2017/12

Ver. 1.10

MXNET2 サーバ説明書

設定ファイル & 起動

有限会社リビッグ

http://www.ribig.co.jp/matrix/remote_api

mail: matrix@ribig.co.jp

内容

1. 配布ファイル

+x64 (64 ビット版サーバ＆クライアント API)

+clientlib

Mxnet2_api.dll	API DLL
Mxnet2_api.lib	mxnet2_api.dll 用インポートライブラリ
Mxnet2st_mt.lib	API スタティックライブラリ(/MT)
Mxnet2st_md.lib	API スタティックライブラリ(/MD)
mxnetapi.ini	ライブラリ設定ファイルサンプル

+server

Mxnet2.exe	mxnet2 サーバ
Mxnet2_ui.exe	mxnet2 サーバ用 UI プログラム
Mxnet2_detect.exe	mxnet2 サーバ用 UI プログラム
Mxnet2.ini	mxnet2 サーバ設定ファイル
Alert.wav	警告サウンドファイル

+x86 (32 ビット版サーバ＆クライアント API)

x64 と同じフォルダ、ファイルを含みます

+Samples

+C Sample

testDLL.sln	VS2013 ソリューションファイル

+C# Sample

testcsharp.sln	VS2013 ソリューションファイル

userpass.exe	ユーザパスワード暗号化プログラム
userpass.ini	設定ファイル

2. Mxnet1 からの変更点

ファイル名の mxnet1 文字が mxnet2 になりました

クライアント API の設定ファイル名が mxnetapi.ini になりました。DLL、スタティックライブラリとも同一名の設定ファイルを読み込みます

1. クライアント API の設定ファイルの [Options] が [Option] になりました。サーバ用設定ファイル Mxnet2.ini に一致させました。

2. Mxnet2.exe と mxnet2_detect.exe のサービス登録オプションが同一になりました。

    a. Mxnet2 -install / mxnet2 -remove
    b. Mxnet2_detect -install / mxnet2_detect -remove

3. AppSlot の名前割り当て方法を拡張しました。
    a. Mxnet1

        [dongle1]
        5=xxxxxx
        6=yyyyy

    USB ポートで最初に見つかったドングルに対して dongle1、2 番目に検出されたドングルに対して dongle2 が適用されていました。
    これまでの方法に加え、シリアル番号でドングルを特定することができるようになりました。USB ではどの順番でドングルが検出されるのか分かりません。1 つのドングルだけを接続していれば問題はありませんが、複数ドングル接続では設定が対象とするドングルに提供されることは保証されませんでした。

    b. Mxnet2

        [dongle1]
        Sernr = 0000000000
        Name=dongle name
        5=xxxxxx
        6=yyyyyy

シリアル番号でドングルが識別されるようになりました。ドングルにも名前を割り当てることができるようになりました． SerNr を指定しない従来の方法でも ドングルの名前を Name で指定できます。

4. C サンプルソリューションのプロジェクトを分かりやすい配置に変更しました。

5. "接続状況表示"メニュで表示されるウィンドウで、選択したログインをログアウトさせるには（バグにより）時間がかかっていました、このような問題点を修正しました。

3. Mxnet2 サーバのインストール

配布ファイルは 32 ビット版(x86)と 64 ビット版(x64)のサーバを含みます。64 ビット Windows では 32 ビット版/64 ビット版のどちらでも、32 ビット Windows では 32 ビット版をご利用ください。

x86/x64 内の server フォルダの 4 つのファイルをサーバとなるコンピュータの固定ディスクにコピーしてください。4 つのファイルは同じフォルダに配置しなければなりません。

1. Mxnet2.exe (mxnet2 サーバ本体)
2. Mxnet2_ui.exe (mxnet2 サービス用ユーザインターフェース)
3. MxNet2_detect.exe (mxnet2 サーバ起動＆監視プログラム)
4. Mxnet2.ini (mxnet2 設定ファイル)

4. MxNet2 サーバ設定ファイル

MxNet2 は起動時に設定ファイル MxNet2.ini を読み込みます。起動前に MxNet2.ini の内容を確認してください。Port については利用環境に合わせて変更してください。

3.1 mxnet2.ini

```
[Option]
#言語設定 英語表示は Lang=eng
Lang=jpn

# mxnet2 が待ち受けするポート番号
Port=12300

#LogIn_MatixNet 呼出し後、一定時間を経過すると自動的に
#ログアウトされてしまうため、LogIn_MatrixNet でログイン
#成功後は、一定時間内に LogIn_MatixNet を#呼び出し続けな
#ければならない。
#自動ログアウトされる時間を秒で指定
LoginTimeOut=300

#Init_MatrixAPI でセッションを開始後、一定時間内に
#リモート API のいずれかを呼び出さなければ、セッションは
#自動的に閉じられてしまうため、一定時間内に API を呼び出す
#ようにすること
# Init_MatrixAPI 開始セッションが自動的に閉じられるまでの時間を秒で指定
SessionTimeOut=1800

#ライセンス管理をプログラム単位（セッション単位）
#にするのか、コンピュータ単位にするかを決定
#既定：プログラム単位
SessionBaseLicense=1

# mxnet2 サーバとクライアント間通信は暗号化される。
# 暗号化パスワード交換は、サーバ側で秘密鍵/公開鍵キーペアを生成、
#LogFile
LogFileFolder=
LogLevel=0
```

