Опис команд управління таблом

# Загальні відомості

Управління таблом організоване на базі API-серверу, створеного на базі FastAPI - сучасного вебфреймворку для Python.

В контексті роботи даного технічного виробу, API-сервер дозволяє:

* надсилати текстові дані для виводу на табло у вигляді параметризованого веб-запиту з програмного середовища Python;
* отримувати дані про поточні параметри роботи табло у форматі json (зворотний зв’язок із керуючою програмою);
* здійснювати контроль помилок запиту та валідацію даних, що надсилаються.

# Необхідні програмні компоненти і умови для здійснення управління

1. Середовище Python, версії 3.7 і вище;
2. Інстальований і імпортований в програмний застосунок управління модуль **requests**;
3. Мережева доступність IP-адреси і порту API-сервера з місця хостингу програмного застосунку управління таблом.

# Опис структури запиту

API-сервер табла підтримує наступні типи веб-запитів:

* GET-запити - для отримання поточних параметрів роботи табла, або результатів виконання запитів (зворотний зв’язок);
* PUT-запит - для оновлення текстових даних, що виводяться на табло.

## Варіанти GET-запитів

Існує 2 опції GET запиту: *запит повного статусу параметрів табла* та *запит статусу окремої зони табла.*

Отримана відповідь надходитиме у вигляді структурованих даних json.

### Запит повного статусу табла:

requests**.get**(http://**<ip>**:**<port>**/).json()

де:

<ip> - IPv4 адреса за якою доступний API-сервер;

<port> - значення порту, на якому працює API (за замовчуванням: *8000*);

Приклад виконання:

>>> requests.get("<http://192.168.111.106:8000/>").json()

{'dataset': {'0': {'text': ' ', 'brt': 4, 'id': 0, 'addr': 8}, '1': {'text': '', 'brt': 127, 'id': 1, 'addr': 9}, '2': {'text': '', 'brt': 127, 'id': 2, 'addr': 10}, '3': {'text': '', 'brt': 127, 'id': 3, 'addr': 11}, '4': {'text': '', 'brt': 127, 'id': 4, 'addr': 12}}}

де:

'id' - ідентифікатор зони табла;

'text' - поточне значення тексту, що виводиться;

'brt' - поточне значення яскравості (4-255);

'addr' - адреса мікроконтролера зони на внутрішній шині обміну даними i2c.

Запит статусу окремої зони табла:

requests**.get**(http://**<ip>**:**<port>**/zone/**0**).json()

де:

<ip> - IPv4 адреса за якою доступний API-сервер;

<port> - значення порту, на якому працює API (за замовчуванням: *8000*);

0 - ідентифікатор зони табла: *0* - верхній ряд, *1* - другий згори, …,*4* - нижній;

Приклад виконання:

>>> requests.get(["http://192.168.111.106:8000/zone/0"](http://192.168.111.106:8000/zone/0)).json()

{'text': 'HELLO!', 'brt': 4, 'id': 0, 'addr': 8}

Використання PUT-запитів. Негайний вивід тексту.

Для того, щоб негайно вивести певний текст у певну зону табла, програма управління має надати в запиті наступні дані:

* ідентифікатор зони табла *(0 (верхній рядок) | 1 | 2 | 3 | 4 (нижній рядок))*;
* текст даних для виводу на табло, згідно вимог, що будуть описані далі.

Структура запиту:

requests**.put**("http://**<ip>**:**<port>**/update/**0**?text=**SOME\_TEXT**").json()

де:

<ip> - IPv4 адреса за якою доступний API-сервер;

<port> - значення порту, на якому працює API (за замовчуванням: *8000*);

0 - ідентифікатор зони табла;

SOME\_TEXT - текст для відображення у зоні табло зі вказаним попередньо ідентифікатором

.json() - необов’язковий елемент, що необхідний для отримання статусу виконання запиту (зворотний зв’язок).

Приклади виконання:

>>>requests.put("http://192.168.111.106:8000/update/0?text=HELLO!")

