取扱説明書： VU データ HTTP サーバー

Pal8000.83001@gmail.com

1. はじめに

Raspberry pi 上の Volumio2 で音楽サーバーを構築している方も多数おられることと存じます。
利用する上で、VU メーター表示が欲しいというニーズはある、というか、自分は欲しいです。
追加ハードウェアなしで実現するために、拙作「VUMeter for Windows」を改造しました。
HTTP 通信でリアルタイムに音量情報を受信し、メーター表示に反映させることができます。
本書では、このソフトに組み合わせる Raspberry pi 側環境である、VU データ HTTP サーバー（以下、VUServer）のインストール、運用を中心に記載します。

LAN でやり取りをしているため、動作環境により、リアルタイム処理が間に合わない部分が出てくるかも知れませんが、現在 Volumio2 を Windows から使っている環境の方には、追加するハードウェアがなく、手軽にお試し頂けます。

本書と付属するソフトウェアは予告なく更新される場合があります。

1-1. 無保証

趣味の範囲では充分に実用的ながら、HTTP 通信に頼るため、録音モニター等で使うには不安があります。
本書を記述後、開発機とは別のマシンで内容に沿って最初から導入し、クライアントと組み合わせた動作が再現することを確認しておりますが、用途に関わらず、ご自身の責任で導入、ご利用ください。
運用した結果については何らの責任も負いません。

1-2. 参考文献

MPD の再生音量をリアルタイムに取得する

MPD, FIFO, Python, Audioop, Arduino, and Voltmeter: "Faking" a VU Meter Stack overflow

1-3. 本書の対象者

Volumio2 を既にインストールして動作させている方を想定して記載しています。
初心者寄りを意識しましたので、くどい記述がある一方、本当の初心者には必要なことが
欠けていることもあり得ます。ご了承ください。

本書で意図的に記述を省いた部分に以下の様なものがあります。
情報は簡単に見つかるので、判らない部分は Web.や文献等で補強してください。

- SD カードのバックアップ・リストア（Win32DiskImager 等）
- Volumio2 自体のインストール、運用（Volumio）
- ターミナルソフトで Raspberry pi にログインする方法（TeraTerm 等）
- 圧縮ファイルの解凍方法（tar）
- Linux テキストエディタの利用方法（vi、nano 等）
- Windows から Raspberry pi にファイル転送する方法（WinSCP , FFFTP 等）
- Systemd のその他のコマンド