CNG 版には鍵サイズ指定オプションは存在しません

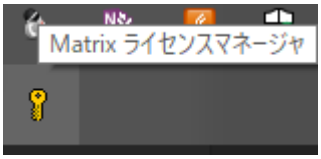
MxNet2 サーバが待機する既定ポート番号は 12300 です。クライアント側の設定と一致する値をセットしてください。

5. MxNet2 の起動方法

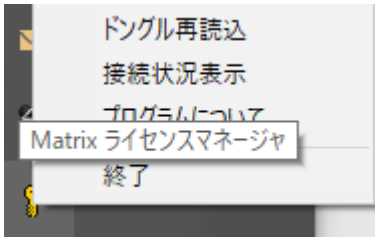
Mxnet2.exe は通常のアプリケーション、又は、Windows サービスとして実行できます。

5.1 通常のアプリケーション

起動後に画面右下のタスクトレイに黄色い鍵のアイコンが表示されます。



a. アイコン右クリックの「終了」で MxNet2 を終了できます。



b. 「接続状況表示」で現在接続しているセッション（クライアント、利用しているドングル、LogIn_MatrixNet）の情報を表示します。セッション途中で異常終了したセッションやログイン状態を、このウィンドウで強制ログアウトさせることができます。

5.2 Windows サービス

ファイアウォールの設定

Windows サービスとして実行する前に Windows ファイアウォールで MxNet2 が使うポートを必ず開けるようにしてください。ポートが開いていないと MxNet2 は起動しますが、

クライアントは接続できません。

Windows サービスは管理者権限のあるユーザだけが登録可能です。UAC が有効な Windows Vista/7/8/10 では管理者ユーザであっても、権限を昇格させてから実行する必要がありますので、DOS プロンプトを管理者として開いてください。

Mxnet2 をサービスとして登録

- Mxnet2 -install

又は

- Mxnet2 /install

サービスとして登録後、サービスを開始します。開始するとメッセージが表示されます。メッセージが表示されるまで 5-10 秒ほどかかることがあります。

Windows サービスとして起動すると通常のアプリケーションとして実行した場合と同じように画面右下のタスクトレイに黄色い鍵のアイコンが表示されます。 このアイコンは、mxnet2 サービスが表示するものではありません。mxnet2.exe と同じフォルダ内の Mxnet2_ui.exe が表示するものです。タスクトレイ内の黄色い鍵アイコンを右クリックして「終了」を選択すると mxnet2_ui.exe が終了します。MxNet2 サービスが停止するわけではありません。Mxnet2 サービスが開始していれば、いつでも手動で mxnet2_ui.exe を実行できます。

重要：Windows-サービスの登録では、プログラムのパスをレジストリに記録されます。Mxnet2 をサービス登録すると、登録して時点の Mxnet2 のパスが登録されます。登録後にパスを変更したり削除すると mxnet2 サービスは開始することができません。また、取り外し可能なメディア上の mxnet2 を Windows サービス登録すると、メディアが存在しない場合やドライブ名などが変更された場合、問題が発生してしまいます。必ず固定ディスクなどに起動に必要なファイルを置くようにしてください。

Windows サービスの解除コマンドは以下の通りです。コマンドプロンプトを管理者として開いて実行してください。

- Mxnet2 -remove

又は、

- mxnet2 /remove

1. Mxnet2 サーバがアプリケーションとして起動しているときに、mxnet2 サービスを開始すると、アプリケーションを終了してサービスが開始します。
2. mxnet2 サービスが動作中に、Mxnet2 をアプリケーションとして起動しようとすると、サービスが優先されますのでアプリケーションとしては起動しません

6. MxNet2 サーバの起動、状態監視プログラム

Mxnet2 サービスの可用性を高めるため、mxnet2 の異常をプロテクトする監視サービスを利用いただけます。Mxnet2 サーバーが何等かの原因で異常終了してしまうとクライアント要求への応答ができなくなります。そのような場合、手動で再開しなければなりません。このような事態を避けるため Mxnet2 サーバが動作しているかどうかを常に監視して、動作していなければ自動的に Mxnet2 を開始するサービスを登録してください。

監視プログラムは mxnet2.exe と同じフォルダ内にある Mxnet2_detect.exe です。

1. MxNet2.exe と Mxnet2_detect.exe は同じフォルダ内になければなりません
2. Mxnet2_detect サービスは MxNet2 が動いているかどうかを定期的に確認します。mxnet2 から応答がなければ mxnet2 サービスの開始を試行します。
3. Mxnet2 を再開しようとしても動作が確認できなければ、同じフォルダ内の alerm.wav サウンドファイルを再生して警告音を出します

監理者として開いたコマンドプロンプトで実行してください。

Mxnet2_detect の Windows サービス登録

```
>mxnet2_detect -install
```

又は

```
>mxnet2_detect /install
```

Mxnet2_detect の Windows サービス解除

```
>mxnet2_detect -remove
```

又は

```
>mxnet2_detect /remove
```

サービスマネージャで mxnet2 サービスを停止すると mxnet2_detect サービスも停止します。

7. MxNet2 作動概要と設定

Remote API は Matrix API とほぼ同じ呼び出し方で利用できます。しかし、Matrix API クライアントは直接ドングルとやり取りする一方、RemoteAPI では通信チャネルを経由してサーバとやり取りします。クライアントとサーバ間でどのようなやり取りが行われるかが分ると異常発生時に原因を特定しやすくなります。