<Response [200]>

>>>requests.put("http://192.168.111.106:8000/update/0?text=HELLO!").json()

{'updated': {'text': 'HELLO!', 'brt': 4, 'id': 0, 'addr': 8}}

Результат:



Відкладений вивід тексту. Примусова яскравість.

Існує можливість записати у пам’ять табла дані, без негайного виводу записаного тексту. Це може знадобитися для одночасного виводу текстової групи, що займає всі зони табла, а також для виводу тексту з заданим значенням яскравості (замість встановленого автоматично за датчиком освітленості).

### Запис даних у табло без негайного виводу

Для запису даних в пам’ять табла використовується наступний запит:

requests**.put**("http://**<ip>**:**<port>**/upload/**0**?text=**SOME\_TEXT**&brt=**<brt\_value>**").json()

де:

0 - ідентифікатор зони табла;

SOME\_TEXT - текст для відображення у зоні табло зі вказаним попередньо ідентифікатором

&brt=**<brt\_value>** - обов’язковий параметр, що примусово встановлює значення яскравості *(*<brt\_value> *= 0-255*) для тексту в зоні..   
<brt\_value> мусить бути цілим числом в діапазоні від 0 до 255.

.json() - необов’язковий елемент, що необхідний для отримання зворотного зв’язку з інформацією про статус виконання запиту.

Ознакою успішного запису даних у пам’ять є відповідь виду:

{**'new data uploaded':** {'text': '**BLA-BLA-BLA**', 'brt': **127**, 'id': 0, 'addr': 8}}

Текст, що поточно демонструється у вказаній зоні табло, при цьому, не зміниться.

Увага! Новий запит типу *update* оновлює попередньо записані параметри зони.

### Вивід даних з пам’яті на табло

Для відображення на табло даних з пам’яті застосовується наступний запит:

requests**.put**("http://**<ip>**:**<port>**/display/**0**").json()

де:

<ip> - IPv4 адреса за якою доступний API-сервер;

<port> - значення порту, на якому працює API (за замовчуванням: *8000*);

0 - ідентифікатор зони табла;

Текст, що міститься в пам’яті для зони із вказаним ідентифікатором буде негайно виведений на табло. Яскравість тексту, що виводиться, буде відповідати попередньо записаному значенню поля brt. Таким чином можна виділити текст певної зони на фоні інших. Ознакою успішного відпрацювання запиту є відповідь виду:

{'**the data is now being displayed**': {'text': **BLA-BLA-BLA**', 'brt': **127**, 'id': 0, 'addr': 8}}

Таким чином, запити *upload* та *display* доцільно використовувати в парі, один за одним.

Вимоги до тексту, що надсилається на табло

Табло наразі не підтримує символи з адресацією поза межами ASCII.   
Тим не менш, пристрій спроможний відображати усі літери кириличної абетки. Щоправда, через це вивід літер має певні особливості і існують певні вимоги до тексту:

1. В обох абетках доступний вивід тільки символів верхнього регістру;
2. Вивід символів латинської абетки виконується символами верхнього регістру латинки. Символи нижнього регістру відсутні;
3. Символи, що спільні для обох абеток задаються символами верхнього регістру латинської абетки;
4. Кириличні символи, яких немає в латинці закодовані на символах нижнього регістру латинської розкладки із наступними відповідностями:

| **Cyrillic** | **Latin** | **Cyrillic** | **Latin** | **Cyrillic** | **Latin** |
| --- | --- | --- | --- | --- | --- |
| Б | b | И | y | Щ | w |
| Г | g | Л | l | Й | j |
| Д | d | П | p | Ю | o |
| Є | e | Ф | f | Я | q |
| Ж | h | Ц | c |  |  |
| З | z | Ч | t |  |  |
| Ї | r | Ш | s |  |  |

1. Текст мусить займати не більше 10 знакомісць на 1 зону;
2. Для очистки зони достатньо надіслати текст, що складається лише з “пробілу”;

Приклад виводу тексту кирилицею:

>>>requests.put("http://192.168.111.106:8000/update/0?text=bABOBHqHyj").json()

{'updated': {'text': 'bABOBHqHyj', 'brt': 4, 'id': 0, 'addr': 8}}

Результат:

