Utilisation de Preferences.h avec ESP32 (Arduino)

Introduction

La bibliothèque `Preferences.h` permet de stocker des données de manière persistante dans la mémoire flash de l'ESP32. Contrairement aux variables classiques stockées en RAM, les données enregistrées avec `Preferences` sont conservées après un redémarrage ou une coupure d'alimentation.

Elle repose sur le système interne NVS (Non-Volatile Storage) de l'ESP32 et offre une interface simple pour manipuler des paires clé-valeur.

Principe de fonctionnement

Les données sont organisées sous forme de :

- **namespace** (espace de stockage, comparable à un dossier)
- **clés** associées à des valeurs

Chaque namespace contient plusieurs variables identifiées par une clé unique.

Utilisation de base

1. Inclusion de la bibliothèque

```
#include <Preferences.h>
```

2. Déclaration de l'objet

```
Preferences preferences;
```

3. Ouverture d’un namespace

```
preferences.begin("mon-espace", false);
```

- `"mon-espace"` : nom du namespace
- `false` : mode lecture/écriture
- `true` : mode lecture seule

4. Écriture de données

```
preferences.putInt("compteur", 42);
preferences.putString("nom", "ESP32");
preferences.putBool("etat", true);
```

Types supportés :

- entiers (int, uint)
- flottants (float, double)
- booléens
- chaînes de caractères (String)
- tableaux de bytes

5. Lecture de données

```
int compteur = preferences.getInt("compteur", 0);
String nom = preferences.getString("nom", "inconnu");
bool etat = preferences.getBool("etat", false);
```

Le second paramètre correspond à la valeur par défaut si la clé n’existe pas.

6. Fermeture

```
preferences.end();
```

Cette étape libère les ressources associées.

Exemple complet

```
#include <Preferences.h>

Preferences preferences;

void setup() {
  Serial.begin(115200);

  preferences.begin("config", false);

  int bootCount = preferences.getInt("boot", 0);
  bootCount++;

  preferences.putInt("boot", bootCount);

  Serial.println(bootCount);

  preferences.end();
}

void loop() {}
```

Dans cet exemple, une valeur est stockée en mémoire flash et incrémentée à chaque redémarrage.

Opérations supplémentaires

Supprimer une clé

```
preferences.remove("cle");
```

Effacer tout un namespace

```
preferences.clear();
```

Cas d’utilisation

- Stockage de paramètres (WiFi, configuration utilisateur)
- Compteurs persistants
- Calibration de capteurs
- Sauvegarde d’état (ex : relais, LED)

Limitations et bonnes pratiques

- La mémoire flash a un nombre limité de cycles d’écriture
  → éviter les écritures trop fréquentes dans une boucle
- Les clés doivent être uniques dans un namespace
- Les données écrites avec une clé existante écrasent les précédentes

Conclusion

`Preferences.h` fournit une méthode simple et efficace pour gérer des données persistantes sur ESP32. Elle évite de manipuler directement la mémoire flash tout en offrant une interface claire et adaptée à la majorité des besoins embarqués.

Revision #2
Created 2026-03-28 15:17:56 UTC by Axolito
Updated 2026-03-28 15:24:19 UTC by Axolito