7.1 rInit_MatrixAPI & rRelease_MatrixAPI

Matrix API の初期化関数 Init_MatrixAPI, 解放関数 Release_MatrixAPI は他の API 呼び出し前後に呼び出します。RemoteAPI でも必ず最初に rInit_MatrixAPI を、最後に rRelease_MatrixAPI を呼び出すようにします。

RemoteAPI の rInit_MatrixAPI はリモートサーバとのセッションを開始します。他の RemoteAPI は rInit_MatrixAPI が開始したセッション内でサーバとやり取りします。rRelease_MatrixAPI はクライアントのセッションをリモートサーバからクローズします。セッションを閉じると、rInit_MatrixAPI 以外の API 呼び出しは必ず失敗します。

rInit_MatrixAPI でセッションを開始したら、必ず rRelease_MatrixAPI でセッションを閉じてください。もし、セッションを閉じなければ、セッションタイムアウト処理によって閉じられるまでサーバ側に残ったままになります。そのようなセッションはサーバの処理に悪影響があるわけではありませんが、サーバ側で”生きている‟セッションと”死んでいる‟セッションとを見分けることは困難なため、人によるセッションの管理を困難にしてしまいます。

7.2 セッション

クライアントが rInit_MatrixAPI を呼び出すと、リクエストがサーバに送られ、サーバはクライアントを識別するための情報を記録し、セッション ID をクライアントに送ります。その後のクライアントからのサーバへのリクエストには、必ずそのセッション ID が含まれています。Mxnet2 サーバは、このセッション ID でクライアントを識別します。

rRelease_MatrixAPI のリクエストがサーバに送られると、セッション ID に対応した情報がサーバ上から削除されます。rRelease_MatrixAPI を呼び出さなければ、rInit_MatrixAPI で開始したセッションは、セッションタイムアウトになるまでゾンビセッションとして残ってしまいます。セッションタイムアウト処理によって、ゾンビセッションは自動削除されます。

7.3 セッションタイムアウト

セッション開始後、Remote API が呼び出される毎にサーバはセッションの最終アクセス時間を更新します。セッションタイムアウトは、セッションで最後に Remote API を呼び出した時間を基準に計算されます。セッションは、その最終アクセス時間が、現在時間よりタイムアウト時間以上前であるとタイムアウトします。

タイムアウト時間内に Remote API を呼び出さなければサーバはセッションを閉じてしまいます。サーバ側のタイムアウト処理によってセッションが閉じてしまうと、クライアントの Remote API 呼び出は失敗します。そのような場合 rInit_MatrixAPI を呼び出して新しいセッションを開始してください。

セッションタイムアウトの既定値は 1 時間(3600 秒)です。サーバのセッションタイムアウトは設定ファイルで調整可能です。

なお、ゾンビセッションは「接続状況表示」で強制的にログアウトさせることができます。

7.4 Remote API セッション vs. 通信セッション

rInit_MatrixAPI が開始するセッションは通信セッションとは異なります。MxNet2 では、http プロトコールのようにクライアントからサーバにコマンドを送り、サーバの結果をクライアントが受け取ると通信は切断します。通信回線をセッション開始から終了まで占有するわけではありません。Remote API を発行する毎にサーバへのコネクト、送受信、切

断が発生します。

Remote API のセッションとは、API 呼び出し間で状態を記憶するためのサーバ側とクライアント側で共通のセッション情報を保持している状態のことです。サーバ側が一方的にセッションを閉じるとセッション情報が破棄され、クライアントはサーバとやり取りが行えなくなります。クライアントはいつでも rInit_MatrixAPI で新しいセッションを開始できます。セッションを rRelease_MatrixAPI で閉じないと、サーバ側には使われなくなったセッション情報が残ってしまいます（ゾンビセッション）。

7.5 マルチスレッドでの rInit_MatrixAPI & rRelease_MatrixAPI

RemoteAPI はスレッド毎にセッションを作成することはありません。１つのプロセスでセッションが開始されたら、すべてのスレッドはそのセッション内でサーバと通信します。

複数のスレッドから rInit_MatrixAPI を呼び出すと、最初の rInit_MatrixAPI の呼び出しでサーバとのやり取りが行われ、サーバ側でセッションが作成されます。その後の rInit_MatrixAPI 呼び出しではクライアントライブラリ内部カウンターが１つ増加して、rInit_MatrixAPI の呼び出をカウントするだけです。サーバとのやり取りは発生しません。

逆に rRelease_MatrixAPI は内部カウンターが 1 より大きければ、カウンターを１つ減少させるだけです。サーバとのやり取りは発生しません。カウンターが 1 の時に rRelease_MatrixAPI を呼びだすと、サーバとやり取りが行われてプロセスのセッションが閉じます。

内部カウンターのリセット

内部カウンタをリセットしなければ、通信が行えない状況が発生することがあります。例えば、2 つのスレッドが rInit_MatrixAPI を呼び出し、内部カウンターが 2 の状態で、サーバが落ちたとします。サーバを再起動後に rInit_MatrixAPI を呼び出しても、内部カウンターが 2 のままではサーバとのやり取りは発生しません。rRelease_MatrixAPI を内部カウントが 0 になるまで呼び出すか、リセット API Reset_MatrixAPI を呼び出してください。Reset_MatrixAPI は無条件で内部カウンタを 0 にします。この API 呼び出し後、rInit_MatrixAPI は必ずサーバとのやり取りを行います。

```
void   __stdcall rReset_MatrixAPI(void);
```

```
[DllImport("mxnet2_api.DLL", EntryPoint = "rReset_MatrixAPI", CallingConvention =
CallingConvention.StdCall)]
public static extern void rReset_MatrixAPI();
```