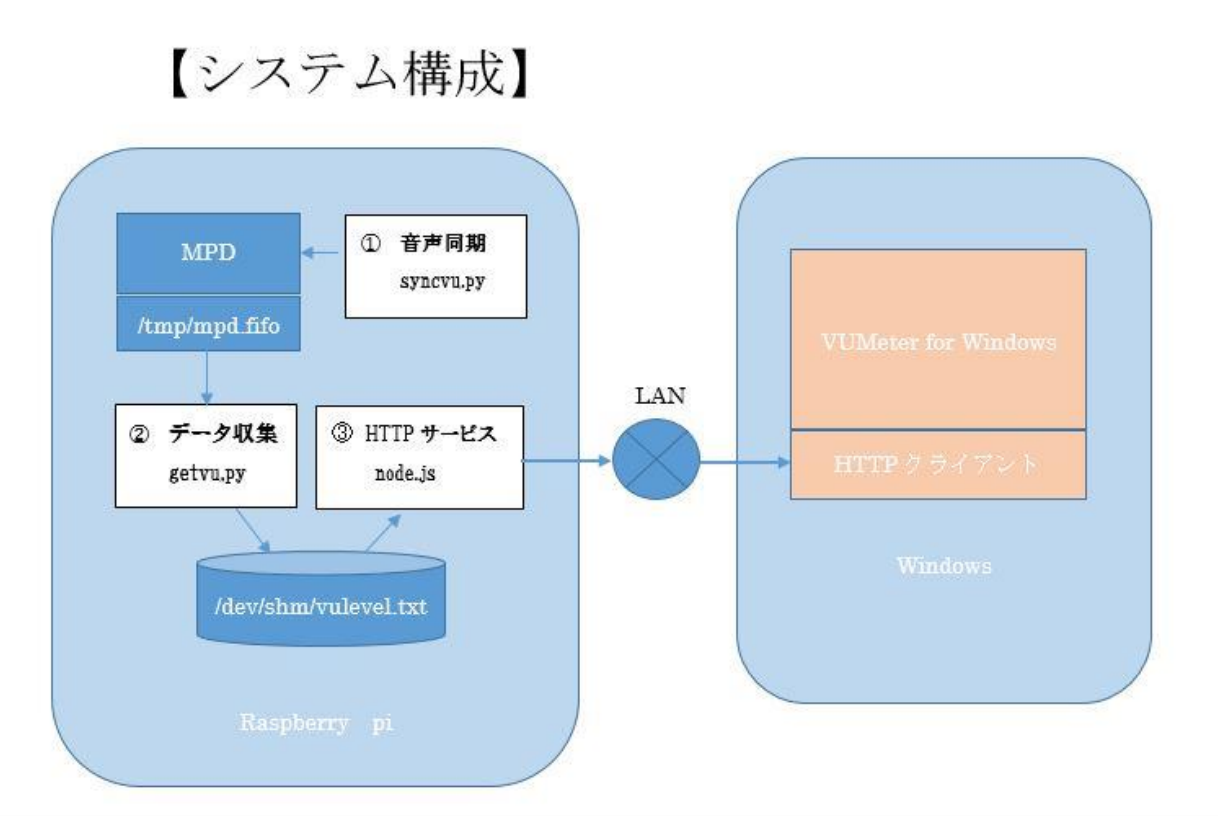
1-4. サポート情報のご提供依頼と利用条件

本ソフトウェアで利用している技術として、MPD、Python、node.js、Systemd 等を
利用しましたが、作者自身 Linux 自体の知識も含め、精通しているとは言い難いです。
やりたいことを調べ、使えるものを使ってまとめたプロジェクトなので、行儀の悪い部分や、
不具合、他にもっと良いやり方があるかも知れません。

サポート情報のご提供を歓迎します。
お気付きの点がありましたら、お手数でも文頭に記載したメールアドレスまでお知らせください。
ソフトウェアやドキュメントをバージョンアップする際に、参考とさせて頂きます。
頂いたサポート情報は、将来的に本ソフトや「VUMeter for Windows」を改良する上で参考にし、
有用な情報を配布パッケージに利用した場合、解説書等にお名前を明記させて頂きます。
この際、無償での利用を認めて頂いたものとします。
但し、ライセンスによる制限があるものに関しては、情報ご提供時に明記してください。
その定めに従い、利用するかどうかを判断させて頂きます。

2. 機能概要

Volumio2 では、音楽再生に MPD(Music Player Daemon)を利用しています。
MPD からリアルタイムの再生音量を取得することができるので、常時取得し、
HTTP 送信サーバーをたてることで Windows 側に音量レベルを通知する機能を実現します。



Raspberry pi (サーバー)
① Python による常時ループ処理により、再生音声とデータ収集処理の同期処理
② Python による常時ループ処理により、MPD の音量データを仮想ドライブにファイル化
③ node.js で HTTP サーバーをたて、リクエスト時点の最新データをクライアントに渡す

Windows(クライアント)
VUMeter for Windows に組み込まれた HTTP クライアントで、HTTP 通信により
取得した音量データを利用して、メーター表示する

本書では Volumio2 を前面にたてていますが、構成上、MPD からの音量データを取得して利用する
仕組みのため、Raspbian OS や、他の Linux ディストリビューションでも動作するはずです。
但し、3 項で示す構成要素でしか試行していないため、OS 自体、Python や node.js 等の
バージョン違いにより、そのままでは動作しない可能性があります。
サーバー側のソースはすべて本パッケージに付属しますので、それぞれの環境への適応は
ご利用者にてお試しください。

3. 構成要素(必要なハード、ソフト）

3-1. サーバー側に必要なハード、ソフト

*3-1-1.*Raspberry pi 本体

小型のボードコンピュータです。
本記事の開発と検証は、Raspberry pi 2B で行いました。
また、Raspberry pi 3B にてインストール手順の検証を行い、再現性を確認しました。
これ以外のモデルでは検証していません。

