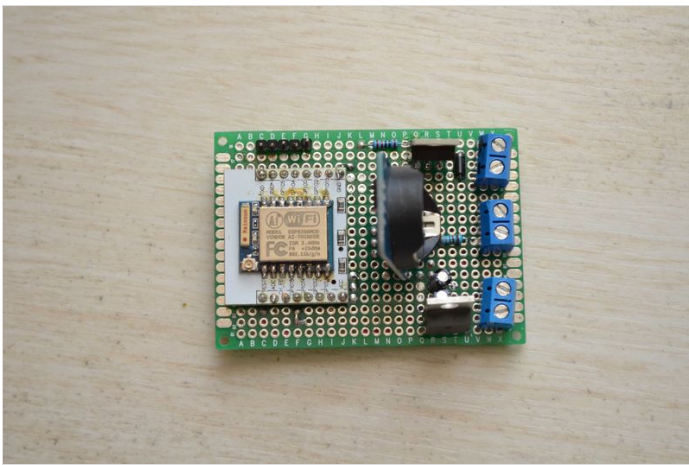
On trouve ensuite la photorésistance chargée de détecter les variations de luminosité reliée par l'intermédiaire d'une résistance de 330 Ohms à la broche ADC de l'ESP.

Les LEDs nécessitant un courant supérieur à ce qu'est capable de fournir l'ESP elles sont contrôlées par un l'intermédiaire d'un Mosfet (IRF520 / 530 / 540) relié par l'intermédiaire d'une résistance de 1K Ohms à la broche 7 (GPIO13) de l'ESP.. De plus comme ces dernières doivent être alimentées en 12V et que la batterie utilisée ne fournit que 6V, il est nécessaire d'ajouter un élévateur de tension (boost converter).

Et pour finir on retrouve une RTC DS3231 qui permet au système d'être toujours à l'heure après un reboot. Cette dernière est reliée à l'ESP par l'intermédiaire d'une liaison i2C disponible sur les broches 13 (GPIO 5) et 14 (GPIO 4)

Une fois tous les composants soudés sur une plaque d'essai installez le tout dans une boîte de dérivation et fixez la à la structure principale.

💡 Si votre box internet se trouve être loin de votre boîte aux lettres (comme dans mon cas) je vous conseille de déporter l'antenne wifi de l'ESP en utilisant câble U.FL vers RP-SMA et une antenne wifi RP-SMA. Vous pouvez ensuite la fixer avec les panneaux solaires par exemple.



Étape 5 - Gestion de l'alimentation

Pour alimenter le système j'ai choisi d'utiliser une batterie au plomb 6V de 4A couplé à deux panneaux solaire de 250 mA (valeur théorique loin de la réalité...) branchés en parallèle le tout géré par un régulateur de charge.

Il aurait sûrement été plus rentable niveau prix/capacité/taille d'utiliser des batteries au lithium. Mais ces dernières étant sensibles aux différences de température, à l'humidité et nécessitant des circuits de recharge plutôt compliqués, j'ai préféré rester sur une batterie classique au plomb.

Laisser les LEDs allumées en permanence entraîne une consommation bien trop importante (autours des 250mA) ce qui empêche d'alimenter le circuit sur batterie. La solution est donc de calibrer une fois le système sans courrier puis de faire des mesure à intervalles réguliers pour détecter la présence de courrier.

Ainsi le circuit consomme environ 75mA (non connecté à un réseau wifi), ce qui donne avec la batterie une autonomie d'environ deux jours. D'après mes mesures les panneaux solaires fournissent au maximum 300 mA en plein soleil mais cela chute au alentours de 100 mA avec un ciel nuageux. Ce qui sur une base de 10h de soleil par jour étend l'autonomie à environ 5 jours...