7.6 シングルスレッドプログラムと内部カウンタ

単一スレッドのプログラムでも内部カウンタを考慮する必要があります。rInit_MatrixAPI でセッションを開始後、サーバ側のタイムアウト処理でセッションが閉じられたとします。この時、内部カウンタは 1 です。その後 クライアントが保持しているセッションで Remote API を呼び出しても失敗します。セッションを再開するため、rInit_MatrixAPI を呼び出したとします。セッションは開始しません。内部カウンタが 2 になるだけです。rInit_MatrixAPI がサーバとのやり取りするのは内部カウントが 0 の時だけです。
セッション再開前に rRelease_MatrixAPI を呼び出して戻り値が 0 であることを確認してから、rInit_MatrixAPI を呼び出すとセッションは開始します。または、リセット API Reset_MatrixAPI 呼び出し後、rInit_MatrixAPI を呼び出してください。

7.7 ライセンス管理 rLogIn_MatrixNet & rLogOut_MatrixNet

Remote API には rLogIn_MatrixNet 関数があります。メモリフィールドを使ってライセンス管理を行うためのものです。クライアントがこの API を初めて呼び出すと指定 AppSlot のライセンス数が 1 つ減ります。同じセッションで同じ AppSlot に対して再度 rLogIn_MatrixNet を呼び出してもライセンス数が減少することはありません。取得済みのライセンスを維持するための呼出しになります。

rLogOut_MatrixNet は必ずライセンス数を 1 つ戻します。

rLogIn_MatrixNet 呼出してライセンスを取得後、ログアウトせずにプログラムが異常終了してしまうと AppSlot のライセンス数が 1 つ減った状態のままになります。ただし、rLogOut_MatrixNet を呼び出さなくても rRelease_MatrixAPI 呼出しによって、そのセッション内のすべての AppSlot へのログインは自動的にログアウトします。

サーバは、rLogIn_MatrixNet が呼出される毎にセッションの rLogIn_MatrixNet 呼出し時間を記録しています。ログインタイムアウトで指定した時間内に rLogIn_MatrixNet を呼び出さないと、mxnet2 サーバによって強制的に AppSlot からクライアントはログアウトさせられてしまいます（ログインタイムアウトは設定ファイルで調整可能です)。タイムアウト時間内の同じ AppSlot に対して rLogIn_MatrixNet 呼び出すことで、 AppSlot の

ライセンスを維持できます。ライセンスを維持するには繰り返しタイムアウト時間内にrLogIn_MatrixNet を呼び出してください。

ログアウトタイムアウトの既定値は 5 分(300 秒)です。

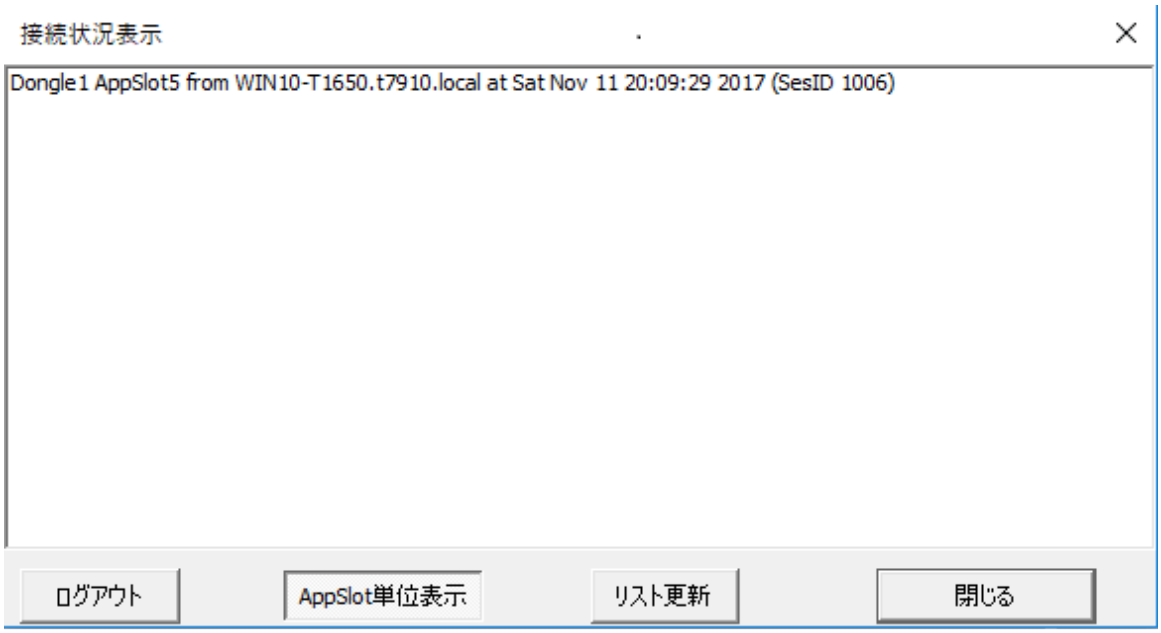
rLogIn_MatrixNet と rLogOut_MatrixNet は、一対で使う API ではありません。

例:マルチスレッドプログラムのスレッド A で AppSlot 1 に対して rLogIn_MatrixNet を呼び出すと、プログラムがライセンスカウントを取得します。別スレッド B で AppSlot 1 に対して rLogIn_MatrixNet を呼び出すと AppSlot1 のライセンスを維持するだけで、新しくライセンスカウントが取得できるわけではありません。スレッド B で rLogOut_MatrixNet を呼び出すと、ライセンスカウントが減少して、プログラムがライセンスを戻したことになります。その後、スレッド A で rLogOut_MatrixNet を呼び出しても、すでにプログラムは AppSlot1 に対するライセンスを取得していない状態なのでエラー(-11)になります.