*3-1-*2.Volumio2

Raspberry pi に OS 込みで音楽サーバーをインストールするソフトウェアパッケージです。
利用するソフトウェアに Python2.7 と node.js 8.1.1 が標準で入っており、そのまま利用するものとします。
版数は、開発版は **2.513** (07-12-2018)をインストールし、**2.526** (12-01-2018)まで更新したもので動作確認しました。
新規インストールによる再現性検証は、**2.526** (12-01-2018)で行いました。
その後の **2.673**(12-1-2019)更新でも動作しています。

*3-1-*3.MPD クライアントライブラリ

python-mpd は MPD クライアントの機能を Python から呼び出すライブラリです。
ソフトウェア導入に先立ち、インストールが必要です。

```
# sudo apt-get install python-mpd
```

ワンポイント

apt-get で新しいソフトを導入する場合、前もって、update を
行っておいてください。

```
# sudo apt-get update
```

いきなり install しようとすると E: Unable to locate package python-mpd のエラーが出ます。

*3-1-*4.テキストエディタ

Volumio2 のディストリビューションではテキストエディタの vi がすぐに使えません。
今回は、vi 互換の vim を追加でインストールしました。
vim 導入

```
# sudo apt-get install vim
```

導入時、テキストエディタが必須なのは１ヶ所のみです。
vi に限らず、使い慣れたものがあれば、そちらをご利用ください。

*3-1-4.*Raspberry pi 側ソースコード

必要なファイルを一覧します。

各ファイルは同梱した vuserverx_x_x.tar.gz に圧縮して入っています。

vuserver1_1_0.tar.gz　：1.1.0 版

解凍して /home/volumio ディレクトリ直下にあるものとして進めます。

ファイル名	変更/新規	摘要
addmpdconf.txt	—	/etc/mpd.conf に追記するテキストです
/etc/mpd.conf	変更	MPD からの音量出力を受け取る定義を追記します
syncvu.py	新規	常時起動して、データ収集用 FIFO を再生データに同期させます
getvu.py	新規	常時起動して、リアルタイムに MPD の音量を取得します
node.js	新規	取得した音量データを配信する Web サーバーです
vuserver.js	新規	Syncvu.py, getvu,py,node.js の 3 本を起動します
vuserver.service	新規	ブート時に自動起動させるため、Systemd に渡すスクリプトです

3-2.　サーバー側その他情報

3-2-1.SSH 設定

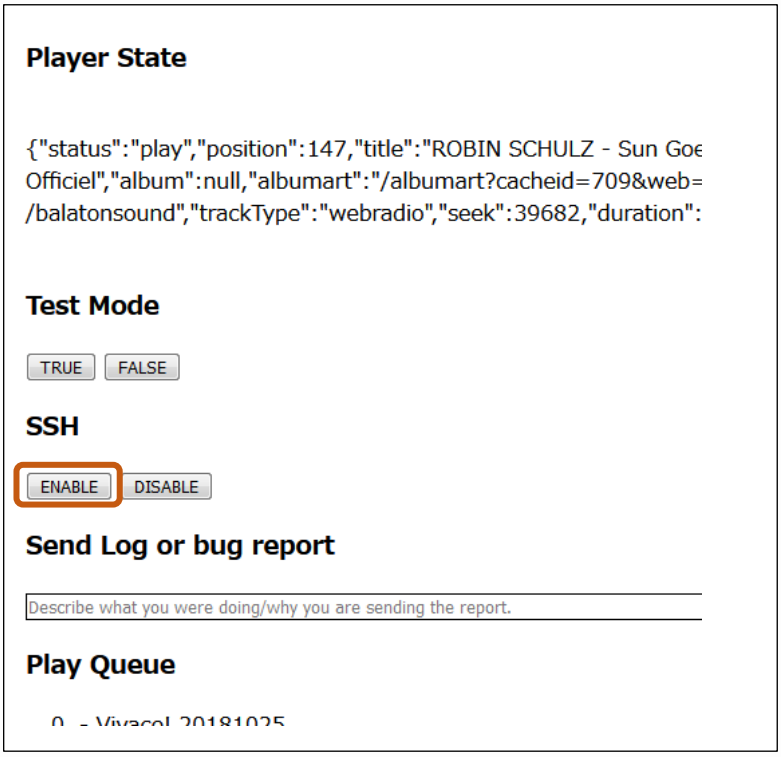
サーバー側のインストール作業には、外部からターミナルソフト(Tera Term 等)により SSH でログインする必要があります。

Raspberry pi 側の準備として、以下の様に SSH 通信を有効にします。

まず、Volumio2 を導入後、ブラウザでメイン画面を表示します。

URL の最後、/playback の個所を、/dev に変更して再度読み込みます。

以下の様な画面に表示が変わるので、SSH の項目で[ENABLE]を押してください。



これで、SSH 通信ができる様になります。

3-2-2. バックアップ、アンインストール

インストール作業自体はさほど難しくないと思いますが、心配な方は、SD カードを Windows 側でバックアップしておけば、最悪の場合でもインストール作業直前の状態まで戻すことができます。
アンインストールは、インストール順序を逆にたどり、元に戻して行きます。
実行中に作成されるファイルは次期起動時に消滅します。
アンインストールもバックアップからの復元なら、確実にその時点まで遡れます。

3-2-3. OS のシステム更新、設定変更

Volumio2 側の設定で、再生デバイスを変更すると mpd.conf ファイルが初期化されることが判っています。テキストエディタで設定をやり直してください。
インストール後の apt-get update や、Volumio2 のメニューからのシステム更新はこれまでの所問題ありませんでした。
ファクトリーリセットを行うと Volumio2 初期導入時と同等ですので、インストール作業のやり直しになります。
将来的に、Volumio が大幅にアップグレードした場合や MPD の仕様が変わった場合はこの限りではありません。

3-3. クライアント側要素

3-3-1.Windows PC

Windows7(SP1)以降の OS が載った PC
クライアント側ソフトが、Microsoft .Net Framework4.72 を利用します。
Volumio2 をブラウザからアクセスできることを確認しておいてください。

3-3-2.クライアント側ソフト (VUMeter for Windows)

Vector サイトからインストールのための Zip ファイルをダウンロードしてください。
https://www.vector.co.jp/soft/dl/winnt/art/se494573.html

※3.0.0.0 より前の版では、今回の機能は動作しません。

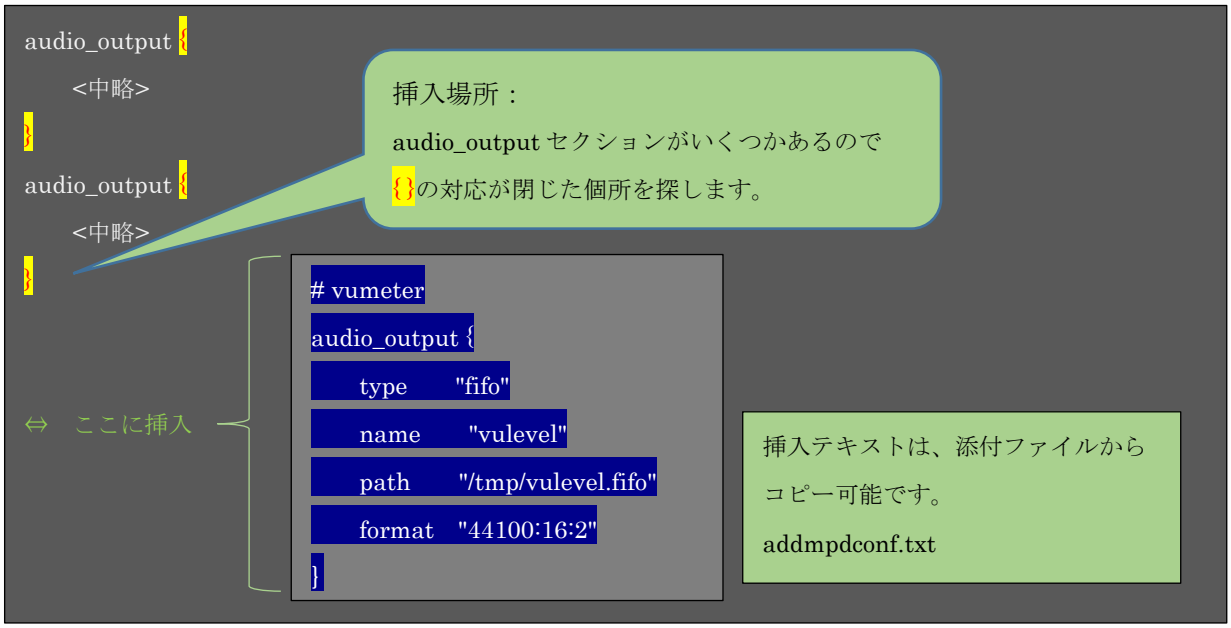
4. Raspberry pi 側インストール

本項はターミナルでの操作になります。volumio ユーザーでログインして進めてください。

4-1. リアルタイム音量情報の取得先を定義

MPD Configuration File を修正します。

/etc/mpd.conf をテキストエディタで開け、以下の行を挿入します。



本項目で登録した tmp/vulevel.fifo を利用開始するため、変更後は一旦**再起動**してください。

ワンポイント

本設定は、Volumio2 の設定画面で、出力デバイスの項目を変更すると削除されてしまいます。

出力デバイスを設定変更した場合は、再度設定してください。

4-2. 各処理スクリプト

利用するスクリプトの配置は、以下の通りです。

起動はブート時の自動実行で行います。

ファイル名	言語	配置	起動
syncvu.py	python	/home/volumio	vuserver.js が起動します
getvu.py	python	/home/volumio	vuserver.js が起動します
node.js	Node.js	/home/volumio	vuserver.js が起動します
vuserver.js	Node.js	/home/volumio	vuserver.service の記述により起動し、配下の 3 つのソフトを起動します
vuserver.service	Service スクリプト	/etc/systemd/system	ブート時に Systemd が本ファイルを参照して vuserver.js を自動起動します 登録方法は 4-3 項で説明します。

スクリプト個別の機能やマニュアル起動方法は、追補編に記載しています。

4-3. Systemd サービス登録

vuserver.service は、Raspberry pi の電源投入時に VUServer を自動起動させる条件を
記述したファイルです。
/etc/systemd/system ディレクトリに置いて、自動起動できる様に登録する操作が必要です。

以下、4-3-1 項から 4.3.6 項まで、サービス登録から自動起動の設定、起動に至る部分を実行ログで示します。
プロンプト(volumio@volumio2-pi2b:~$)のある行の オレンジ色 部分が必須入力です。
確認目的だけで、設定とは直接関係のない入力行があります。（ 緑色 部分）
動作しているかチェックする目的を踏まえ、面倒でも双方入力することをお奨めします。

4-3-1.vuserver.service のコピー、オーナー変更、実行属性付加

vuserver.service ファイルを/etc/systemd/system にコピー、オーナー変更、実行属性の変更

```
volumio@volumio2-pi2b:~$ sudo cp vuserver.service /etc/systemd/system
volumio@volumio2-pi2b:~$ sudo chown root:root /etc/systemd/system/vuserver.service
volumio@volumio2-pi2b:~$ sudo chmod 644 /etc/systemd/system/vuserver.service
volumio@volumio2-pi2b:~$ ls -l /etc/systemd/system/vuserver.service
-rw-r--r-- 1 root root 226  1 月 20 02:21 /etc/systemd/system/vuserver.service
```

4-3-2.再読込で設定ファイルを登録

続けて、Systemd に登録します。

```
volumio@volumio2-pi2b:~$ sudo systemctl daemon-reload
volumio@volumio2-pi2b:~$ sudo systemctl status vuserver.service
● vuserver.service - VUMeterWebServer
   Loaded: loaded (/etc/systemd/system/vuserver.service; disabled)
   Active: inactive (dead)
```

4-3-3.自動起動指定

自動起動を設定します。

```
volumio@volumio2-pi2b:~$ sudo systemctl enable vuserver.service
Created symlink from /etc/systemd/system/multi-user.target.wants/vuserver.service to
/etc/systemd/system/vuserver.service.
```

4-3-4.開始前の状態確認

本項は任意ですが、開始要求前の状態を見ます。サービスが enabled になっているのがわかります。

```
volumio@volumio2-pi2b:~$ sudo systemctl status vuserver.service
● vuserver.service - VUMeterWebServer
   Loaded: loaded (/etc/systemd/system/vuserver.service; enabled)
   Active: inactive (dead)
```

4-3-5.マニュアル開始要求

本項も任意ですが、マニュアルで開始要求を出すことで、4-3-6 項により、正常に動作するか
再起動前に確認できます。

```
volumio@volumio2-pi2b:~$ sudo systemctl start vuserver.service
```

マニュアル起動がうまく行けば、以降はリブートする度に自動起動します。

4-3-6.状態確認

サービスにより起動されているモジュールが確認できます。

確認ポイント

・ステータスが active (running)になっている
・CGroup に各スクリプトが動作している

以下は、ブート後にうまく起動している例です。

```
volumio@volumio2-pi2b:~$ sudo systemctl status vuserver.service
● vuserver.service - VUMeterWebServer
   Loaded: loaded (/etc/systemd/system/vuserver.service; enabled)
   Active: active (running) since 日 2019-02-17 02:57:23 JST; 1min 14s ago
 Main PID: 544 (node)
   CGroup: /system.slice/vuserver.service
           544 /bin/node /home/volumio/vuserver.js
           729 /bin/sh -c python getvu.py ./
           730 python getvu.py ./
           731 /bin/sh -c python syncvu.py ./
           732 /bin/sh -c node node.js ./
           733 python syncvu.py ./
           734 node node.js ./
 2月 17 02:57:23 volumio2-pi2b systemd[1]: Starting VUMeterWebServer...
 2月 17 02:57:24 volumio2-pi2b systemd[1]: Started VUMeterWebServer.
volumio@volumio2-pi2b:~$
```

5. Windows 側ソフトウェア導入と設定

Windows 上で、「VUMeter for Windows」の圧縮ファイルを解凍後、setup.exe の実行によりインストールします。
詳細情報はソフトウェア同梱の「VUMeterforWindows 取扱説明書.pdf」を参照してください。

ソフトウェア起動後に Construction 画面からサーバー接続確認を行います。

5-1.サーバー接続設定

VU メーターパネル(メイン画面)の右ボタンメニューから Construction を選択します。
表示された Construction 画面の Option タブページ内にある、以下の項目を適切に設定します。

- Signal Source
- Server URL
- Delay
- Sample Count

個別項目の詳細は VUMeter for Windows の取扱説明書をご参照ください。

5-2.動作確認のポイント

サーバー側の VUServer が動作していないと、ポップアップで警告が表示されます。
また、針が振れるのを確認するため、Volumio2 でコンテンツを再生したままにしておいてください。

5-3.接続設定情報を残す

ソフトウェアの仕様上、ビルトインされた 3 枚のメーターパネルでは、一旦ソフトを止めて再起動すると、サーバー接続設定情報が初期化され、消えてしまいます。
設定を残して次回起動時に使うには、ユーザー定義パネルの構築が必要になります。
Construction 画面で、付属の画像ファイルを使い、簡単にユーザー定義パネルを作れます。
詳細については VUMeter for Windows の取扱説明書をご参照ください。

5-4.運用上の注意

正しく設定したあとの実運用でも、サーバーを電源断するなど、LAN 回線が不通になると通信が再開されない限り、繰り返しポップアップで警告が表示されます。
その際は、Signal Source を内部信号に切り替えてください。

6. アンインストール

インストール時と逆の手順になります。
Systemd のサービスを削除し、/home/volumio に解凍したファイルを削除します。

6-1.サービス停止～スクリプト消去

プロンプト(volumio@volumio:~$)のある行の　オレンジ色　部分が必須入力です。
確認目的だけで、設定とは直接関係のない入力行は　緑色　部分です。

```
volumio@volumio:~$ systemctl stop vuserver
volumio@volumio:~$ systemctl disable vuserver
Removed symlink /etc/systemd/system/multi-user.target.wants/vuserver.service.
volumio@volumio:~$ sudo rm /etc/systemd/system/vuserver.service
volumio@volumio:~$ sudo systemctl daemon-reload
```

