エントリーポイントURL説明

概要

AssistantSeikaのHTTP機能を有効にすることで提供されるURLの説明です。

例えばAssistantSeikaが localhost:7180 で待ち受けしている状態なら、ブラウザに

- http://localhost:7180/VERSION を入力することでバージョン情報を得られます。
- http://localhost:7180/AVATOR2 を入力することで利用可能話者の情報を得られます。
- http://localhost:7180/AVATOR2/2000 を入力することでcid 2000の話者のパラメタ情報を得られます。

HTTPアクセスができるならブラウザ以外のプログラムからでも利用できます。

エントリーポイントURL

以下のエントリーポイントURLが利用可能です。

メソッド	URI	説明
GET	/VERSION	AssistantSeikaのバージョン文字列を返す。
GET	/AVATOR2	AssistantSeikaで利用可能な話者の一覧を返す
GET	/AVATOR2/{cid}	指定cidの話者のデフォルトパラメタ情報を返す 例: /AVATOR2/2000 → cid=2000の話者のデフォルトパラメタ情報を返す
GET	/AVATOR2/{cid}/current	指定cidの話者の現在のパラメタ情報を返す 例: /AVATOR2/2001/current → cid=2001の話者の現在のパラメタ情報を返す
GET	/app/{path}	指定pathの静的コンテンツを返す。 AssistantSeikaのHTTP機能設定タブの設定値 “ ドキュメントルートフォルダ ” で示すフォルダをドキュメントルートとして処理する。
POST	/PLAY2/{cid}	指定cidの話者に発声させる
POST	/PLAYASYNC2/{cid}	指定cidの話者に非同期発声させる
POST	/SAVE2/{cid}	指定cidの話者の音声データ(wav)を返す
POST	/SAVE2/{cid}/{sampleRate}	指定cidの話者の音声データ(wav)をサンプリングレート sampleRate でリサンプリングして返す
GET	/EFFECT/{cid}/clear	指定cidの話者のエフェクトパラメタ、感情パラメタをクリアする
GET	/EFFECT/{cid}/{param}/{value}	指定cidの話者のエフェクトパラメタparamに値valueを設定する
GET	/EMOTION/{cid}/{param}/{value}	指定cidの話者の感情パラメタparamに値valueを設定する
GET	/PLAY2/{cid}/{talktext}	指定cidの話者にテキストtalktextを発声させる
GET	/PLAYASYNC2/{cid}/{talktext}	指定cidの話者にテキストtalktextを非同期発声させる
GET	/SAVE2/{cid}/{talktext}	指定cidの話者にテキストtalktextを発生させた音声データ(wav)を返す
GET	/SAVE2/{cid}/{sampleRate}/{talktext}	指定cidの話者にテキストtalktextを発生させた音声データ(wav)をサンプリングレート sampleRate でリサンプリングして返す
POST	/STORE2	Bodyで指定するJSON形式のkey-valueペア情報を登録する。
POST	/STORE2/NEWDECLARE	Bodyで指定するJSON形式のkey-valueペア情報を登録する。すでに同じkeynameのkey-valueペア情報が登録されていればエラーになる。
GET	/STORE2/{keyname}	keynameで示すkey-valueペア情報をJSON形式で取得する。
GET	/STORE2/{keyname}/READFLUSH	keynameで示すkey-valueペア情報をJSON形式で取得する。正常に読み取れたら登録されているkeyname のkey-valueペア情報を消去する。

REST API風アクセス

/AVATOR2 で利用可能話者の一覧を取得できます。話者のcidを決めてください。
/AVATOR2/{cid} で話者cidに割り当てられているパラメタを取得できます。取得したeffectの内容がエフェクトパラメタ、emotionの内容が感情パラメタの一覧になります。変更したいパラメタを決めてくだ

さい。

/PLAY2/{cid} で音声発声□/SAVE2/{cid} で音声データ取得、が実行されます。
リクエスト時のBODYにJSON形式で発声させたいテキスト、変更したいパラメタを指定します。以下はJSON形式の例です。

```
{
  "talktext":"おはようございますー！",
  "effects":{
    "speed"      :1.0,
    "volume"     :1.0,
    "pitch"      :1.0,
    "intonation":1.0
  },
  "emotions":{
    "怒り"    :0.00,
    "喜び"    :1.00,
    "悲しみ" :0.20
  }
}
```

また、変更が必要なパラメタのみ渡せるので、以下の形式でも構いません。

```
{
  "talktext":"おはようございますー！"
}
```

以下は、上記JSON形式のデータをファイルにしてcurlコマンドからPOSTメソッドで送り発声させる例です。

このファイル x.json はUTF-8 BOM無しのJSON形式です。

x.json

```
{
  "talktext":"これは 組み込みのプロキシー経由で発声させた例です"
}
```

このx.jsonをcurlコマンドを使ってAssistantSeikaへ送り込みます。

```
curl -X POST -H "Content-Type: application/json" -d @x.json
http://SeikaServerUser:SeikaServerPassword@localhost:7180/PLAY2/2000
```