同一 AppSlot に対して rLogIn_MatrixNet をどのスレッドから何度呼び出しても最初の呼び出しによってライセンスを取得（カウントが 1 つ減少）するだけで、その後の呼び出しはライセンスを維持するための呼び出しになります。rLogOut_MatrixNet はどのスレッドから呼び出しても 1 度呼び出すとプログラムのライセンスを戻す（カウントが 1 つ増加）ことになります（ ただし、ライセンスタイプが IP の場合、rLogOut_MatrixNet は異なる挙動をします ）

7.8 AppSlot / ドングルへの名前割り当て

Mxnet2 サーバのアイコンを右クリックして"接続状況表示"を選択すると、サーバが管理するセッションとログインの情報の一覧を表示できます。この表示では、最初に見つかったドングルが Dongle1，2 番目のドングルが Dongle2 等として表示されます。AppSlot はスロット番号が付いて AppSlot5, AppSlot10 などと表示されます。



具体的にどのドングルなのか、また、AppSlot を利用するアプリケーションが何なのかを分かりやすくするために、ドングルと AppSlot に任意の文字列を割り当てることができます。接続するドングル分のセクションを mxnet2.ini 内に作成してください。

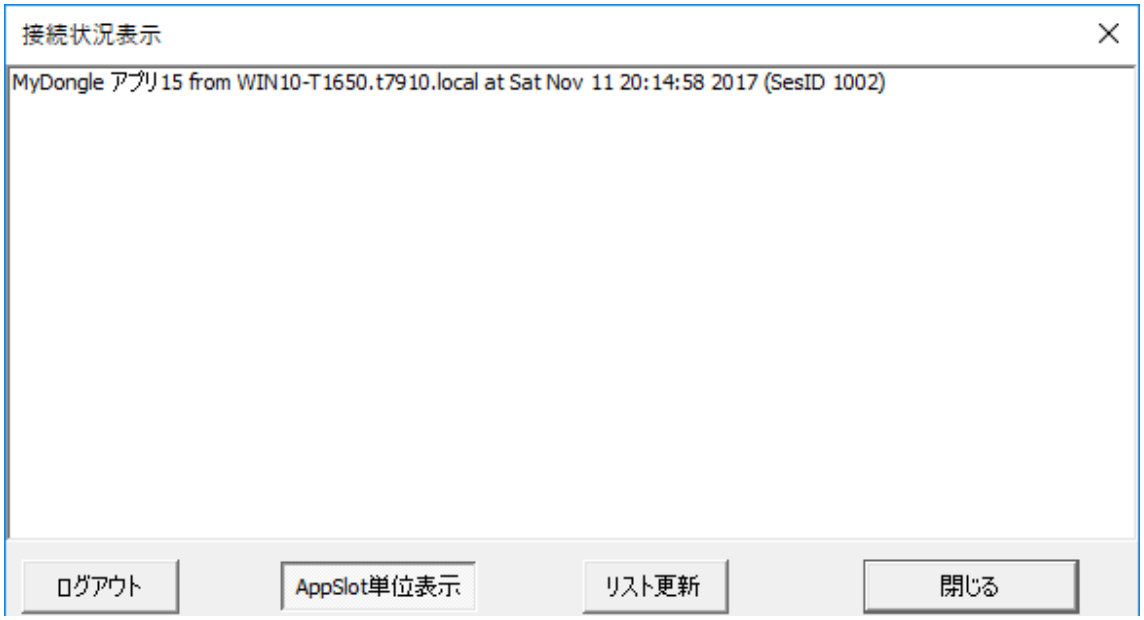
例： 3つのドングルを利用することを想定。同時に接続する必要はなく、いずれかのドングルが接続されることが前提となります。

1. 3つのドングルのデータベースを1から3まで作成.1 から順番に番号を付けていきます。番号を飛ばすことはできません. 1, 3,4 とすると、1番目しか読み込まれません。

   [Dongle1]
   [Dongle2]
   [Dongle3]

2. 各セクションに SerNr(シリアル番号)、Name(ドングル名)、スロット番号のキーを作成

   例：
   [Dongle1]
   sernr=2041005424
   name=MyDongle
   5=アプリ 15



シリアル番号が一致するドングルが見つかると、そのドングルが見つかったセクションの Name, スロット名を表示するようになります。

シリアル番号を指定しない従来の指定方法も有効です。SerNr が指定されていない場合は、これまで通り、最初に検出されたドングルには[Dongle1]セクション、2 番目に検出されたドングルには[Dongle2]セクションが適用されます。