6-2.MPD リアルタイム音量情報の取得先削除

/etc/mpd.conf をテキストエディタで開け、インストール時に追加した以下の行を**削除**します。

```
# vumeter
audio_output {
    type      "fifo"
    name      "vulevel"
    path      "/tmp/vulevel.fifo"
    format    "44100:16:2"
}
```

6-3.再起動

再起動することで、取得先削除が有効になります。

```
volumio@volumio:~$ Reboot
```

再起動後、サービス登録が消えている状態を確認できます。

```
volumio@volumio:~$ sudo systemctl status vuserver
● vuserver.service
   Loaded: not-found (Reason: No such file or directory)
   Active: inactive (dead)
```

6-4.スクリプト削除

再度 VUServer を利用する可能性があるのなら、本項目は実行する必要はありません。
以下の例では、/home/volumio ディレクトリ以下、**すべてのファイルを消去**しています。
ディレクトリ内に必要なファイルがある場合は、個別に消去してください。

```
volumio@volumio:~$ cd
volumio@volumio:~$ pwd
/home/volumio
volumio@volumio:~$ ls -l
total 31
-rw-r--r-- 1 root    root     121  1 月 26 06:28 addmpdconf.txt
-rw-r--r-- 1 volumio volumio 1818  2 月 12 13:10 getvu.py
-rw-r--r-- 1 volumio root     301  1 月 18 19:08 node.js
-rw-r--r-- 1 volumio volumio  519  2 月 12 12:44 syncvu.py
-rw-r--r-- 1 volumio volumio  426  2 月 12 12:48 vuserver.js
-rw-r--r-- 1 root    root     277  1 月 22 08:28 vuserver.service
-rw-r--r-- 1 root    root    1370  1 月 26 16:29 vuserver.tar.gz
volumio@volumio:~$ rm *
volumio@volumio:~$ ls -l
```

6. おわりに

もともと「VUMeter for Windows」を作り、各種の機能拡張をしてきましたが、OS から音量を取得する API 自体がディジタルのバー表示を想定したデータを出力するものでアナログメーターには不適切、という本質的な問題があり、正しい針の振れ方をしているとは言えませんでした。
さらに、WASAPI 排他モードを使った再生では OS 側の制限により音量が取得できなくなり音質向上を追求すると針が振れず、使えない、というジレンマもあります。
外部から音量データを取得する構想自体は 6 年位温めていたのですが、A/D コンバータに渡すアナログデータを作る回路で躓き、実現できていませんでした。
先日、ソフトだけで実現できそうな手段が見つかったので、急遽機能追加しました。

7. 余談

今回の目論見、HTTP 通信でデータのリアルタイム性がどの程度担保されるか、というのがテーマの 1 つでした。
最初に python でサーバープロセスを書いたら、秒 4 回しか通信できず、挫折しかけたのですが node.js で書き換えることで爆速となり、解決しました。
python にしても起動に時間がかかりますが、1 つのプロセスが起動後にループする構成では、結構な速度が出ます。

実は python も node.js も今回初めての挑戦でしたが、割と容易に書けました。
全体に開発期間も短く、クライアント側の方が時間をかけました。
殊に、node.js は、調べ初めて 30 分で書いた実用的なプログラムが一発で動いた、という人生初とも言うべき快挙でした。
たいしたことをしていないのも事実ですが、このコード量で動くのか、というのが正直な感想です。

その昔、ASCII 誌に掲載された N88-Basic インタプリタで実装した UNI+という冗談の様に遅い UNIX OS シミュレータを触りました。当時 UNIX 機の購入は雲の上でした。
Raspberry pi は小さくて安価なボードコンピューターですが、立派に Linux が動作します。

改訂履歴

2019/2/15 初版
2019/7/28 誤記訂正

2019/12/4 mpd.conf への記載変更、誤記訂正