Préalable : IL FAUT QUE TOUT LE MONDE SOIT CONNECTEE AU MEME RESEAU

ATTENTION : Dans le programme, j’utilise SerialWIFI, mais par défaut, lors de l’installation, la classe déclarée est généralement quelque chose comme TelnetStream

Table des matières

[Explication du fonctionnement de l'OTA sur un ESP32 (Pour Moi en soi, une sorte de rappel) 2](#_Toc179746891)

[Étapes pour configurer le Moniteur Série sur VSCode 4](#_Toc179746892)

[Activer Telnet sur Windows 6](#_Toc179746893)

# Explication du fonctionnement de l'OTA sur un ESP32 (Pour Moi en soi, une sorte de rappel)

**Introduction à la programmation OTA (Over-the-Air)**

La programmation OTA permet de mettre à jour le code des cartes ESP32 sans avoir besoin de les connecter physiquement à un ordinateur via un port série. En utilisant l'OTA Web Updater, il est possible de téléverser du nouveau code sur l'ESP32 directement depuis un navigateur web lorsque l'ESP32 et votre navigateur sont connectés au même réseau local.

**Principe de fonctionnement de l'OTA Web Updater**

1. **Premier téléversement via port série** : Le premier programme (ou "sketch") doit être téléversé sur l'ESP32 via une connexion série traditionnelle. Ce sketch contient le code qui permet d'activer la fonctionnalité de mise à jour OTA.
2. **Création d'un serveur web** : Le sketch OTA Web Updater crée un serveur web sur l'ESP32. Ce serveur vous permet d'accéder à une interface via un navigateur où vous pouvez téléverser du nouveau code.
3. **Téléversement via OTA** : Pour chaque mise à jour ultérieure, il est nécessaire d'intégrer le code OTA dans tous les nouveaux sketches. Cela garantit que vous pourrez continuer à téléverser du code via OTA à l'avenir.
4. **Limite** : Si vous téléversez un code qui ne contient pas la routine OTA, vous perdrez la possibilité de mettre à jour l'ESP32 via OTA. Il faudra alors recommencer en utilisant la connexion série.

**OTA dans le Bootloader de l'ESP32**

Le processus de mise à jour OTA sur l'ESP32 repose sur le bootloader, qui est le premier logiciel exécuté au démarrage. Ce bootloader gère les partitions de mémoire nécessaires pour effectuer des mises à jour OTA de manière fiable et sécurisée.

**Configuration des Partitions**

Pour le bon fonctionnement de l'OTA, l'ESP32 dispose de plusieurs partitions dans sa mémoire flash :

* **Partitions d'application** : Deux partitions de firmware sont généralement présentes, souvent appelées ota\_0 et ota\_1. L'une est active et exécute le firmware actuel, tandis que l'autre est prête à recevoir de nouvelles mises à jour.
* **Partition de Données OTA** : Une partition spéciale, appelée *OTA data*, stocke des informations sur la dernière mise à jour réussie. Le bootloader lit cette partition au démarrage pour déterminer quelle image de firmware charger. Si aucune donnée valide n'est trouvée, l'image d'usine est chargée.

**Processus de Mise à Jour OTA**

1. **Démarrer le processus OTA** : L'OTA commence par effacer la partition cible pour faire de la place pour le nouveau firmware.
2. **Écriture du nouveau firmware** : Le nouveau firmware est écrit dans la partition inactive, assurant que le fichier est correctement transféré.
3. **Terminer la mise à jour** : Une fois l'écriture terminée, l'intégrité de l'image nouvellement écrite est validée pour garantir son exactitude.
4. **Changer la partition de démarrage** : La partition de données OTA est mise à jour pour pointer vers la nouvelle partition contenant le firmware mis à jour.

**Redémarrage et Vérification**

Après la mise à jour OTA, le microcontrôleur redémarre. Le bootloader effectue une vérification de l'intégrité du nouveau firmware.

* Si la mise à jour est réussie et valide, le bootloader bascule vers la nouvelle partition, chargeant ainsi le nouveau firmware lors du prochain démarrage.
* En cas d'échec de la mise à jour (par exemple, si le fichier est corrompu), le bootloader retourne à la partition précédente, assurant que le microcontrôleur ne soit pas "briqué".

# Étapes pour configurer le Moniteur Série sur VSCode

1. **Installer les Extensions Nécessaires** :
   * Assurez-vous d'avoir installé l'extension **PlatformIO** ou **Arduino** pour VSCode. Ces extensions facilitent la gestion des projets et l'utilisation du moniteur série.
2. **Ouvrir le Moniteur Série** :
   * Si vous utilisez **PlatformIO**, vous pouvez ouvrir le moniteur série en utilisant l'icône de moniteur série ou en utilisant la commande PlatformIO: Serial Monitor à partir de la palette de commandes (Ctrl+Shift+P).
   * Si vous utilisez l'extension **Arduino**, vous pouvez utiliser l'option Ouvrir le moniteur série.
3. **Configuration du Moniteur Série** :
   * Lorsque vous ouvrez le moniteur série, vous devez spécifier le bon port et la vitesse de communication.
   * Dans NOTRE cas, l'ESP32 est accessible via Telnet, donc vous devez vous connecter à l'adresse BaseEsp32.local sur le port 23. Toutefois, les moniteurs série de VSCode ne se connectent pas directement via Telnet.

**Commandes Telnet**

Pour établir une connexion Telnet avec votre ESP32 depuis le terminal de VSCode, vous pouvez procéder comme suit :

1. **Ouvrir le Terminal** :
   * Allez dans Affichage > Terminal (ou appuyez sur Ctrl+`` `) pour ouvrir le terminal intégré de VSCode.
2. **Utiliser la Commande Telnet** :
   * Dans le terminal, tapez la commande suivante pour vous connecter à votre ESP32 : telnet BaseEsp32.local 23

**Configuration du Moniteur Série**

Pour les paramètres de communication, vous pouvez définir les paramètres suivants :

* **Vitesse de transmission (Baud rate)** **(Cas en liaison Série physique)** : Cela dépend de votre configuration dans le code de l'ESP32. Un réglage commun est 115200, mais vérifiez votre code pour confirmer.
* **Port(Cas en liaison Wifi par exemple)** : Comme mentionné, l'adresse sera BaseEsp32.local et le port 23 pour Telnet

# Activer Telnet sur Windows

1. **Ouvrir le Panneau de Configuration** :
   * Cliquez sur le bouton **Démarrer** et tapez "Panneau de configuration", puis appuyez sur **Entrée**.
2. **Accéder aux Programmes et Fonctionnalités** :
   * Cliquez sur **Programmes** puis sur **Activer ou désactiver des fonctionnalités Windows**.
3. **Activer le Client Telnet** :
   * Dans la fenêtre qui s'ouvre, faites défiler la liste jusqu'à trouver **Client Telnet**.
   * Cochez la case à côté de **Client Telnet**.
   * Cliquez sur **OK** pour installer le client.
4. **Redémarrer le Terminal** :
   * Après l’installation, fermez et rouvrez le terminal dans Visual Studio Code ou utilisez une nouvelle fenêtre de terminal.

Voici un lien vers une vidéo qui montre comment activer Telnet sur Windows. <https://www.youtube.com/watch?v=Q2sNkTJv2fc>

**Tester la Connexion Telnet**

Une fois Telnet activé, vous devriez pouvoir exécuter la commande suivante dans le terminal : telnet BaseEsp32.local 23

Dans mon cas, j'avais nommé mon ESP32 *BaseEsp32* pour l'OTA et le Wi-Fi.

Le 23 est le port choisi.  
  
Veuillez trouver ci-joint une vidéo (nommé Mise en place du monitoring dans le terminal) montrant la démarche pour téléverser le programme et accéder au terminal, comme si vous étiez en liaison série.

À la fin de la vidéo, mes gestes ne sont pas visibles, mais j'appuie sur la touche flèche haut jusqu'à retrouver ma commande *telnet*, puis je clique sur Entrée pour exécuter le programme.