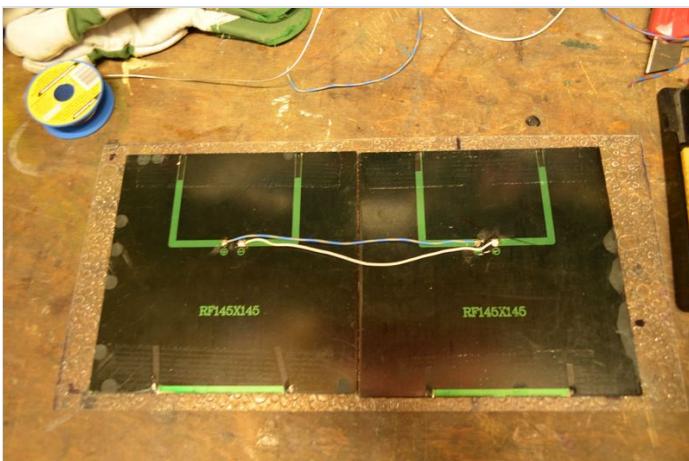
La solution pour ne pas avoir à recharger le système est donc d'utiliser une RTC et la fonction *sleep mode* de l'ESP. Cette fonction s'active en reliant la broche 4 (GPIO 16) à la broche 1 (reset) et au niveau du code avec:

```
ESP.deepSleep(temps en micro seconde);
```

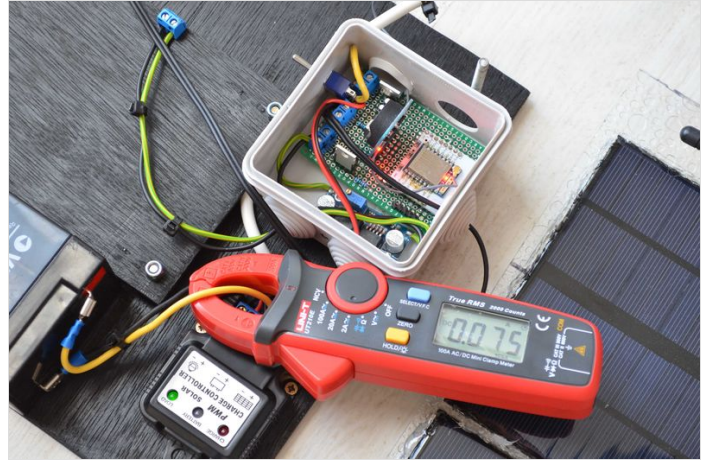
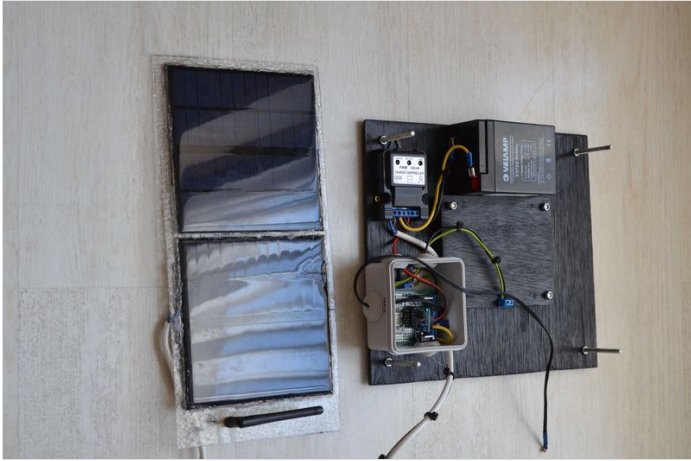
Elle a pour effet de mettre en "sommeil" l'ESP8266 réduisant sa consommation au alentour de 1mA mais entraîne un reboot de ce dernier à son réveil. C'est ici qu'intervient la RTC, qui permet au système de se "repérer" dans le temps.

Connaissant à peut près les horaires de passage de mon facteur, j'ai décidé que l'ESP resterait allumé entre 10h et 14h et vérifierait sur cette plage horaire la présence de courrier toutes les 5 minutes. Le reste du temps (de 14h à 10h) l'ESP est en sommeil, et le circuit consomme environ 10 mA (consommation du à la RTC). Ce qui permet en théorie au système de fonctionner indéfiniment !!

En pratique le système à déjà fonctionné une semaine d'été sans problème. J'attend d'avoir encore un peut plus de recul pour être sur de son autonomie dans le temps







Étape 6 - Utiliser Twitter avec un ESP8266

Je vous conseille fortement de créer un compte Twitter dédié à votre ESP et de le mettre en privé, comme ça vos followers ne seront pas tous informer que vous avez du courrier

Pour permettre à l'ESP8266 nous allons utiliser la *Tweet Library*.

<https://arduino-tweet.appspot.com/>

Ensuite pour pouvoir Tweeter vous allez devoir accorder l'accès au compte Twitter via une clef. Pour l'obtenir rendez vous sur cette page:

https://api.twitter.com/oauth/authorize?oauth_nonce=2828234892369209436&oauth_timestamp=1500815756&oauth_consumer_key=oQA2jr3rWowM4SpGB64yQ&oauth_signature_method=HMAC-SHA1&oauth_version=1.0&oauth_token=DHDffgAAAAAAViGAAABXW-XDZc&oauth_signature=jPqM1kBNpqhVUIDJMFQn%2BB92uo4%3D

Bien sur l'accès peut être supprimé a tout moment via vos paramètre twitter: <https://twitter.com/settings/applications>

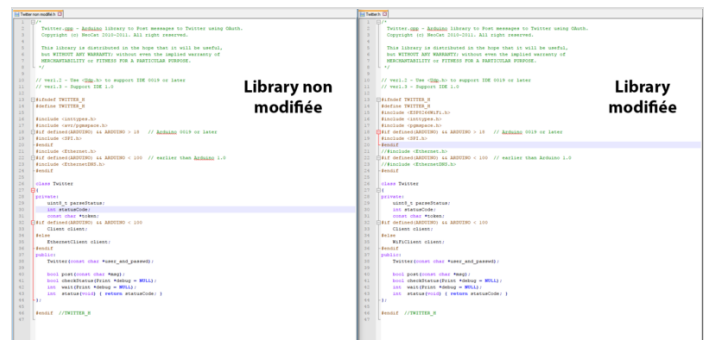
Malheureusement la *Tweet Library* n'est pas nativement compatible avec les ESP car elle a été développée pour être utilisée avec un Arduino et un shield ethernet. Pour la rendre compatible il va vous falloir modifier le header (*Twitter.h*)

Pour cela rajoutez ligne 15

```
#include <ESP8266Wifi.h>
```

prenez en commentaire les lignes 21 et 23
et remplacez *EthernetClient* client; ligne 35 par

```
WifiClient client;
```



Étape 7 - Programmation de l'ESP8266

Pour programmer l'ESP8266 on va utiliser l'IDE Arduino. Pour cela, il vous faut tout d'abord ajouter le support de ce type de carte dans l'IDE en ajoutant l'URL suivante dans Fichier>Préférences>URL de gestionnaire de cartes supplémentaires :

http://arduino.esp8266.com/package_esp8266com_index.json, http://arduino.esp8266.com/stable/package_esp8266com_index.json

Puis choisir "Generic ESP8266 Module " dans Outils>types de carte.

Et enfin relier l'ESP au convertisseur USB/série comme indiqué sur le schéma.