音声を 2000.wav の名前で保存する時はこうなります。

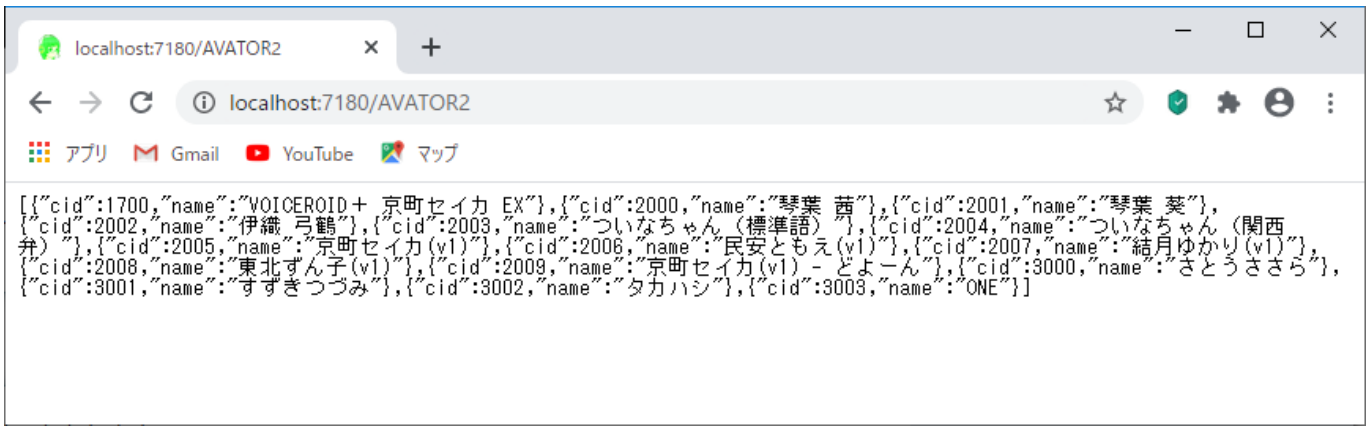
```
curl -X POST -H "Content-Type: application/json" -d @x.json
http://SeikaServerUser:SeikaServerPassword@localhost:7180/SAVE2/2000 -o
2000.wav
```

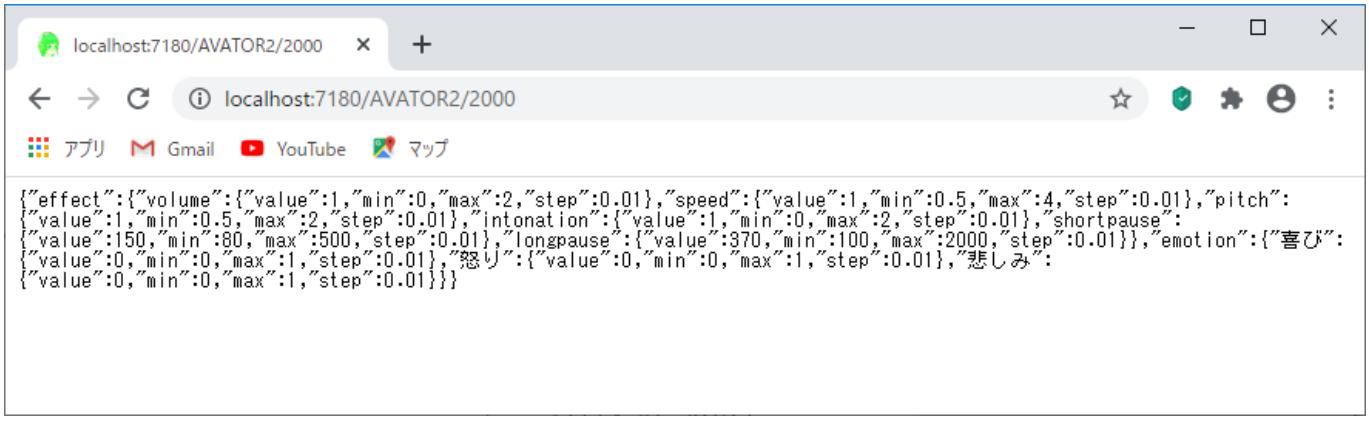
GETオンリーアクセス

GETメソッドだけで操作したい場合にこちらを使います※GETメソッドのサポートしかない組み込み環境から使いたいとの事で対応してみた

最初に /AVATOR2 で利用可能な話者のcidを取得/AVATOR2/{cid} で適用可能なエフェクト・感情、のパラメタ名と設定値範囲を取得します。

```
/AVATOR2
/AVATOR2/2000
```

