Communication avec l'API de TimeTonic

Pour permettre au M5Stack de communiquer avec l’API de TimeTonic, il faut exploiter les capacités réseau du M5Stack (via l’ESP32), utiliser les librairies WiFi et HTTPClient, et structurer les requêtes selon les spécifications de l’API TimeTonic.

---

1. Préparer l’accès à l’API TimeTonic

L’API TimeTonic permet d’accéder, de lire et de modifier les données de vos espaces de travail à distance. Pour chaque requête, il est nécessaire de transmettre un **sesskey** (clé d’API/session), associé à votre compte utilisateur. Cette clé s’obtient via l’interface TimeTonic et doit être gardée confidentielle (elle ne sera donc publiée ici, si vous copiez l'exemple de code, pensez bien à mettre la votre).

Il est aussi important de repérer dès maintenant où trouver les différents codes permettant d'identifier les books, tables, lignes et colonnes dans TimeTonic :

Book ID : nom de l'espace de travail (attention, si celui-ci a été renommé, il gardera son 1er nom en ID, en cas de doute vérifier dans les paramètres du book dans la catégorie workspace informations).

Table ID : code à 6 chiffres accessible en cliquant sur les 3 points verticaux à droite de la fenêtre lorsque vous visualiser la table, puis en cliquant sur organize fields. L'ID sera entre parenthèse en haut de la fenêtre qui apparait.

Field ID : code à 6 chiffres accessible dans la même fenêtre que pour Table ID, sous la catégorie éponyme.

Row ID : correspond simplement au numéro de la ligne. Lorsqu'on voudra en créer une nouvelle en fin de tableau, on utilisera le code `tmpNEW_ROW`.

2. Interaction avec l'API

Lors du développement, il existe plusieurs manière d'interagir avec l'API de TimeTonic :

- Sur le site de TimeTonic : chaque modification de vue, de champ etc... est en fait une requête d'API faite de manière transparente.
- Avec l'outil Postman : cet outil permet de construire des requêtes et de voir en direct la réponse du serveur. C'est l'outil privilégié pour tester et construire des requêtes complexes avant de les implémenter dans le code. Il permet aussi de diagnostiquer et résoudre les problèmes lorsqu'on ne récupère pas les données désirées.
- Dans le code du M5 : avec la librairie HTTPClient, et une requête POST passée sous forme de String. Attention, il est nécessaire de formatter la requête correctement pour qu'elle soit interprêtée comme il faut (avec notamment des \" pour qu'ils n'interagissent pas avec les bornes de la String).

2. Exemple de requête : getAllBooks

La méthode **getAllBooks** permet de récupérer la liste de tous les « books » (espaces de travail) auxquels l'utilisateur a accès. Pour connaitre l'étendu des possibilité de l'API de TimeTonic, je vous renvois vers la documentation de celui-ci. La requête doit être envoyée en POST à l'URL suivante :

`https://timetonic.com/live/api.php`

Les paramètres à inclure sont :

- `version` : version de l'API (ex : "6.20b") (optionnel)
- `req` : "getAllBooks"
- `o_u` : identifiant utilisateur (doit correspondre à celui du sesskey)
- `u_c` : identifiant utilisateur (idem)
- `sesskey` : votre clé d'API/session, qui peut être créée dans les paramètres de votre compte TimeTonic

3. Implémentation sur M5Stack (Arduino)

Voici comment structurer le code avec les librairies **WiFi**, **HTTPClient** et **ArduinoJson** :

```
#include <WiFi.h>
#include <HTTPClient.h>
#include <ArduinoJson.h>

const char* ssid = "fablabstaff";
const char* password = "xxxxxxxxx"; //Pensez à remplacer le SSID et le mot de passe par vos paramètres WiFi

HTTPClient http;

void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("Connecte au WiFi");

  http.begin("https://timetonic.com/live/api.php");
  String postData = "req=getAllBooks"
                    "&o_u=USER"// Remplacez bien USER par votre nom d'utilisateur TimeTonic
                    "&u_c=USER"
                    "&sesskey=xxxxxxxxxxxxxxxxxxxxxxxxxxx"; // Remplacez par votre sesskey
  int httpResponseCode = http.POST(postData);

  if (httpResponseCode > 0) {
    String reponse = http.getString();
    Serial.println("Réponse API :");
    Serial.println(reponse);

    // Traitement JSON (exemple)
    DynamicJsonDocument doc(4096);
```

```
    deserializeJson(doc, reponse);
    if (doc["status"] == "ok") {
      JsonArray books = doc["allBooks"];
      for (JsonObject book : books) {
        Serial.println(book["b_c"].as<String>()); // Affiche le code du book
      }
    }
  }
  else {
    Serial.print("Erreur HTTP : ");
    Serial.println(httpResponseCode);
  }
  http.end();
}



void loop() {
  // Rien ici pour cet exemple
}
```

Ce code permet de récupérer les informatiosn de tous les espaces de travail présents dans un compte TimeTonic.

4. Points importants

- **Connexion sécurisée** : L'API TimeTonic nécessite une connexion HTTPS, ce qui est supporté par l'ESP32.
- **Format POST** : Les paramètres doivent être envoyés en POST, au format `application/x-www-form-urlencoded` 23.
- **Traitement de la réponse** : La réponse de l'API est en JSON, d'où l'utilisation de la librairie ArduinoJson pour extraire les informations utiles.
- **Gestion des erreurs** : Toujours vérifier le code de retour HTTP et le champ `status` dans la réponse JSON.

---

5. Résumé

Pour communiquer avec l’API de TimeTonic depuis un M5Stack, il faut :

- Se connecter au WiFi,
- Construire une requête POST avec les bons paramètres,
- Envoyer la requête à l’API,
- Traiter la réponse JSON pour exploiter les données reçues.

Ce principe peut être adapté à toutes les autres méthodes de l’API TimeTonic, en changeant simplement le paramètre `req` et les paramètres associés.

Revision #4
Created 19 June 2025 09:45:22 by Eyglier De Rooster Mathieu
Updated 21 July 2025 14:20:01 by Eyglier De Rooster Mathieu