Téléchargez et installez ensuite la librairie RTC développée par Makuna pour gérer la DS3231:

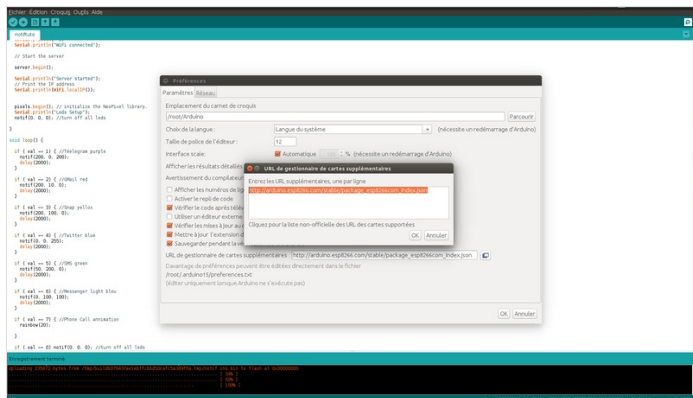
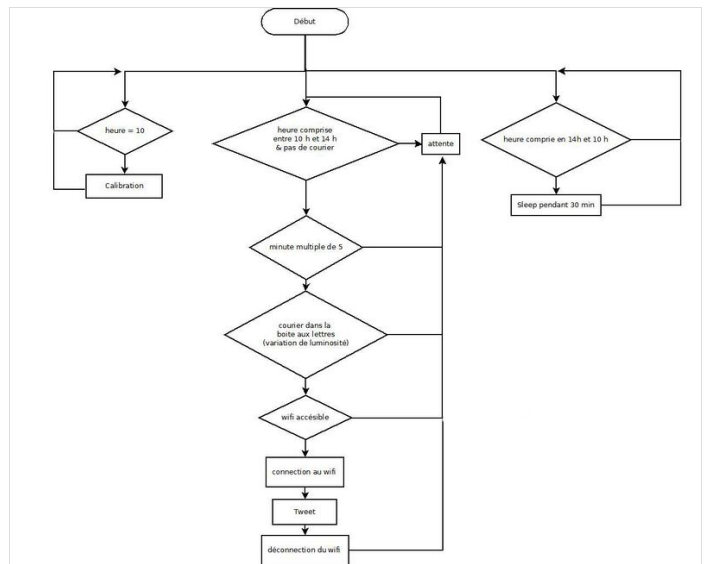
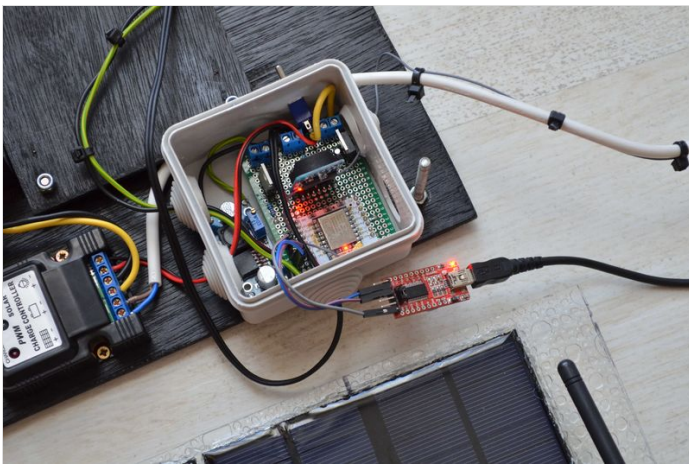
<https://github.com/Makuna/Rtc>

Puis exécuter le programme DS3231_Simple pour mettre à l'heure la RTC.

Et enfin vous pouvez envoyer sur l'ESP8266 le programme suivant pour gérer votre boîte aux lettres connectée (Cf algorithme pour les détails sur son fonctionnement):

