

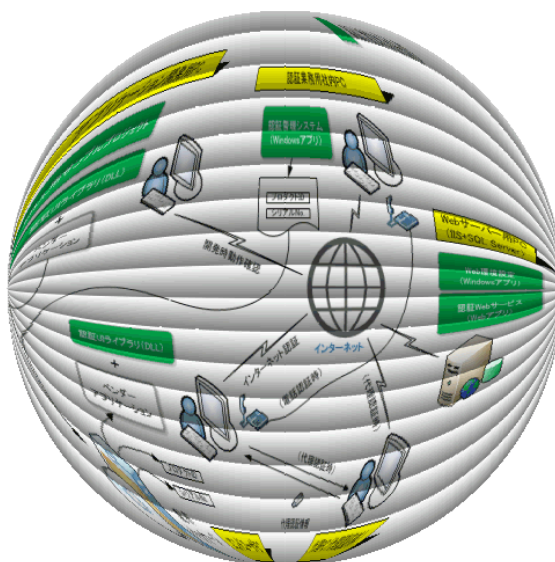
ライセンス認証(アクティベーション) 実装ソリューション

- ◆ 認証 Web サービス
- ◆ 認証管理アプリケーション
- ◆ 配布アプリケーション用認証ライブラリ(DLL)

認証レスキュー! 2

ユーザーズガイド

(2.6.8)



| | |
|------------------------------------------------------------------------------|----|
| ●概要..... | 7 |
| ●主な注目機能一覧..... | 8 |
| ●インストールされる内容..... | 9 |
| ●動作環境..... | 10 |
| ●運用イメージ..... | 11 |
| ●処理・設定一覧..... | 12 |
| ●インストール..... | 13 |
| 1.「Web サーバー用 PC」へのインストール..... | 13 |
| 2.「認証業務用社内 PC」へのインストール..... | 15 |
| 3.「アプリケーション開発用 PC」へのインストール..... | 16 |
| 4.インストールの終了後..... | 16 |
| 5.製品別の制限事項..... | 17 |
| ●初期設定(Web サーバー用 PC と認証業務用社内 PC)..... | 18 |
| 1.Web サーバー用 PC の「環境設定」処理..... | 18 |
| ■データテーブル新規作成..... | 19 |
| ■NR 登録ライセンスの管理..... | 21 |
| ■NR 登録ライセンスの警告..... | 24 |
| ■「登録」ボタン..... | 25 |
| ■データベース更新..... | 27 |
| ■データベースのバックアップ..... | 27 |
| ■データベースの復元..... | 27 |
| 2.認証業務用社内 PC の「認証管理システム」..... | 28 |
| ■環境設定..... | 28 |
| ■認証キー作成(自動ナンバリング)..... | 30 |
| ■ラベル印刷..... | 32 |
| ●Microsoft Azure で「認証レスキュー！」を利用する方法..... | 35 |
| ■Microsoft Azure の仮想マシンを利用する..... | 36 |
| 1.仮想マシンの作成(参考)..... | 36 |
| 2.作成した仮想マシン(例)..... | 37 |
| 3.仮想マシンの開始(例)..... | 37 |
| 4.仮想マシンの接続(例)..... | 38 |
| 5.仮想マシンへの認証レスキュー！のインストール..... | 39 |
| ■Microsoft Azure の Web アプリ(旧 Web サイト)と SQL データベースを利用する..... | 40 |
| 1.認証レスキュー！の「Web サーバー用 PC」へのインストールを行う..... | 41 |
| 2.認証レスキュー！用のデータベースを Microsoft Azure の「SQL データベース」に配置する..... | 42 |
| 3.認証レスキュー！の Web サーバー用 PC の「環境設定」処理を完了する..... | 45 |
| 4.Microsoft Azure の「Web アプリ」に認証レスキュー！の Web サービスを配置する..... | 47 |
| 5. Microsoft Azure 上で認証レスキュー！を運用している場合の Web アプリと SQL データベースのバージョンアップの方法..... | 48 |
| ■Azure でのシステム標準時刻に関する注意点..... | 50 |
| ●アプリケーション開発用 PC での「認証 UI ライブラリ」の利用..... | 52 |
| ■認証 UI ライブラリ関連ファイル..... | 52 |
| ■認証 UI ライブラリの利用形態による選択..... | 53 |
| ■同一 PC 内で異なるアプリケーションやオプションのライセンスを識別する方法..... | 54 |
| ■UI 系サンプルプロジェクトと API 系サンプルプロジェクトについて..... | 56 |
| ■ASP.NET 系サンプルプロジェクトについて..... | 56 |
| ■認証 UI ライブラリ機能一覧..... | 57 |
| <クラス>..... | 57 |
| <UI 系プロパティ>..... | 58 |

| | |
|------------------------------------------------------|-----|
| EncryptionPassword プロパティ..... | 59 |
| EncryptionSaltString プロパティ..... | 60 |
| ProductIdNumberOfDigits プロパティ..... | 61 |
| RentalPeriod プロパティ..... | 62 |
| RentalPeriodName プロパティ..... | 63 |
| SerialNoNumberOfDigits プロパティ..... | 64 |
| SetProductID プロパティ..... | 65 |
| SetSerialNo プロパティ..... | 66 |
| TelephoneNumber プロパティ..... | 67 |
| TrialPeriod プロパティ..... | 68 |
| TrialPeriodName プロパティ..... | 69 |
| UseCpuInfo プロパティ..... | 70 |
| UseMacAddress プロパティ..... | 71 |
| VendorsProductStartRegistryKeyPath プロパティ..... | 72 |
| WebServiceBasicAuthenticationPassword プロパティ..... | 73 |
| WebServiceBasicAuthenticationUserName プロパティ..... | 74 |
| WebServiceCheckPassword プロパティ..... | 75 |
| WebServiceTimeout プロパティ..... | 76 |
| WebServiceURL プロパティ..... | 77 |
| WebServiceUseBasicAuthentication プロパティ..... | 78 |
| <UI 系メソッド>..... | 79 |
| ActivateRegisterInternet メソッド..... | 80 |
| ActivateRegisterTelephone メソッド..... | 82 |
| ActivateRemoveInternet メソッド..... | 83 |
| ActivateRemoveTelephone メソッド..... | 85 |
| ActivateStatusCheck メソッド..... | 86 |
| ActivateStatusCheckOnline メソッド..... | 88 |
| ActivateStatusDisp メソッド..... | 90 |
| DeterminationOfProxyUpdateOfExpirationDate メソッド..... | 91 |
| GetProxyUpdateOfExpirationDate メソッド..... | 92 |
| PreparationOfProxyUpdateOfExpirationDate メソッド..... | 93 |
| ProxyActivateRegisterExecute メソッド..... | 95 |
| ProxyActivateRegisterFix メソッド..... | 96 |
| ProxyActivateRegisterPrepare メソッド..... | 97 |
| ProxyActivateRemoveExecute メソッド..... | 98 |
| ProxyActivateRemovePrepare メソッド..... | 99 |
| RestoreCancelStatus メソッド..... | 100 |
| RestoreRegisterStatus メソッド..... | 101 |
| UpdateOfExpirationDate メソッド..... | 102 |
| <API 系プロパティ>..... | 104 |
| APIError 列挙体..... | 105 |
| APICertificationID プロパティ..... | 125 |
| APICurrentExpirationDate プロパティ..... | 126 |
| APIEncryptionPassword プロパティ..... | 127 |
| APIEncryptionSaltString プロパティ..... | 128 |
| APIErrorStatus プロパティ..... | 129 |
| APIExternalLinkKey プロパティ..... | 130 |
| APINewExpirationDate プロパティ..... | 131 |
| APIFreeItem1~5 プロパティ..... | 132 |
| APILicenseKey プロパティ..... | 133 |
| APIOverwriteModeOfExpirationDateUpdate プロパティ..... | 134 |

| | |
|----------------------------------------------------------|-----|
| APIProductID プロパティ | 135 |
| APIProductIdSerialNoList プロパティ | 136 |
| APIProxyDataPath プロパティ | 137 |
| APIProxyServerAddress プロパティ | 138 |
| APIProxyServerPassword プロパティ | 139 |
| APIProxyServerPort プロパティ | 140 |
| APIProxyServerUserName プロパティ | 141 |
| APIReleaseKey プロパティ | 142 |
| APIReleaseStatus プロパティ | 143 |
| APIRentalPeriod プロパティ | 144 |
| APISelectRunAppDatePathFlag プロパティ | 145 |
| APISerialNo プロパティ | 146 |
| APITrialPeriod プロパティ | 147 |
| APIUseCpuInfo プロパティ | 148 |
| APIUseMacAddress プロパティ | 149 |
| APIUseProxyServer プロパティ | 150 |
| APIVendorsProductStartRegistryKeyPath プロパティ | 151 |
| APIWebServiceBasicAuthenticationPassword プロパティ | 152 |
| APIWebServiceBasicAuthenticationUserName プロパティ | 153 |
| APIWebServiceCheckPassword プロパティ | 154 |
| APIWebServiceTimeout プロパティ | 155 |
| APIWebServiceURL プロパティ | 156 |
| APIWebServiceUseBasicAuthentication プロパティ | 157 |
| <API系メソッド> | 158 |
| APIActivateRegisterInternet メソッド | 160 |
| APIActivateRegisterTelephone メソッド | 163 |
| APIActivateRemoveInternet メソッド | 166 |
| APIActivateRemoveTelephone メソッド | 169 |
| APIActivateStatusCheck メソッド | 172 |
| APIActivateStatusCheck2 メソッド | 174 |
| APIActivateStatusCheckOnline メソッド | 176 |
| APIActivateStatusCheckOnline2 メソッド | 178 |
| APIDeterminationOfProxyUpdateOfExpirationDate メソッド | 180 |
| APIGenerationOfNewCertificationID メソッド | 183 |
| APIGetFreeItem メソッド | 185 |
| APIGetProductIdSerialNoList メソッド | 187 |
| APIGetProxyDataForExpirationDate メソッド | 189 |
| APIGetProxyDataForRegister メソッド | 191 |
| APIGetProxyDataForRemove メソッド | 193 |
| APIGetProxyUpdateOfExpirationDate メソッド | 195 |
| APIGetRegisteredInfoFromRegistry メソッド | 198 |
| APIGetRegisteredInfoFromRegistry2 メソッド | 200 |
| APIGetRegisteredInfoFromWeb メソッド | 202 |
| APIPreparationOfProxyUpdateOfExpirationDate メソッド | 205 |
| APIProxyActivateRegisterExecute メソッド | 207 |
| APIProxyActivateRegisterFix メソッド | 210 |
| APIProxyActivateRegisterPrepare メソッド | 213 |
| APIProxyActivateRemoveExecute メソッド | 215 |
| APIProxyActivateRemovePrepare メソッド | 218 |
| APIReadProxyServerInfoFromRegistry メソッド | 220 |
| APIRestoreCancelStatus メソッド | 222 |

| | |
|----------------------------------------------------|-----|
| APIRestoreRegisterStatus メソッド | 225 |
| APIRunNR2AppDateRemove メソッド | 227 |
| APIRunNR2AppDateRemove2 メソッド | 228 |
| APITrialStartDateRemove メソッド | 229 |
| APIUpdateOfExpirationDate メソッド | 230 |
| APIWriteProxyServerInfoToRegistry メソッド | 233 |
| <ASP.NET 系プロパティ> | 235 |
| APIxError 列挙体 | 236 |
| APIxErrorStatus プロパティ | 245 |
| APIxExpirationDate プロパティ | 246 |
| APIxExternalLinkKey プロパティ | 247 |
| APIxFreeItem1~5 プロパティ | 248 |
| APIxKindOfRandom プロパティ | 249 |
| APIxLicenseCount プロパティ | 250 |
| APIxNumberingCount プロパティ | 251 |
| APIxProductID プロパティ | 252 |
| APIxProxyServerAddress プロパティ | 253 |
| APIxProxyServerPassword プロパティ | 254 |
| APIxProxyServerPort プロパティ | 255 |
| APIxProxyServerUserName プロパティ | 256 |
| APIxSerialNo プロパティ | 257 |
| APIxSerialNoString プロパティ | 258 |
| APIxStartNo プロパティ | 259 |
| APIxStartFixedString プロパティ | 260 |
| APIxStepNo プロパティ | 261 |
| APIxUseProxyServer プロパティ | 262 |
| APIxUseRental プロパティ | 263 |
| APIxUseExpirationDate プロパティ | 264 |
| APIxWebServiceBasicAuthenticationPassword プロパティ | 265 |
| APIxWebServiceBasicAuthenticationUserName プロパティ | 266 |
| APIxWebServiceCheckPassword プロパティ | 267 |
| APIxWebServiceTimeout プロパティ | 268 |
| APIxWebServiceURL プロパティ | 269 |
| APIxWebServiceUseBasicAuthentication プロパティ | 270 |
| <ASP.NET 系メソッド> | 271 |
| APIxCreateNumberingActivationKey メソッド | 272 |
| APIxCreateOneActivationKey メソッド | 274 |
| APIxCreateRandomActivationKey メソッド | 276 |
| APIxDeleteActivationKey メソッド | 278 |
| APIxEditOfExpirationDate メソッド | 280 |
| 代理認証機能について | 282 |
| ■ 認証 UI ライブラリの参照 | 284 |
| Visual Studio 2010 (Visual Basic 2010/C# 2010) の場合 | 284 |
| Visual Studio 6.0 (Visual Basic 6.0) の場合 | 284 |
| Visual C++ の場合 | 287 |
| ■ 認証 UI ライブラリ (DLL) を利用したコーディング <UI 系の場合> | 289 |
| Visual Basic 2010 の場合 | 289 |
| C# 2010 の場合 | 294 |
| Visual Studio 6.0 (Visual Basic 6.0) の場合 | 300 |
| Visual C++ の場合 | 305 |
| ■ 認証 UI ライブラリ (DLL) の配布 | 312 |

| | |
|---------------------------------------------|-----|
| ● 認証レスキュー！で使う主なテーブルの概要 | 313 |
| < 認証キーテーブル > | 313 |
| < 認証データテーブル > | 313 |
| < 認証キーテーブル > の「プラス許可数」項目の必要性 | 315 |
| < 認証ログテーブル > | 316 |
| ● 「認証管理システム」のその他の処理説明 | 319 |
| ■ 認証キー作成（表形式） | 320 |
| ■ 認証キー作成（個別） | 321 |
| ■ 認証キー作成（ランダム生成） | 322 |
| ■ 認証キー作成（インポート） | 323 |
| ■ 認証キー編集（表形式） | 324 |
| ■ 認証キー削除（表形式） | 326 |
| ■ 認証キー削除（個別） | 327 |
| ■ 認証状況 | 328 |
| ■ ログの表示 | 330 |
| ■ 電話認証登録の対応 | 332 |
| ■ 電話認証解除の対応 | 333 |
| ● 有効期限機能の利用方法 | 336 |
| ◆ Web アプリケーション (ASP.NET) から有効期限の更新を行う | 340 |
| ● アプリケーション難読化の必要性 | 341 |

●概要

「認証レスキュー！2(以降、認証レスキュー！)」は、貴社のパッケージアプリケーションにライセンス認証(アクティベーション)機能を付加して運用するためのソリューションです。認証レスキュー！では、次の3つのソリューションを提供します。

1.Web サーバー用 PC に対するソリューション

インストーラが SQL Server 2012 Express のインストールや認証レスキュー！用のデータベースの設定、IIS の設定を行い、認証 Web サービス、環境設定をインストールします。稼働後は認証 Web サービスが常時動作し、インターネットを経由したお客様(エンドユーザ) PC 上の貴社アプリケーションからの認証処理のリクエストに自動的に応答します。また、同様にインターネット経由で、貴社の認証業務用社内 PC での認証管理システムからの認証状況の確認や新しいパッケージ製品の認証キーの作成などのリクエストも処理します。

2.認証業務用社内 PC に対するソリューション

社内 PC で使用する「認証管理システム」をインストールします。認証管理システムでは出荷前製品の認証キーの作成やプロダクト ID やシリアル No.のラベル印刷、お客様(エンドユーザ)がインターネット経由で認証登録した記録などを閲覧できます。

3.「アプリケーション開発用 PC」へのソリューション

アプリケーションに認証登録用 UI(ユーザインターフェース)の機能を付加するための認証 UI ライブラリ(DLL)とサンプルプログラムをインストールします。この認証 UI ライブラリ(DLL)を利用することで貴社のアプリケーションにライセンス認証(アクティベーション)機能を簡単に実装することができます。

この「認証レスキュー！」を導入することで驚くほど早く確実に、貴社パッケージアプリケーションにライセンス認証機能を実装でき、ライセンス不正利用による損害を防止できます！

●主な注目機能一覧

◆従来機能

- ・エンドユーザのプロキシサーバー環境に対応
- ・PC クラッシュ時の認証解除機能
- ・オフライン時電話認証機能

◆拡張機能

- ・認証 UI ライブラリ(DLL)の提供
- ・Web サービス用「環境設定」アプリの提供
- ・認証業務用「認証管理システム」アプリの提供
- ・マイクロソフト社クラウド Microsoft Azure 対応
- ・代理認証機能
- ・レンタル機能
- ・試用期間機能
- ・有効期限機能
- ・プロダクト ID 桁数自由設定機能
- ・シリアル No.桁数自由設定機能
- ・MAC アドレス識別情報付加
- ・CPU 情報識別情報付加
- ・認証状況の Excel データ出力
- ・ログ表示の Excel データ出力
- ・認証状態オンライン確認機能
- ・認証状況およびログにエンドユーザ IP アドレスを記録

◆便利機能

- ・各種データ閲覧時の検索条件設定機能
- ・認証キー自動ナンバリング機能
- ・認証キー一覧編集機能
- ・データベースのバックアップ(スケジュール化可)および、復元機能
- ・サンプルデータベース切替お試し機能
- ・DLL 用サンプルプロジェクトの提供

◆セキュリティ関連

- ・環境設定でのログインダイアログ指定機能
- ・DLL での貴社独自の暗号化情報設定機能
(認証レスキュー！2 を利用した他社とは別の暗号化)
- ・DLL はアセンブリ署名により偽装対策済み
- ・DLL は逆コンパイル対策の難読化済み
- ・SQL Server の SQL インジェクション対策済み(Web サービス)
- ・Web サービスのブラウザ表示隠ぺい化

●インストールされる内容

| | | |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------|
| Web サーバー用 PC | <ul style="list-style-type: none"> ・認証 Web サービス (Web アプリケーション) ・Web 環境設定 (Windows アプリケーション) ・SQL Server 2012 Express + Management Studio ・体験版ライセンス (10 登録ライセンス) ※製品版の基本パックに付属する「500 登録ライセンス」はインストールとは別の形で提供されます。 | ユーザーズ ガイドなど |
| 認証業務用 PC | <ul style="list-style-type: none"> ・認証管理システム (Windows アプリケーション) | |
| アプリケーション 開発用 PC | <ul style="list-style-type: none"> ・認証 UI ライブラリ (DLL)、同タイプライブラリ (TLB) ・サンプルプロジェクト (UI 系サンプル、API 系サンプル) Visual Basic 2010 用、C# 2010 用、Visual Basic 6.0 用 、Visual C++ 2010 用 | |

●動作環境

・「Web サーバー用 PC」へのインストールが対応している OS

日本語 Microsoft Windows Server 2022/2019/2016/2012 R2/2012/2008 R2/2008
 日本語 Microsoft Windows 11(Home は除く)/10(Home は除く)/8.1(Pro 以上)/8(Pro 以上)/7(Home Basic は除く)/Vista(Home Basic は除く)各 x86、x64 対応
 ※上記 Windows OS 上の対応暦:グレゴリオ暦(西暦)のみ

・「Web サーバー用 PC」へのインストールが対応している IIS

IIS 10.0/8.5/8.0/7.5/7.0

・「認証業務用社内 PC」へのインストールによる「認証管理システム」が対応している OS

日本語 Microsoft Windows Server 2022/2019/2016/2012 R2/2012/2008 R2/ 2008
 日本語 Microsoft Windows 11/10/8.1/8/7/Vista(x86、x64 対応)
 日本語 Microsoft Windows Server 2003 R2/2003
 日本語 Microsoft Windows XP(x86、x64 対応)
 ※上記 Windows OS 上の対応暦:グレゴリオ暦(西暦)のみ

・「アプリケーション開発用 PC」へのインストールによる「認証 UI ライブラリ(DLL)」が対応している OS

Microsoft Windows Server 2022/2019/2016/2012 R2/2012/2008 R2/2008
 Microsoft Windows 11/10/8.1/8/7/Vista(x86、x64 対応)
 Microsoft Windows Server 2003 R2/2003
 Microsoft Windows XP(x86、x64 対応)

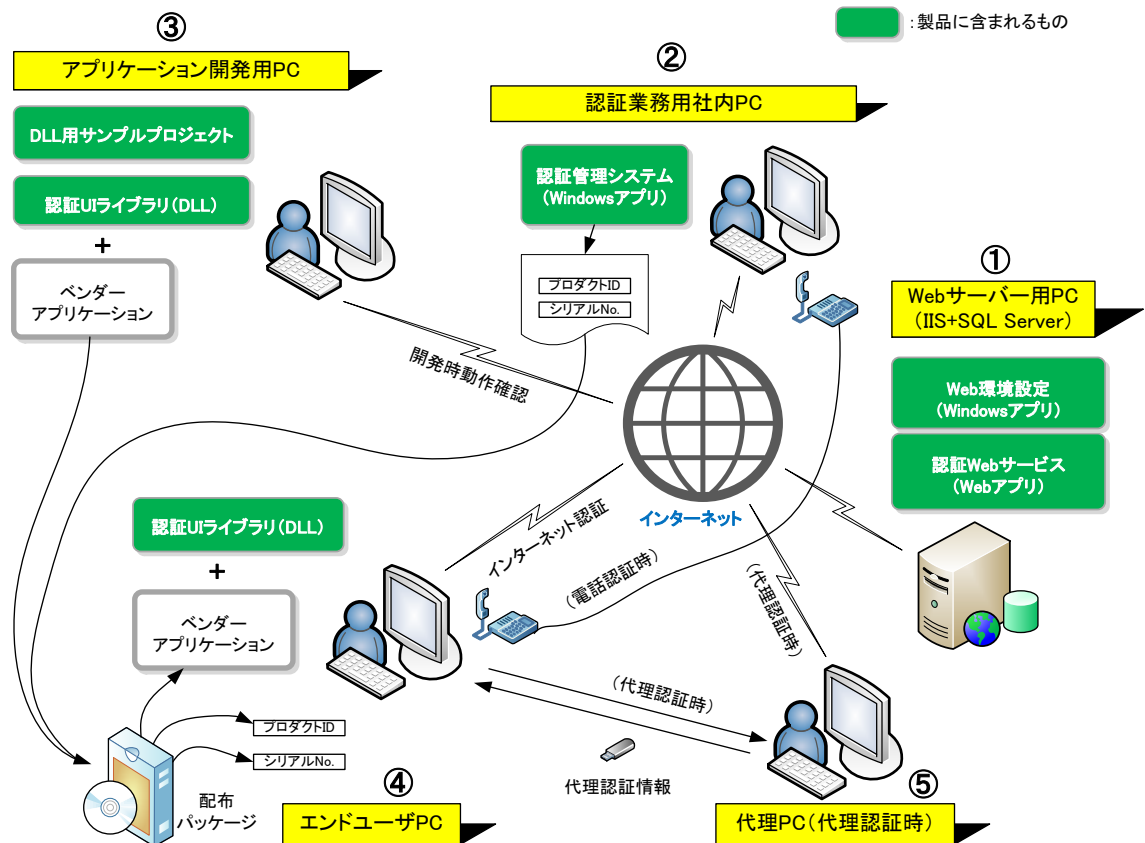
[DLL による OS および暦対応表]

| DLL | 【UI 系、API 系 DLL】 Newtone.NR.dll | 【ASP.NET 系 DLL】 Newtone.NR.ASPNET.dll |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|
| Windows 対応 OS | 日本語版および英語版推奨 | 日本語版のみ |
| 対応 OS 上の対応暦 | <ul style="list-style-type: none"> ・ChineseLunisolarCalendar/中国の太陰太陽暦 ・GregorianCalendar/グレゴリオ暦(西暦) ・HebrewCalendar/ヘブライ暦 ・HijriCalendar/回教暦 ・JapaneseCalendar/和暦 ・JapaneseLunisolarCalendar/日本の太陰太陽暦 ・JulianCalendar/ユリウス暦 ・KoreanCalendar/韓国暦 ・KoreanLunisolarCalendar/韓国の太陰太陽暦 ・PersianCalendar/ペルシャ暦 ・TaiwanCalendar/台湾暦 ・TaiwanLunisolarCalendar/台湾の太陰太陽暦 ・ThaiBuddhistCalendar/タイ仏暦 ・UmAlQuraCalendar/ウムアルクラ暦 | <ul style="list-style-type: none"> ・GregorianCalendar/グレゴリオ暦(西暦)のみ |

・認証 UI ライブラリ(DLL)を利用するための開発環境(すべて日本語版のみ)

Visual Studio 2022/2019/2017/2015/2013/2012/2010/2008、Visual Studio 6.0

●運用イメージ



主な運用手順は次の通りです。具体的な手順は、後述します。(丸付き数字は上図の各 PC です。)

1. 認証レスキュー！のインストールを行う

①Web サーバー用 PC、②認証業務用社内 PC、③アプリケーション開発用 PC にそれぞれインストールを行います。

2. インストールしたアプリケーションを使用して必要な各設定を行う

①Web サーバー用 PC と②認証業務用社内 PC で設定します。

3. インストールしたアプリケーションを使用して必要な各処理を行う

パッケージソフト出荷のための認証キー(プロダクト ID やシリアル No.など)を作成し、プロダクト ID とシリアル No.ラベル印刷を行います。②認証業務用社内 PC が対象です。

4. インストールした認証 UI ライブラリ(DLL)を利用し貴社アプリケーションに認証機能を実装する

③アプリケーション開発用 PC が対象です。

5. 貴社アプリケーションをお客様(エンドユーザ)へ配布する

④エンドユーザ PC が対象です。上記 4 で完成した貴社アプリケーションと上記 3 で発行したプロダクト ID とシリアル No.をお客様(エンドユーザ)へ配布します。

6. お客様(エンドユーザ)がライセンス認証を実行する

④エンドユーザ PC または⑤(エンドユーザの)代理 PC で、お客様(エンドユーザ)が貴社アプリケーションの認証 UI を使用してライセンス認証を行い、その内容が貴社 Web サーバーなどの認証レスキュー！用のデータベースに記録されます。

その内容は、②認証業務用社内 PC の「認証状況」処理などで確認できます。

●処理・設定一覧

認証レスキュー！では運用に必要な各処理の他に、ライセンス認証に関連する細かな設定が指定できます。

認証レスキュー！で利用できる処理や設定の一覧を示します。

◆Web サーバー用 PC

- ・Web サービスの設定
- ・環境設定へのログイン設定
- ・データベースの指定
 - NR2 規定/NR2 サンプルデータ
 - /任意接続文字列(マイクロソフト社クラウド Microsoft Azure など)
- ・データテーブル新規作成処理
- ・データベース更新処理
- ・NR 登録ライセンスの管理
- ・データベースのバックアップ処理(スケジュール化可能)
- ・データベースの復元

◆認証業務用社内 PC 「認証管理システム」

- ・Web サービスのアクセス設定
- ・プロキシサーバーの設定
- ・環境設定へのログイン設定
- ・認証キー作成(自動ナンバリング)処理
- ・認証キー作成(表形式)処理
- ・認証キー作成(個別)処理
- ・認証キー編集処理
- ・認証キー削除(表形式)処理
- ・認証キー削除(個別)処理
- ・ラベル印刷処理
- ・認証状況の表示/出力処理
- ・ログの表示処理

◆アプリケーション開発用 PC 「認証 UI ライブラリ(DLL)」の利用

<付属サンプルプロジェクト>

Visual Basic 2010 用(UI系サンプル、API系サンプル、ASP.NET系サンプル)

C# 2010 用(UI系サンプル、API系サンプル、ASP.NET系サンプル)

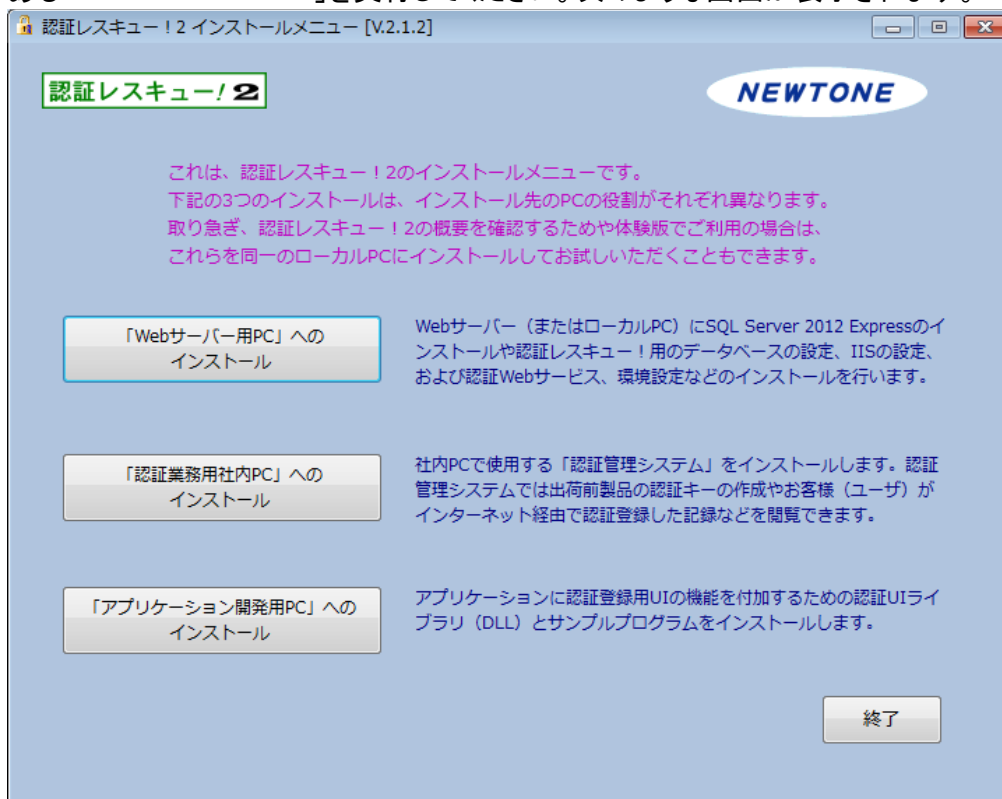
Visual Basic 6.0 用(UI系サンプル、API系サンプル)

Visual C++ 2010 用(UI系サンプル、API系サンプル)

<プロパティ>および<メソッド>の各機能(UI系、API系、ASP.NET系)

●インストール

(パッケージの場合は) ディスク内のルートまたは(ダウンロードなどの場合は) 解凍したフォルダにある「NR2InstallMenu.exe」を実行してください。次のような画面が表示されます。



これは、認証レスキュー！のインストールメニューです。

3つのインストールボタンがあり、インストール先のPCの役割がそれぞれ異なります。

取り急ぎ、認証レスキュー！の概要を確認するためや体験版でご利用の場合は、これらを同一のローカルPCにインストールしてお試しいただくこともできます。

1. 「Webサーバー用PC」へのインストール

Webサーバー(またはローカルPC)にSQL Server 2012 Expressのインストールや認証レスキュー！用のデータベースの設定、IISの設定、および認証Webサービス、環境設定などのインストールを行います。

このインストールを選択するとさらに次のWebインストーラの画面が表示されます。



ここでは、「データベースのインストール」と「IIS 設定と Web サービスのインストール」を行います。

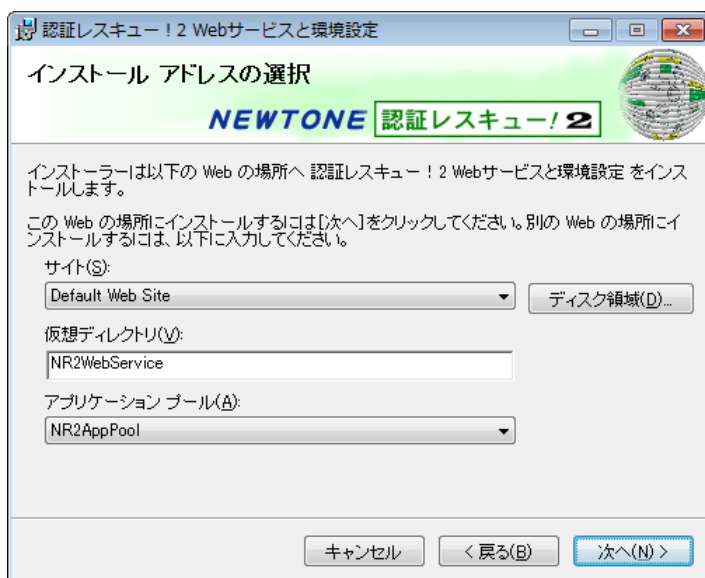
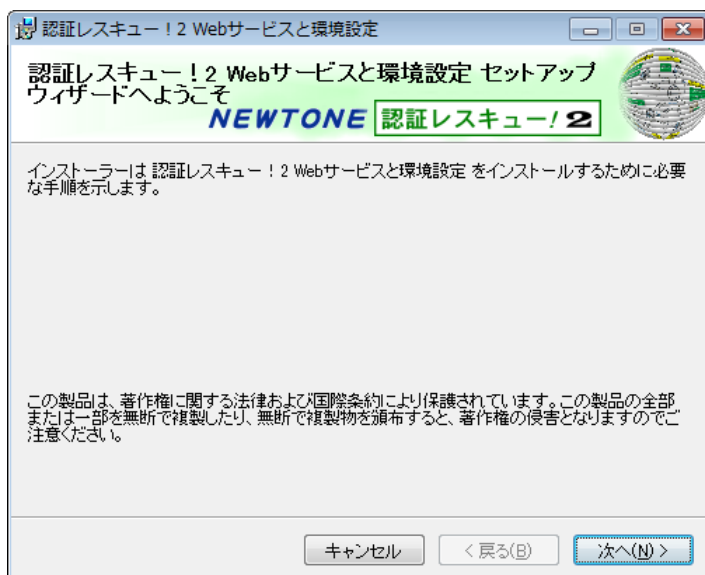
・データベースのインストール

認証レスキュー！用のデータベース(SQL Server Express)のインストールを行います。日本語版か英語版を選択できます。英語の OS にインストールする場合は英語版を選択してください。また、認証レスキュー！用のインスタンスと(データを除く)データベースも作成します。既に同じバージョンの SQL Server Express がインストールされている場合は、認証レスキュー！に必要な設定だけを行います。

・IIS 設定と Web サービスのインストール

認証 Web サービスのインストールを行います。Internet Information Services (IIS) がインストールされている必要がありますが、認証レスキュー！で必要な IIS の機能で不足している機能は自動的にインストールされます。以下で有効になっているボタンを選択するとインストールが始まります。選択できるボタンが無い場合は、現在の OS が対応外の可能性がありますのでご確認ください。

インストールを開始して IIS の設定が済むと確認画面の表示後、次のような「Web サービスと環境設定」のインストーラが起動されます。

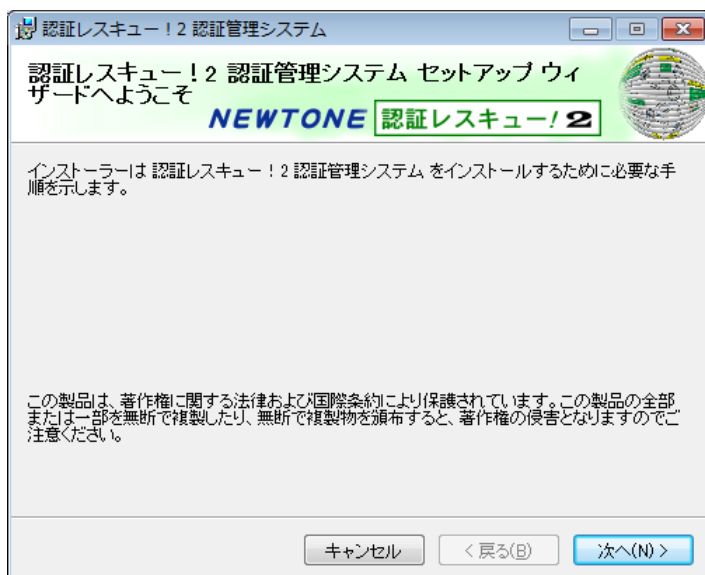


この画面では、サイト、仮想ディレクトリ、アプリケーションプールを指定しますが、通常はデフォルト(初期設定)のまま「次へ」ボタンを押します。以降は画面の指示に従ってください。

マイクロソフト社のクラウドサービス Microsoft Azure の「Web アプリ (旧 Web サイト)」に Web サービスを配置する場合は、一度この「IIS 設定と Web サービスのインストール」をローカル PC にインストールします。その後、ローカル PC の Web サービスのフォルダ内のすべてのファイルと Web サービス環境設定データ(WebServEnv.wai) ファイル(後述)を FTP などを使用して Microsoft Azure の「Web アプリ (旧 Web サイト)」に手動でコピーする必要があります。詳しくは、後述の「Microsoft Azure で利用する方法」をご覧ください。

2. 「認証業務用社内 PC」へのインストール

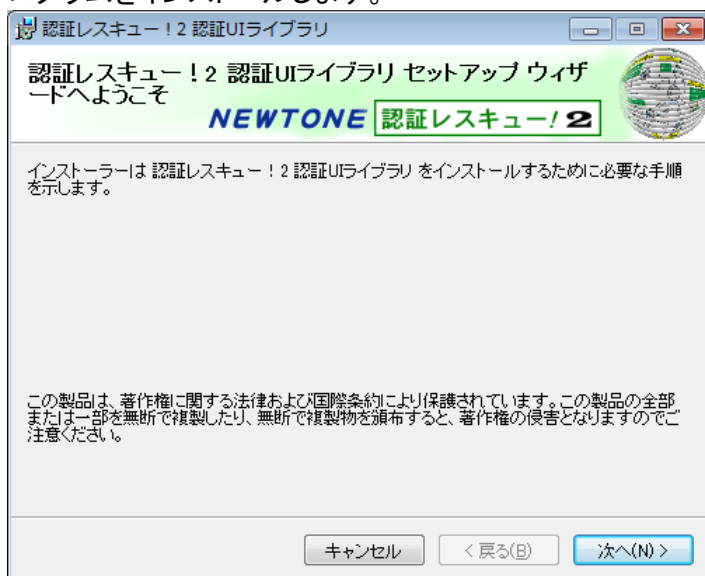
社内 PC で使用する「認証管理システム」をインストールします。認証管理システムでは出荷前製品の認証キーの作成やお客様(エンドユーザ)がインターネット経由で認証登録した記録などを閲覧できます。



インストールするには画面の指示に従ってください。

3. 「アプリケーション開発用 PC」 へのインストール

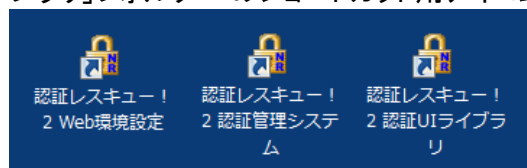
アプリケーションに認証登録用 UI の機能を付加するための認証 UI ライブラリ(DLL)とサンプルプログラムをインストールします。



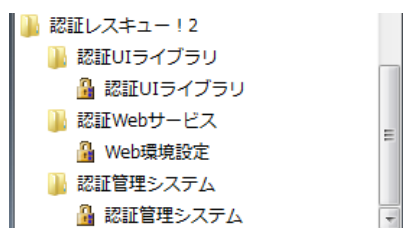
インストールするには画面の指示に従ってください。

4. インストールの終了後

すべてのセットアップが終了すると、デスクトップに「認証レスキュー! Web 環境設定」へのショートカット、「認証レスキュー! 認証管理システム」へのショートカット、「認証レスキュー! 認証 UI ライブラリ」フォルダへのショートカット用アイコンがそれぞれ次のように作成されます。