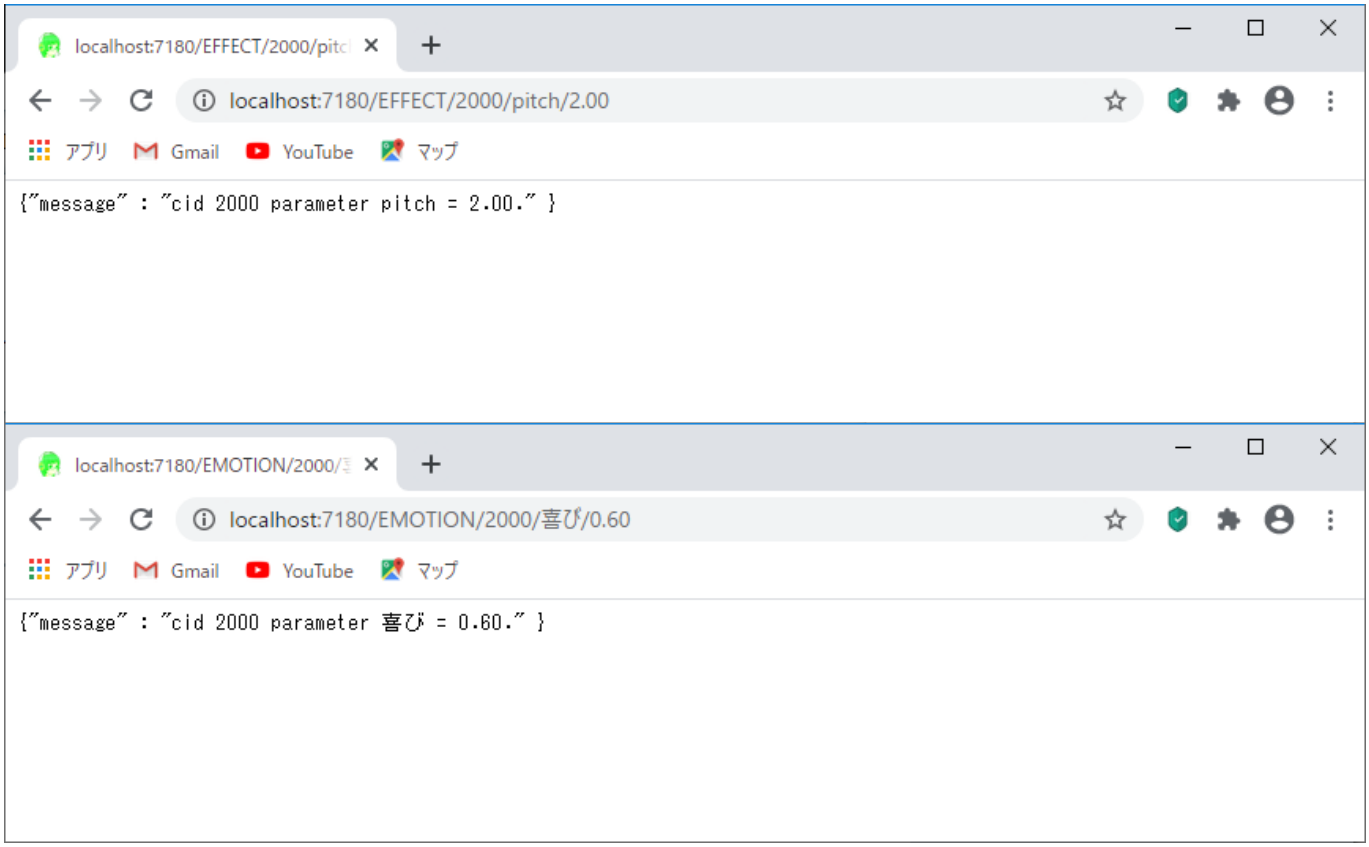




その後、/EFFECT/{cid}/{param}/{value} や /EMOTION/{cid}/{param}/{value} でエフェクト・感情パラメタに値を設定します。

```
/EFFECT/2000/pitch/2.00
/EMOTION/2000/喜び/0.60
```

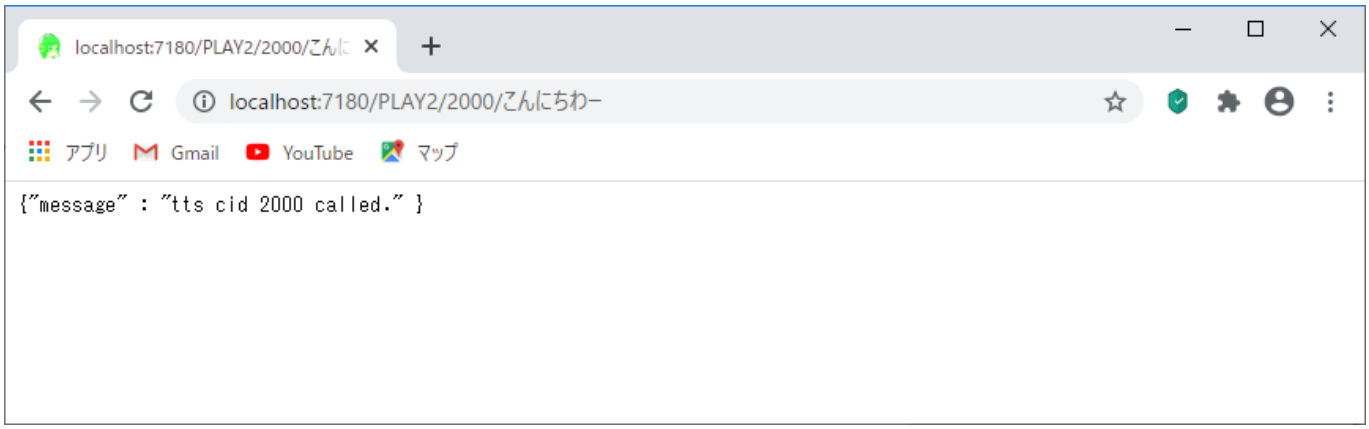
上記はエフェクト pitch に 2.00,感情パラメタ 喜び に 0.60 を指定した例です。日本語文字列はUTF8のテキストをURLエンコードしたものになります。



/PLAY2/{cid}/{talktext}, /SAVE2/{cid}/{talktext} のtalktextはURLエンコードしたUTF8のテキストです。

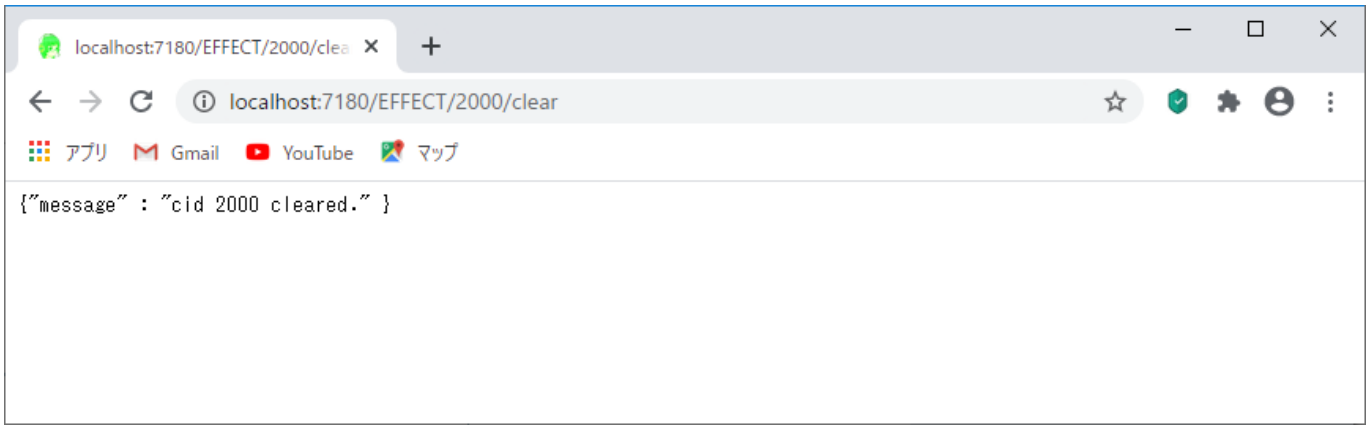
```
/PLAY2/2000/こんにちわー
/SAVE2/2000/こんにちわー
```

talktextで適用可能な長さ等の制限については検証していません。各環境で利用者自身が判断してください。これは発声の例(/PLAY2)です。



変更したpitch,喜び のパラメタを戻します。

```
/EFFECT/2000/clear
```



サンプリングレート変換機能

製作者環境下で試したものだけ。

#	メソッド	URI	説明
1	POST	/SAVE2/{cid}	TinySeikaServer利用時は音声がモノラル化される。たいてい44.1kHzだけど稀に48kHzの時がある。
2	POST	/SAVE2/{cid}/8000	SeikaCenterでキャプチャした音声を8kHzモノラルにリサンプリング。
3	POST	/SAVE2/{cid}/16000	SeikaCenterでキャプチャした音声を16kHzモノラルにリサンプリング。
4	POST	/SAVE2/{cid}/22050	SeikaCenterでキャプチャした音声を22.05kHzモノラルにリサンプリング。
5	POST	/SAVE2/{cid}/44100	SeikaCenterでキャプチャした音声を44.1kHzモノラルにリサンプリング。
6	POST	/SAVE2/{cid}/48000	SeikaCenterでキャプチャした音声を48kHzモノラルにリサンプリング。

例えば http://localhost:7180/SAVE2/2000/8000 へパラメタをPOSTすると8kHzにリサンプリングされた音声データがレスポンスとして返ってくる。

簡易Webサーバ機能

/app から始まるURIは簡易Webサーバ機能で使用されます。GETメソッドのみサポートされます。
/app/{path}はAssistantSeikaのHTTP機能設定タブで指定したドキュメントルートフォルダ直下に配置したコンテンツを返します。

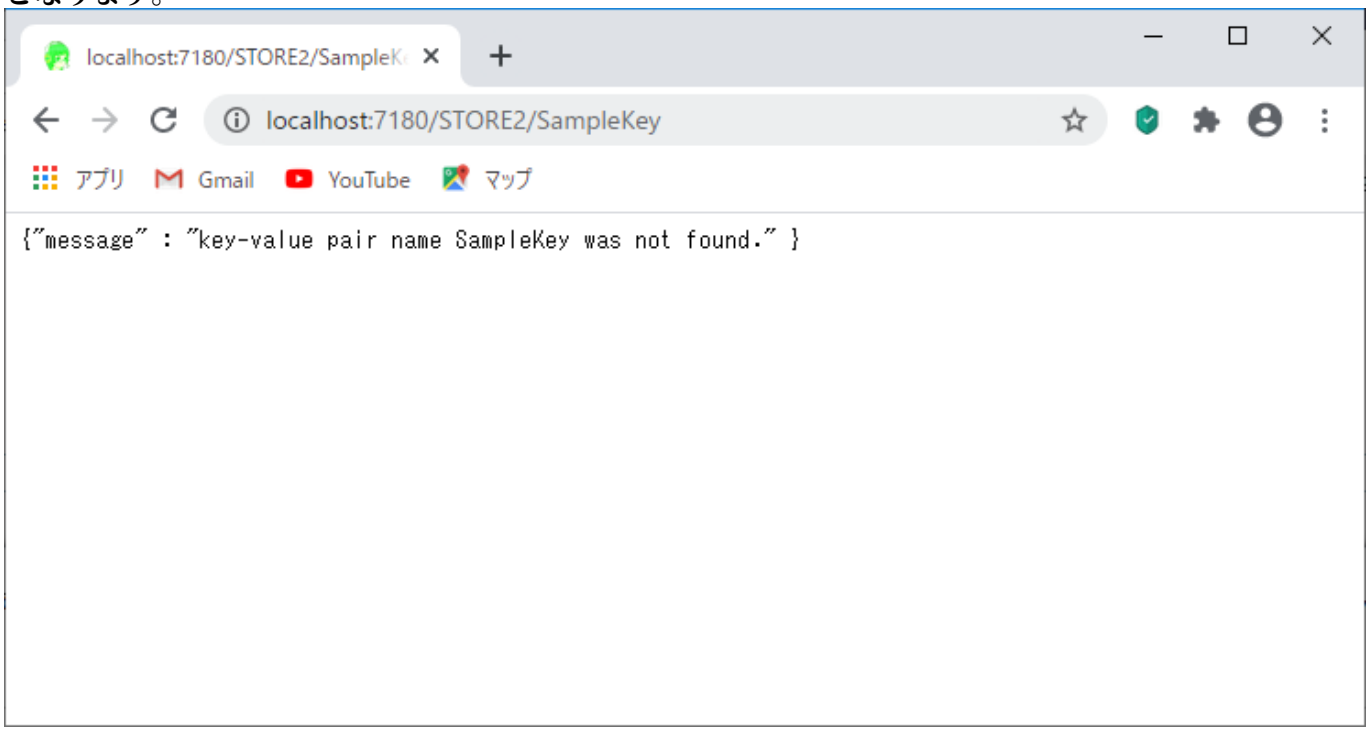
例えばドキュメントルートフォルダが C:\Docroot\web だった場合、http://localhost:7180/app/index.html のリクエストに対するレスポンスコンテンツは、C:\Docroot\web\index.html となります。