<https://drive.google.com/open?id=0B8tCTkPLfNNraHhIbnNaTmx3VIU>

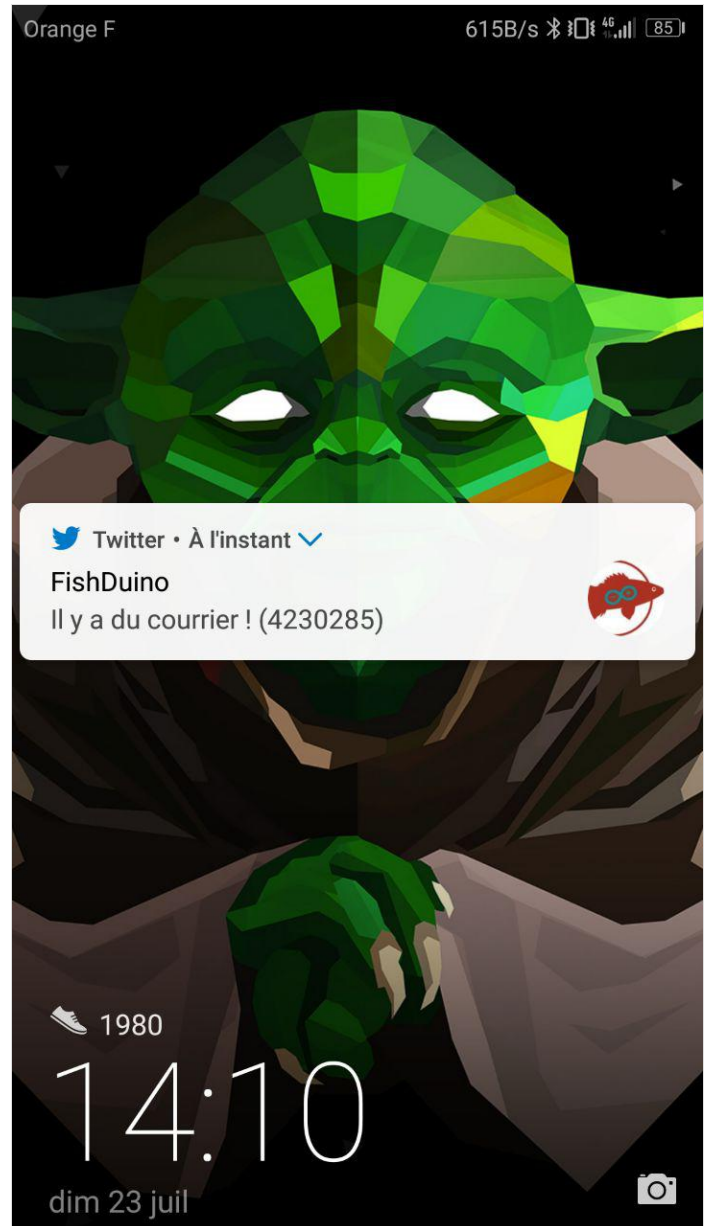
Pensez à adapter les plages de fonctionnement en fonction de vos besoins



Étape 8 - Conclusion

Ce tutoriel arrive à son terme, j'espère avoir été clair dans mes explications. Cependant, si jamais vous ne comprenez pas un point n'hésitez pas à engager la conversation ☺ En tous cas si vous reproduisez ce système ou que vous avez des idées d'amélioration partagez les !!!

💡 J'ai choisi d'utiliser Twitter comme moyen de notification mais il est tout a fait envisageable de charger ce point. Je pense par exemple à une lumière allumé sans fil, ou un système de requête http ou même un couplage avec un projet domotique comme <https://gladysproject.com/fr/> ☺



Notes et références

Hackable magazine numéro 2 et 7

ESP8266:

- <http://arduino.esp8266.com/versions/1.6.5-1160-gef26c5f/doc/reference.html>
- <http://esp8266.github.io/Arduino/versions/2.0.0/doc/boards.html>
- <http://esp8266.github.io/Arduino/versions/2.0.0/doc/libraries.html>
- http://www.esp8266.com/wiki/doku.php?id=esp8266_power_usage
- <https://www.losant.com/blog/making-the-esp8266-low-powered-with-deep-sleep>

Tweet Library:

- <https://arduino-tweet.appspot.com/>
- <https://gist.github.com/hofmannsven/6226895>
- <https://hofmannsven.com/2013/laboratory/arduino-twitter-library/>

Choix de la batterie:

- <https://www.youtube.com/watch?v=LqgP16JQ24I>