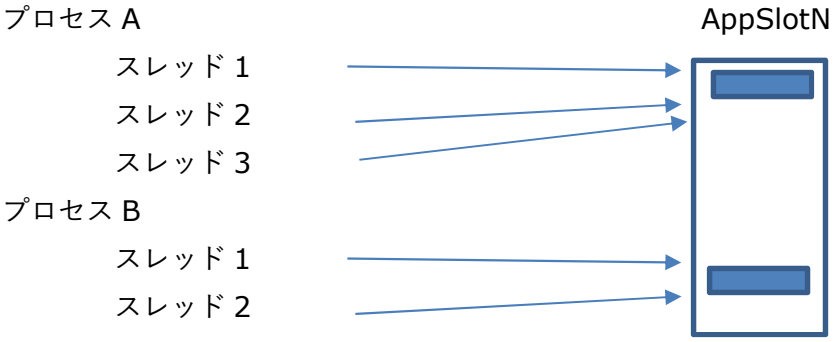
7.9 ライセンスタイプ

MxNet2 はクライアントから AppSlot へのログインを、クライアントのセッション ID か IP アドレスかによって管理します。既定はセッション ID の管理となってます (MxNet2.ini の SessionBaseLicense=1 が既定設定です)

a. セッション単位（プログラム）管理　SessionBaseLicense=1

ライセンス管理をプログラム単位にすると、同じ端末上で複数の rLogIn_MatrixNet を呼び出すプログラムがある場合、それぞれのプログラムが呼び出した LogIn_MatrixNet によって AppSlot のライセンス数が減少します。rRelease_MatrixNet を呼び出すとライセンスが戻ります

端末 A



- 端末 A のプロセス A のスレッド 1 が rLogIn_MatrixNet を呼び出して AppSlotN からライセンスを取得、また、クライアント API 内部カウンターが増加
- 端末 A のプロセス A のスレッド 2 が rLogIn_MatrixNet を呼び出してもサーバには送られずにクライアント API 内部カウンタが増加するのみ
- 端末 A のプロセス A のスレッド 3 が rLogIn_MatrixNet を呼び出してもサーバには送られずにクライアント API 内部カウンタが増加するのみ

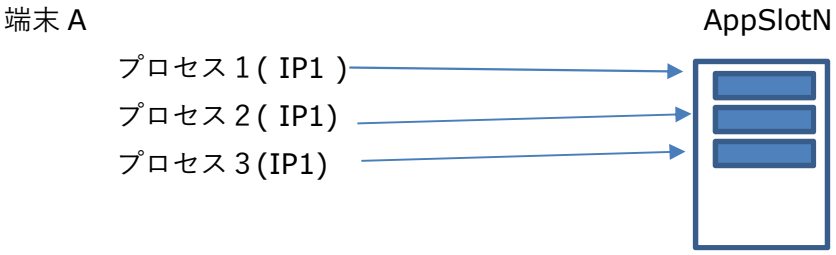
- 端末 A のプロセス A のスレッド 2 が rLogOut_MatrixNet を呼び出すと、クライアント API 内部カウンターが 3 なのでカウンターが減少するのみ
- 端末 A のプロセス A のスレッド 1 が rLogOut_MatrixNet を呼び出すと、クライアント API 内部カウンターが 2 なのでカウンターが減少するのみ
- 端末 A のプロセス A のスレッド 3 が rLogOut_MatrixNet を呼びだすと内部カウンタが 1 なのでサーバとやり取りが発生して AppSlotX のライセンスが戻る

- プロセス B はプロセス A と同じ

b. IP アドレス（端末）単位管理　SessionBaseLicense=0

一方、IP 単位にすると同じコンピュータ上の複数のプログラムが rLogIn_MatrixNet を呼び出したとしても、ライセンス数は 1 しか減少しません。Mxnet2 は同じコンピュータ上の複数のプログラムからの rLogIn_MatrixNet リクエストを IP で識別するため、そのコンピュータ上のすべてのプログラムからのリクエストを同一クライアントから発行されたものとして扱います。

rLogOut_MatrixNet の動作はセッション単位と異なります。クライアントからのリクエストはセッション ID でも管理しているため、同一端末上の複数のプロセスが rLogOut_MatrixNet を呼び出したとしても、同一端末上のログインしているプロセスの最後の rLogOut_MatrixNet の呼び出しまでライセンスは戻りません。



- 端末 A のプロセス 1 が AppSlotN からライセンスを取得
- 端末 A のプロセス 2 が rLogIn_MatrixNet を呼び出してもプロセス 1 が取得したライセンスを維持するのみ(プロセス 2 がログインしたという記録は保存される)
- 端末 A のプロセス 3 が rLogIn_MatrixNet を呼び出してもプロセス 1 が取得したライセンスを維持するのみ(プロセス 3 がログインしたという記録は保存される)

- 端末 A のプロセス 2 が rLogOut_MatrixNet を呼び出しても、プロセス 1，3 がログインしているためライセンスは維持されたまま(プロセス 2 がログインした記録は削除される)
- 端末 A のプロセス 1 が rLogOut_MatrixNet を呼びだしても、プロセス 3 がまだログインしているのでライセンスは維持されたまま(プロセス 1 がログインした記録は削除される)
- 端末 A のプロセス 3 が rLogOut_MatrixNet を呼びだすと IP1 からのログイン記録はプロセス 3 だけなので、プロセス 3 のログイン記録を削除して、AppSlotN のライセンスカウントが 1 つ増加する

各プロセス内ではセッション単位管理と同じように、クライアント API 内部カウンタによってサーバへのリクエストが発行されるタイミングが決まります。

7.10 サーバとクライアントコンピュータのクロックの同期

クライアントからの API 呼び出し時間がサーバへの要求に埋め込まれています。サーバはクライアントのリクエストのタイムスタンプとサーバ側の現在時を比較して、一定以上（既定では６０秒）の時間が経過しているとリクエストを受け付けません。

