

Windows API ファイル

Intel 32bit & 64bit

DLL

Mxnet2_api.dll

.NET Framework 用(mxnet2DotNet.dll)

.NET 用(mxnet2DotNetCore.DLL)

LIB

スタティックライブラリ

MT 版(VC++ランタイム不要)

mxnet2st_mt2015

mxnet2st_mt2017

mxnet2st_mt2019

mxnet2st_mt2022

MD 版(VC++ランタイム要)

mxnet2st_md2015

mxnet2st_md2017

mxnet2st_md2019

mxnet2st_md2022

ARM64

[Lib]

DLL

Mxnet2_api.dll

.NET Framework 用(mxnet2DotNet.dll)

.NET 用(mxnet2DotNetCore.DLL)

LIB

スタティックライブラリ

MT 版(VC++ランタイム不要)

mxnet2st_mt2022

MD 版(VC++ランタイム要)

mxnet2st_md2022

[Test-prog]

Testc.exe

Test-gui.exe

設定ファイル mxnetapi.ini

Windows DLL/LIB は Mxnet2 サーバの IP アドレス/ポート、SOCKS サーバのアドレス/ポートを設定ファイル mxnetapi.ini から読み込みます。

API は以下の順番で設定ファイルを探します

DLL と同じフォルダの mxnetapi.ini

LIB ファイルをリンクした実行ファイルと同じフォルダの mxnetapi.ini

見つからなければ

ProgramData¥RiBiG¥mxnet2 フォルダの mxnetapi.ini

OPTION セクションで MxNet2 サーバのアドレスとポートを指定します

[Option]

IP=AAA.BBB.CCC.DDD

Port=12300

SOCK5 サーバを利用するには SocksServer セクションで SOCKS サーバのアドレスとポートを指定します。

[SocksServer]

IP=SSS.TTT.YYY.ZZZ

Port=1080

UserPass=

SOCKS5 サーバの資格情報は UserPass キーに暗号化文字列としてセットします(対応している認証方式はプレーンテキストのみ)。 UserPass.exe でユーザ名とパスワードを指定して暗号化文字列を生成します。

Socksユーザパスワード暗号化

ユーザ名 user1

パスワード ●●●●●●●●

パスワード確認 ●●●●●●●●

了解

fxS90i3BnjTsw1CjPRWjcwABOHT/fJiSpYZRZhBG0i3Me
PiYQQQT8YLbQbD1xGPJd2MyLS0098j6tLf686fWe/VRcx
U1kOIY
+UyttbvKfIRcKDwqI0SVFgX9loaVmkIY6/W32nvfVoovBH
U1PqdtBieMIzd8E3RyDm/DvgFHSJhIVNtkHc/ZpDYZGJ
hF+zeIBzim2yzGlzWqpH60ddwi9of/Y

サンプル

サンプルプログラムは、リモートライセンス管理に必要な以下の処理を行います。

1. 起動時に MxNet2 サーバからライセンスを取得
2. その後、定期的にライセンスを更新しつづける

この処理にはそれぞれのユーザに依存する 3つの値を正しく設定する必要があります。

1. ユーザコード(USERCODE)
MxNet2 サーバには、ユーザ毎に異なるユーザコードが割り当てられています。ただしユーザコードを指定して API を呼び出さなければなりません。
2. 最大ライセンス数の保管場所(APPSLOT)
ライセンス数は、USB キーの 複数ある AppSlot にセットします。いずれかの AppSlot を指定してライセンスを取得しなければなりません。
3. 更新時間間隔 (INTERVAL)
どの程度頻繁に更新をするのか指定します。

サンプルでは、これらの値は、USERCODE, APPSLOT,INTERVAL という変数で指定します。

C サンプル(Visual Studio 2022 プロジェクト)

処理内容

起動後、MxNet2 サーバに接続してライセンスを取得、

その後、バックグラウンドでライセンス更新。

フォアグラウンドではキー入力待ち

test.cpp

```
const int    USERCODE = 1234;
```

```
const short APPSLOT = 7;
```

```
const DWORD INTERVAL = 5000; //ミリ秒単位
```

変更してください。

出来上がった EXE と同じフォルダに設定ファイル mxnetapi.ini を
配置してください。このファイルからサーバの IP/Port を読み込みます。

MxNet2 API スタティックライブラリは MT Release 版です。

サンプルも MT Release でビルドしてください。

Debug でビルドするとエラーになります。

C++ サンプル(Visual Studio 2022 プロジェクト)

処理内容

起動後、MxNet2 サーバに接続してライセンスを取得、

その後、バックグラウンドでライセンス更新。

フォアグラウンドではキー入力待ち

他サンプルに比べ、以下の追加処理のため複雑になっています。

- スレッドの使いまわし
- エラー時の接続解放

test.cpp

```
const int USERCODE = 1234;  
const short APPSLOT = 7;  
const int INTERVAL = 5; // 秒単位
```

変更してください

出来上がった EXE と同じフォルダに設定ファイル mxnetapi.ini を配置してください。このファイルからサーバの IP/Port を読み込みます。

MxNet2 API スタティックライブラリは MT Release 版です。

サンプルも MT Release でビルドしてください。

Debug でビルドするとエラーになります。

同じソースコードは Linux でもビルドできます。

```
g++ -o test test.cpp loginupdate.cpp  
-l mxnet2api -l pthread -l dl
```

C-Sharp-DotNetFramework(Visual Studio 2022)

処理内容

.Net Framework フォームで3つのスレッドで同時に API 呼び出し

Form1.cs

```
const int MYUSERCODE = 1234;
```

```
const short APPSLOT = 5;
```

変更してください。ライセンス更新をしませんので、INTERVAL は不要です

出来上がった EXE と同じフォルダに設定ファイル mxnetapi.ini を
配置してください。このファイルからサーバの IP/Port を読み込みます。

32bit 版 / 64bit 版 DLL のいずれかを呼び出さなければなりません。
参照設定で正しい DLL をセットしてください。プログラム本体と
DLL のビット数は一致していなければなりません。

WinFormsApp-DotNet(Visual Studio 2022)

処理内容

Net8(Net Core) フォームが開くときに MxNet2 サーバに接続してライセンスを取得、その後、バックグラウンドでライセンス更新。起動時、ライセンスを取得できなければ、プログラムを終了します。

Form1.cs

```
const int MYUSERCODE = 1234;
```

```
const short APPSLOT = 7;
```

```
const int INTERVAL = 5000; //ミリ秒単位
```

変更してください。

プロジェクトでは 64 ビット版 DLL を参照しています。

出来上がった EXE と同じフォルダに設定ファイル mxnetapi.ini を
配置してください。このファイルからサーバの IP/Port を読み込みます。

32bit 版 / 64bit 版 DLL のいずれかを呼び出さなければなりません。
参照設定で正しい DLL をセットしてください。プログラム本体と
DLL のビット数は一致していなければなりません。

Java

フォルダ内の readmd.txt をご覧ください。

API ファイルを正しく配置することで、同一ソースコードで Windows / Linux
/ macOS に対応します。

Excel

処理内容

Excel 用 DLL を呼び出すことで、MxNet2 サーバに接続してライセンスを取
得、その後、バックグラウンドでライセンス更新。

フォルダ内の readmd.txt をご覧ください