簡易KVS機能

/STORE2のURIで簡易KVS機能を利用できます。HTTPを話せるなら他の言語やツールからも利用できます。

登録と更新

何も登録がない状態でキー名 SampleKey に紐付けた値を表示してみます。未だ登録がないのでエラーとなります。



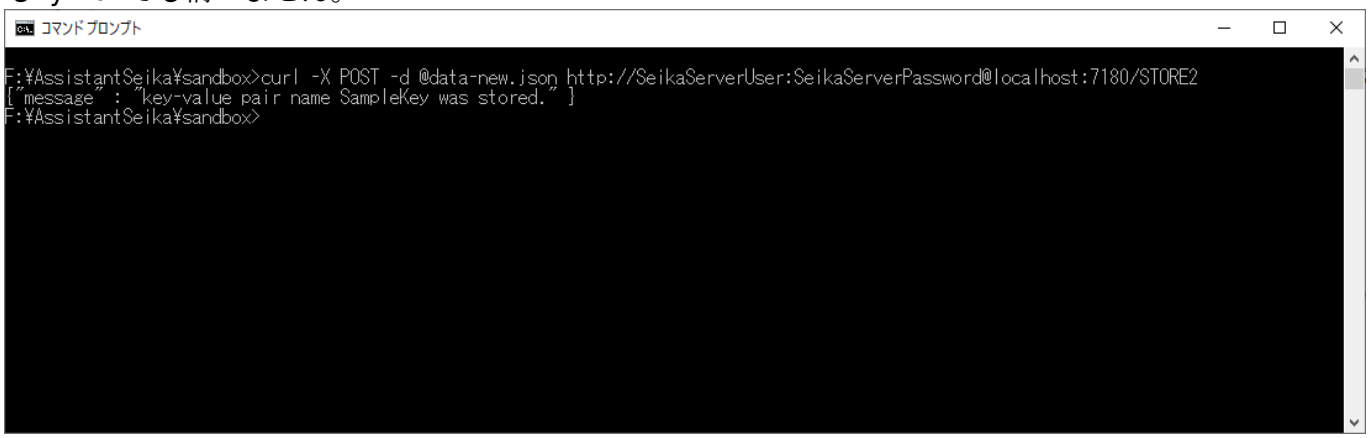
登録する情報は以下の形式のJSONをPOSTリクエストのBODYに指定します。“name”プロパティは検索時のキーになります。“value”プロパティは文字列配列になります。※プロパティ“value”の要素数は増やせます

data-new.json

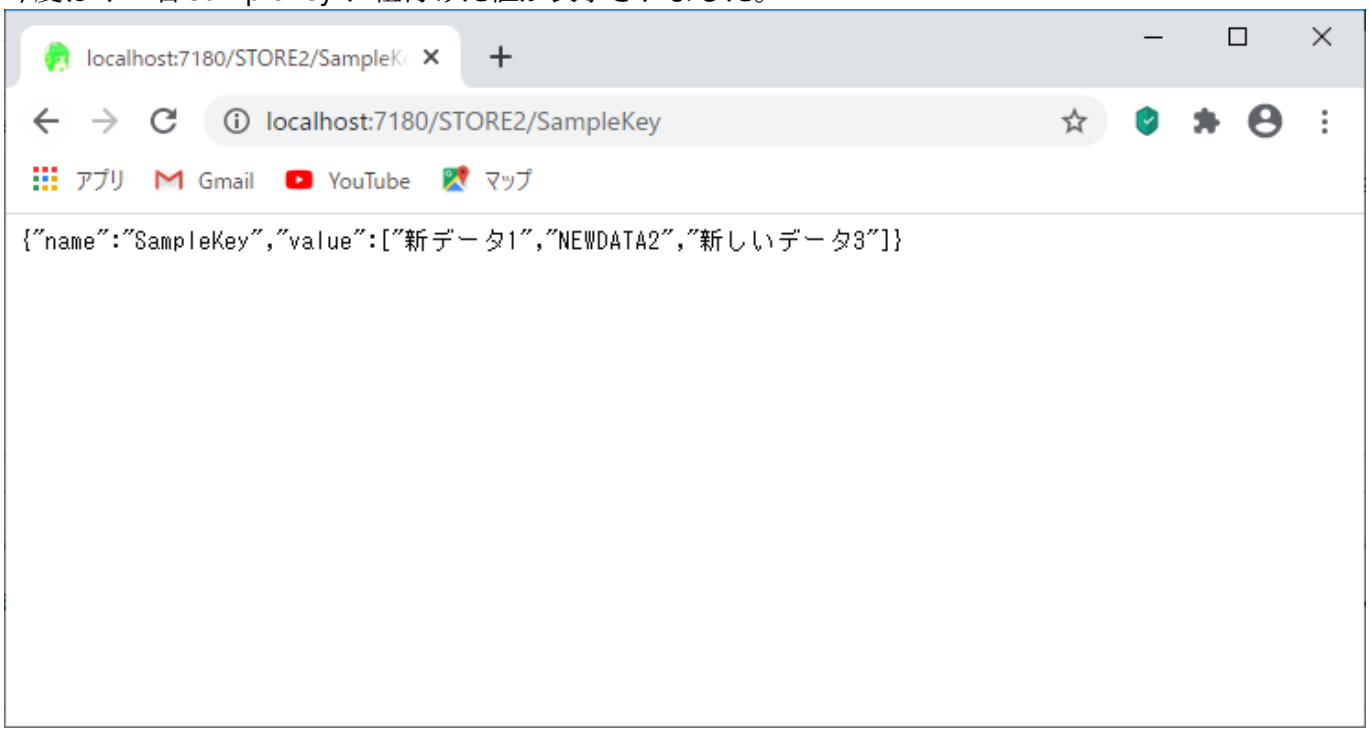
```
{
  "name": "SampleKey",
  "value": [
    "新データ1",
    "NEWDATA2",
    "新しいデータ3"
```

```
    ]
}
```

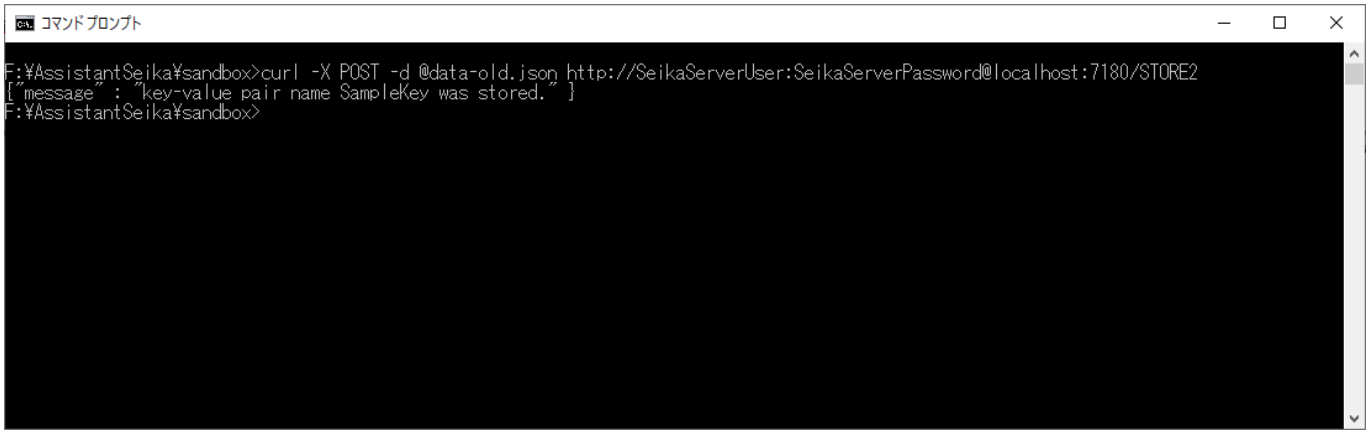
次にcurlコマンドで値を登録します。HTTPのPOSTを理解できるなら JavaScriptでもRubyでもPerlでもPythonでも構いません。