設定ファイルの［Option］で **CmdIssueTimeInterval** キーを使って要求の間隔時間調整が可能です。**CmdIssueTimeInterval** に秒単位の値を設定してください。

サーバとクライアントのクロックの同期が不可能な場合、MxNet2 にクライアントからのリクエストのタイムスタンプを確認させないようにすることが可能です。設定ファイルの［Option］に

CheckCmdIssueTime=0

と設定してください。

7.11 ログファイル

ログファイルを保存するフォルダを指定すると、指定フォルダにエラーログファイルを作成します。ログファイル名は mxnetYYYY-MM-DD.log になります。YYYY-MM-DD はログファイルを作成した日時です。

既定ではエラーのログを残します。LogLevel を指定することでさらに詳細なログを取得できます。LogLevel=1 で情報ログ、LogLevel=2 でデバッグログを出力します。

[Option]
#ログファイルを保存するフォルダ
LogFileFolder=c:¥mxnet_log
LogLevel=1 or 2

8. MxNet2 の返すエラー番号

エラー番号	エラーの内容
-100	コネクトエラー MxNet2 に接続（コネクト）できません
-102	Matrix キー未接続
-104	パケット時間エラー
-105	セッションが存在しない
-106	パケット長エラー
-107	パケット識別ヘッダ不良
-108	呼び出した API 関数はサポートされていない
-115	パケット番号エラー
-116	クライアント API が公開鍵を保持していない
-122	パケットバージョンエラー

-100 はもっとも一般的なエラーで、MxNet2 サーバに接続できないときに発生します。MxNet2 が起動/動作していない、ポートが開いていない、ポート不一致等、いくつかの原因で発生します。

-104 はサーバ PC とクライアント PC のクロックが同期していないと発生します。クライアントがサーバに送るパケットにはクライアント PC のクロックで時刻が設定されます。サーバはクライアントから送られたパケットの時刻をサーバ PC のクロックの時刻と比べて、一定の時間以上経過していたらこのエラーを発生させます。

-105 はタイムアウトによってサーバ側でセッションが閉じられたなどに発生します。Reset_MatrixAPI() -> rInit_MatrixAPI で新しいセッションを開始してください。

-108 はドングルへの書込み API を呼び出すと発生します。クライアント側で書込み API 呼び出しには -108 を返しますが、サーバ側に要求があったとしても MxNet2 が-108 を返します。

9. クライアント API

RemoteAPI はクライアント API として DLL と VC 用スタティックライブラリを含みます。

7.1 リモート API DLL(mxnet2_api.dll)
.NET 言語(C# 等)ではこの DLL を InteropServices でインポートします。 VC プログラムはインポートライブラリ(mxnet2_api.lib)をリンクしてください。他の C/C++コンパイラは、DLL からインポートライブラリを生成ツールで生成したインポートライブラリをリンクしてください。

リモート API DLL(mxnet2_api.dll)の設定ファイル ： mxnetapi.ini

リモート API DLL の API 動作は設定ファイル mxnetapi.ini 内の設定に従います。

```
[Option]
# MxNet2 が実行しているコンピュータの IP アドレス
ip=127.0.0.1

# MxNet2 サーバの待ち受けポート番号
port=12300

# サーバからの応答待ちタイムアウト(単位：ミリ秒）
timeout=3000

# remote=1 ... MxNet2 に接続（リモートドングル利用)
# remote=0 ... ローカルドングルにアクセス
remote=1
```

利用環境に応じて IP, Port, timeout, remote 設定値を調整してください。
IP, Port が正しく設定されていないと mxnet2 サーバにアクセスできません

RemoteAPI はローカルに接続された Matrix ドングルに直接アクセスすることができます。 Remote=0 とすると、mxnet2 サーバに接続しません。ローカルのドングルに直接アクセスします。

Remote=1 になっていると IP アドレスと Port 値を使って mxnet2 サーバに接続し、mxnet2 が返す値を待ちます。

7.2 スタティック API

Mxnet2st_mt.lib

このライブラリをリンクするプログラムはコード生成オプション /MT をセットします。VC++ランタイムは不要です。他のファイルに依存しません。

Mxnet2st_md.lib

このライブラリをリンクするプログラムはコード生成オプション /MD をセットします。プログラムを実行するには VC++ラインタイムを別途インストールしなければなりません。

スタティック API の設定ファイル

リモート API DLL(mxnet2_api.dll)同様に、スタティック API を正しく動作させるには設定ファイルが必要です。リモート API DLL 用の設定ファイルと同一ファイル (mxnetapi.ini) をスタティックライブラリをリンクしたプログラムと同じフォルダに置いてください。

7.3 ランタイム時オプション設定

プログラムからオプションを設定することもできます。設定ファイルを作成せずに、プログラムからオプションを設定するような使い方が可能です。

_mxINT32 WINAPI MxNet_SetPort(_mxINT32 _port)

サーバのポートを指定します。(設定ファイルの"port"に対応)

戻り値：呼び出し前に設定されていたポート番号

_mxINT16 WINAPI rSetConfig_MatrixNet(_mxINT16 nAccess, char* nFile)

- nAccess = 1 で設定ファイルの Remote=1 同等
- nAccess = 0 で設定ファイルの Remote=0 同等
- nFile にはサーバのコンピュータ名かIPアドレスを設定

戻り値： 1＝リモート　 0＝ローカル

WCHAR* WINAPI MxNet_SetIP(WCHAR* _ip)

サーバの IP アドレスを設定（ 設定ファイルの IP に対応）

通常は rSetConfig_MatrixNet の nFile で指定するため、このAPI呼び出しは不要

戻り値： 呼び出し前に設定されていた IP

_mxINT32 WINAPI MxNet_SetTimeOut(_mxINT32 nTimeOut)

タイムアウトを設定 (設定ファイルの"timeout"に対応)

戻り値： 呼び出し前に設定されていたタイムアウト

これらのオプション設定 API でオプションを設定してから rInit_MatrixAPI を呼び出してください（逆では意味がありません）。rInit_MatrixAPI は設定されたオプション値でサーバに接続します。

8. サンプルプログラム

配布ファイルには C/C++ と C# のサンプルプログラムが含まれます。

8.1 C Sample

Samples¥C Sample 内の test.sln ファイルに API DLL, スタティックライブラリをリンクするテストプロジェクトが含まれます。

複数のスレッドを開始して、各スレッドでテストルーチンを指定回数繰り返し呼び出します。10 スレッド、各スレッドで 50 回の繰り返しをすると、一連のテストルーチンが 500 回動きます。テストプログラムの複数インスタンを起動することもできます。必ずテストプログラムと同じフォルダに mxnetapp.ini 設定ファイルを置いてください。

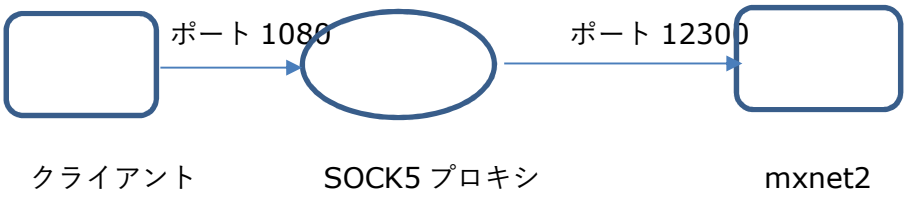
8.2 C# Sample

リモート API DLL を InteropServices でインポートしています。
プラットフォームターゲットを AnyCPU にすると .NET プログラムは同じプログラムでも 32 ビット OS では 32 ビットプログラム、64 ビット OS では 64 ビットプログラムとして動作します。この時インポートされるリモート API DLL は動作するプログラムと同じビットでなければなりません。64 ビット OS で 64 ビットプログラムとして動作している場合は、64 ビット DLL、32 ビット OS で 32 ビットプログラムとして動作している場合は、32 ビット DLL がロードされるようになっていなければなりません。

曖昧さを避けるには、明示的にプラットフォームターゲットを x86 / x64 に設定してください。

9 プロキシサーバ経由接続

クライアント API は SOCK5 対応プロキシサーバ経由で mxnet2 サーバに接続できます。



クライアント AP の設定ファイル(mxnetapi.ini) 内に [SOCKSSERVER]セクションを作成してください。このセクションにプロキシサーバの IP アドレスとポートを設定します。.

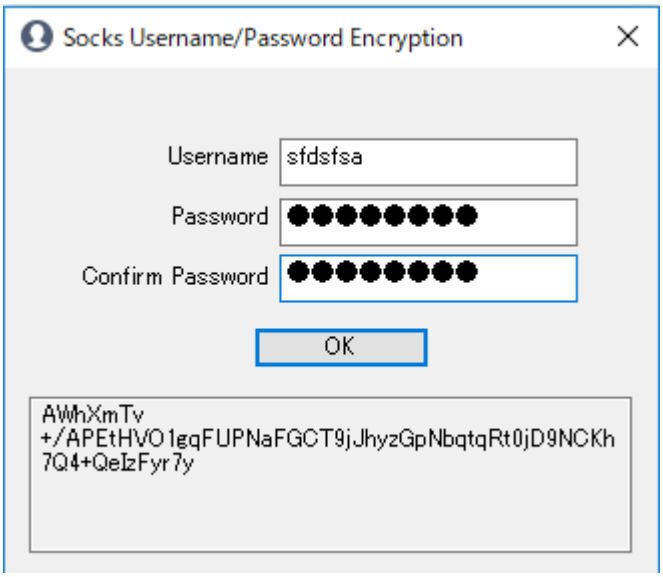
```
[SocksServer]
＃SOCKS5 プロキシ IP
IP=127.0.0.1
#SOCKS5 プロキシポート
Port=1080

[Options]
ip=127.0.0.1
port=12300
```

SOCKS5 がユーザ・パスワード認証を求める場合、UserPass キーにユーザ名とパスワードを暗号化して文字列を指定してください。

```
[SocksServer]
＃SOCKS5 プロキシ IP
IP=127.0.0.1
#SOCKS5 プロキシポート
Port=1080
UserPass=AWhXmTv+/APEtHVO1gqFUPNaFGCT9j……
```

ユーザ名とパスワードの暗号化は添付の userpass.exe を使って生成してください。



SOCK5 プロキシサーバのユーザ名とパスワードを入力後、[OK]ボタンでウィンドウ下側に暗号化文字列が表示されます。それをコピーして、userpass キーに設定してください。

クライアント内部で発生するエラー/SOCK5 関連エラー

エラー番号	説明
-200	SOCKS バージョンエラー
-201	SOCKS は認証をサポートしない
-202	SOCKS 認証エラー
-203	Send エラー
-204	Receive エラー
-205	Socket エラー(WSAEventSelect)
-206	Socket エラー(WSAEnumNetworkEvents)