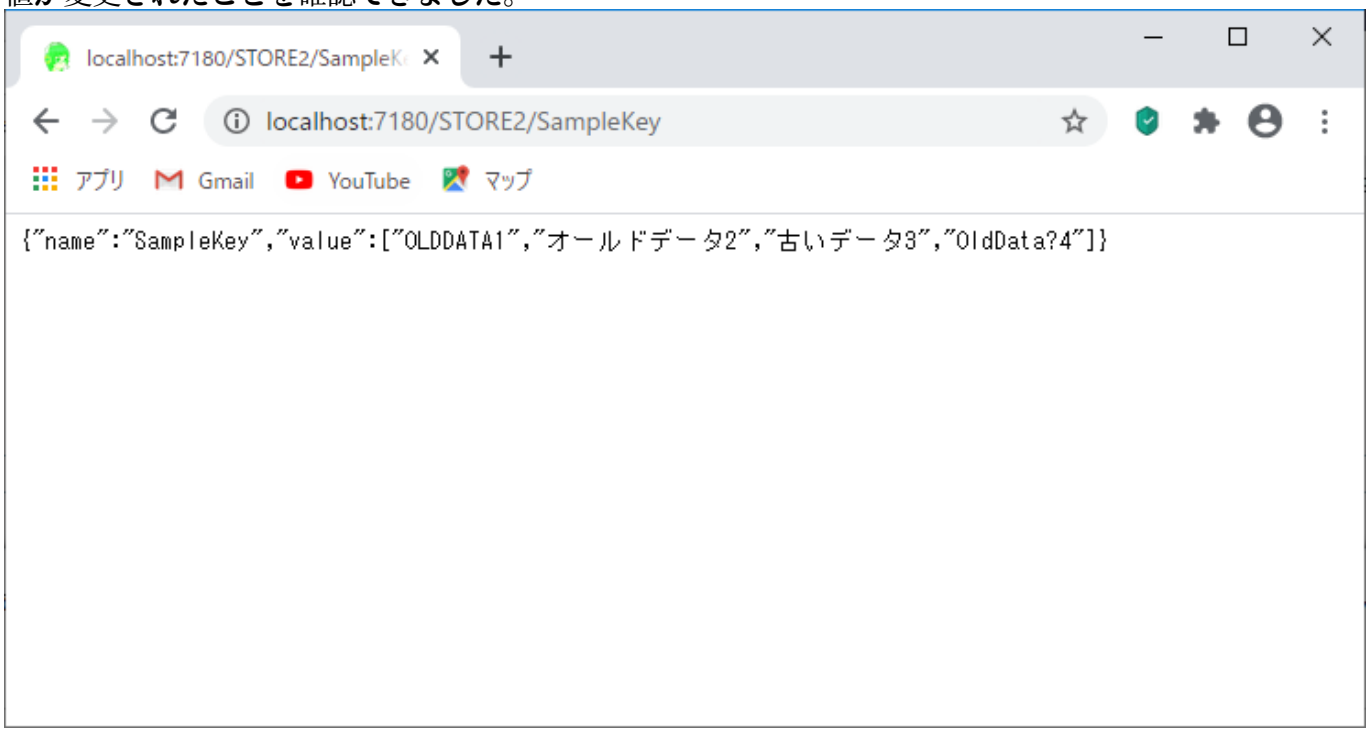
今度はキー名 SampleKey に紐付けた値が表示されました。



値を変更して再登録します。



値が変更されたことを確認できました。

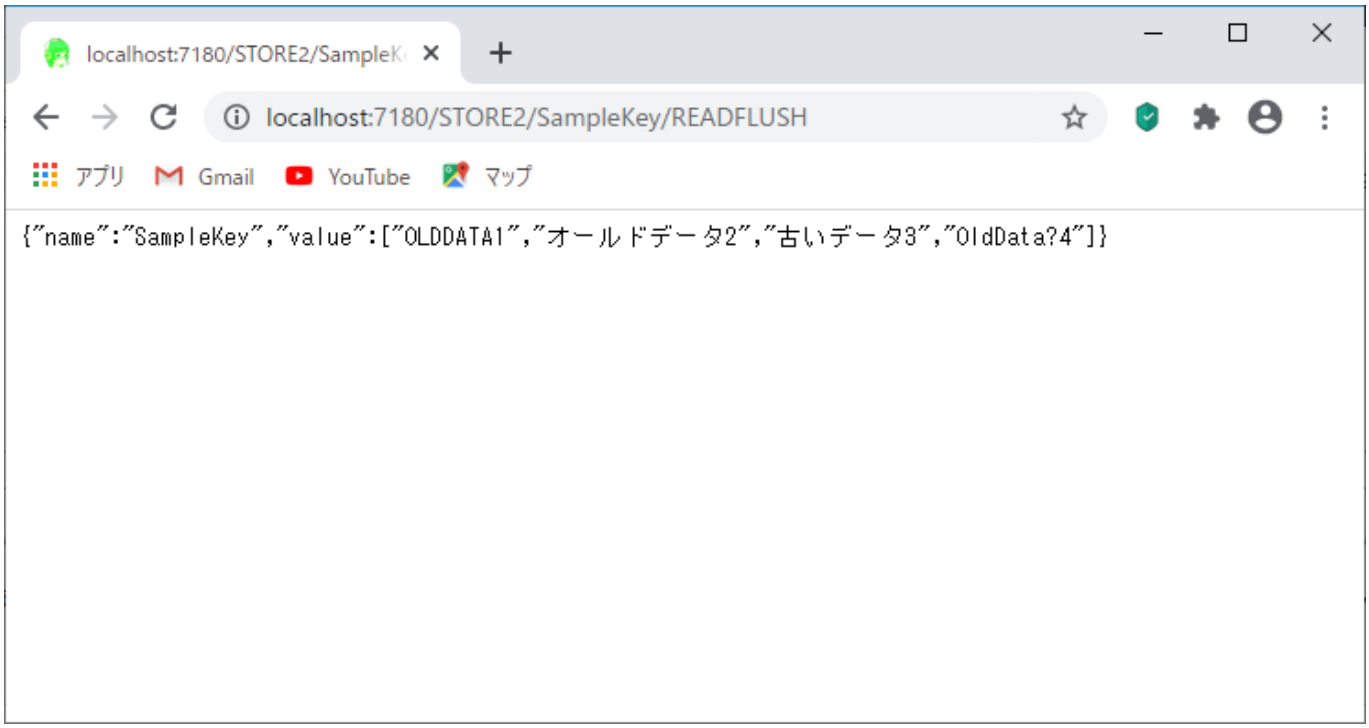


簡易Webサーバ機能の停止や再起動で登録内容は失われます。

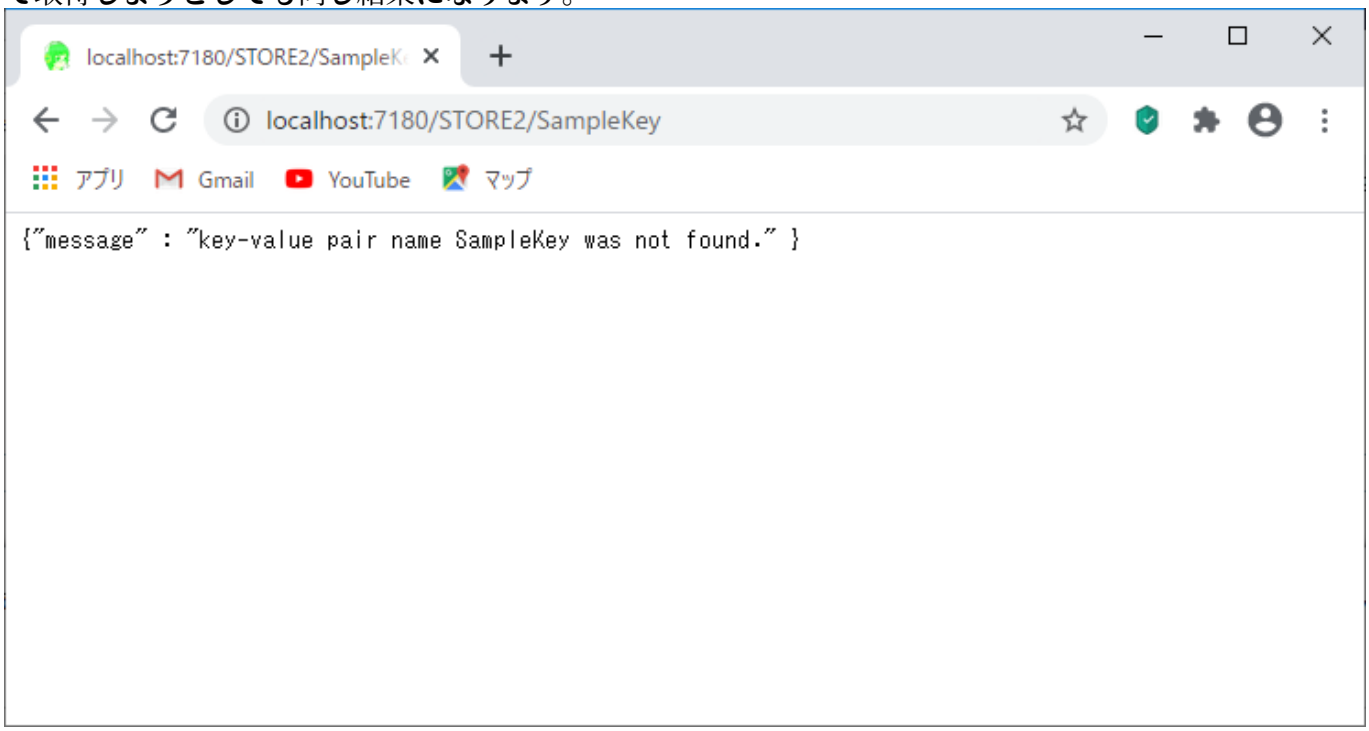
消去と抑止

URIに修飾子 READFLUSH を付けて消去を行うことができます。

キー名 SampleKey を修飾子付きで読み出し情報取得と消去を同時に行います。

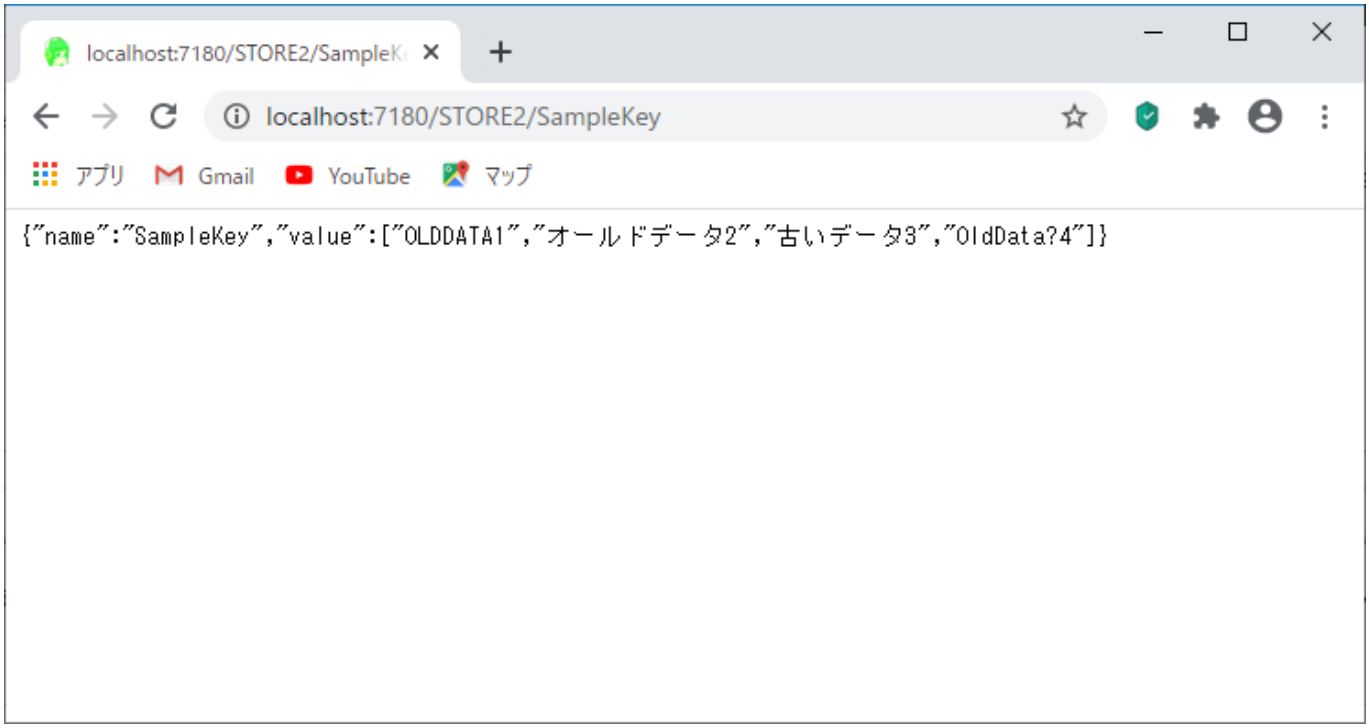


再度キー名 SampleKey を取得しても既に存在しない事が分かります。/STORE2/SampleKey/READFLUSH で取得しようとしても同じ結果になります。



URIに修飾子 NEWDECLARE を付けて上書き抑止を行うことができます。

キー名 SampleKey が登録されていることを確認します。



キー名 SampleKey へ上書きを試みましたが失敗します。

キー名 SampleKey の内容に変更がないことを確認できました。

処理Aが登録処理、処理Bが読み出し処理、を非同期で行う場合、

- 処理Aは /STORE2/NEWDECLARE で処理Bの未処理データ破壊を回避できます。
- 処理Bは /STORE2/{keyname}/READFLUSH で処理Aに対して処理済みなので更新して構わないことを明示できます。

技術資料, Windows, AssistantSeika, HTTP, Web