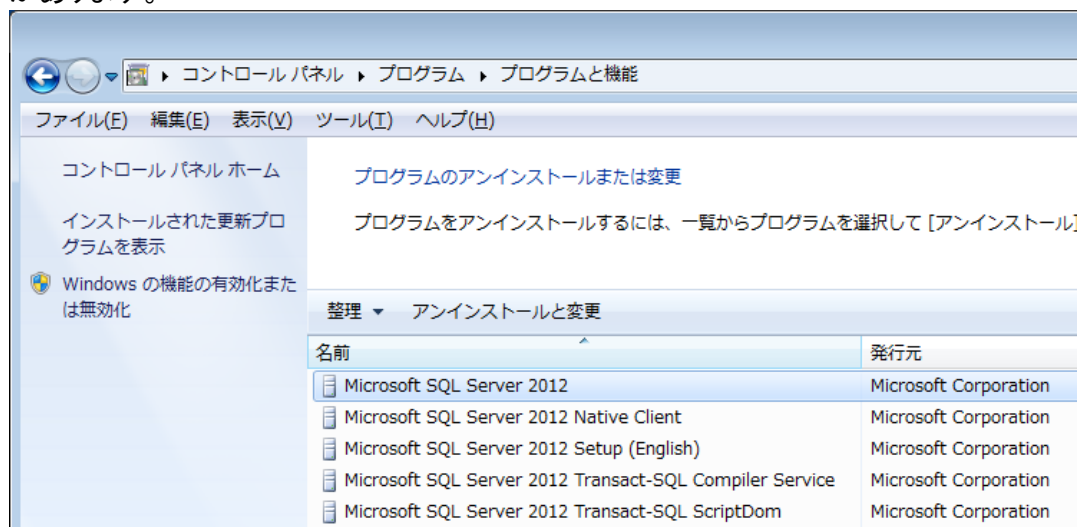
また、プログラムメニューには次のように「認証レスキュー!」が登録されます。



アンインストールは、「プログラムと機能」または「プログラムの追加と削除」から行います。



SQL Server Express 2012 をアンインストールする場合は、「プログラムと機能」または「プログラムの追加と削除」で次の項目をアンインストールします。PC の環境によっては、項目が異なる場合があります。



5.製品別の制限事項

インストールした製品による制限事項は次表の通りです。

| インストーラ | 認証 UI ライブラリ (DLL) の制限 | |
|------------|-----------------------|------------|
| | UI 系機能 | API 系機能 |
| 体験版 | 体験版ダイアログ表示 | 体験版ダイアログ表示 |
| 基本パック | 製品版 | 体験版ダイアログ表示 |
| 基本パック +API | 製品版 | 製品版 |

●初期設定（Web サーバー用 PC と認証業務用社内 PC）

インストールが終了したら、認証レスキュー！を利用する前の各種設定が必要です。

1.Web サーバー用 PC の「環境設定」処理

デスクトップ上の「認証レスキュー！ Web 環境設定」へのショートカットを起動すると次の画面が表示されます。

この実行ファイルは、インストール先がデフォルトなら、

<32bitOS の場合> C:\Program Files\Newtone\NR2\NR2Web\WebAdmin.exe、

<64bitOS の場合> C:\Program Files (x86)\Newtone\NR2\NR2Web\WebAdmin.exe です。

ここでは、設定が必須で省略できない項目についてだけ説明します。

・Web サービス/確認パスワード

Web サービスを利用する場合の確認用のパスワードを設定します。

ここでは、Web サーバー側設定アプリケーション「認証 Web サービス」の環境設定処理の Web サービスの確認パスワードと同じものを設定します。確認パスワードは必須項目です、省略はできません。

8文字以上で半角の次の文字が使用できます。

- ・大文字の英字(A～Z)

- ・小文字の英字(a～z)
- ・数字(0～9)
- ・記号(+*!/#\$%&()=¥@<>?)

・データベースの指定

認証レスキュー！で使用するデータベースを指定します。選択肢は3種類です。

NR2 規定

認証レスキュー！での規定のデータベースです。
実際の運用には通常、この「NR2 規定」を選択します。

マイクロソフト社のクラウドサービス Microsoft Azure の仮想マシン(Windows Server 2019 など) を利用する場合は、この「NR2 規定」を指定します。

認証レスキュー！では、Web サーバー用 PC へのインストール時に SQL Server 2012 Express のインストールを選択できます。SQL Server 2012 Express をインストールして認証レスキュー！のインスタンス(Newtone)を作成し、NR2 規定のデータベース(NR2)を作成します。この段階では、データベース(NR2)にテーブルはまだありません。

後述の「テーブルデータ新規作成」を実行することで空のテーブルが作成されます。

NR2 サンプルデータベース

認証レスキュー！のサンプル用のデータベースです。簡単なサンプルデータが各テーブルに既に格納されたデータベースです。取り急ぎ認証レスキュー！全体を把握したい場合などに選択します。

後述の「テーブルデータ新規作成」を実行することでテーブルが作成され、サンプルデータが格納されます。

任意接続文字列(Azure など)

Web サーバー用 PC とは異なる(データベース)サーバーなどに SQL Server が用意され、そこに認証レスキュー！用のデータベースを置く場合やマイクロソフト社のクラウドサービス Microsoft Azure の「Web アプリ(旧 Web サイト)」に Web サービスを配置し、同「SQL データベース」に認証レスキュー！のデータベースを配置する場合に選択します。

Microsoft Azure で利用する場合や Web サーバー用 PC とは異なる(データベース)サーバーで利用する場合の利用方法は後述の説明ページをご覧ください。

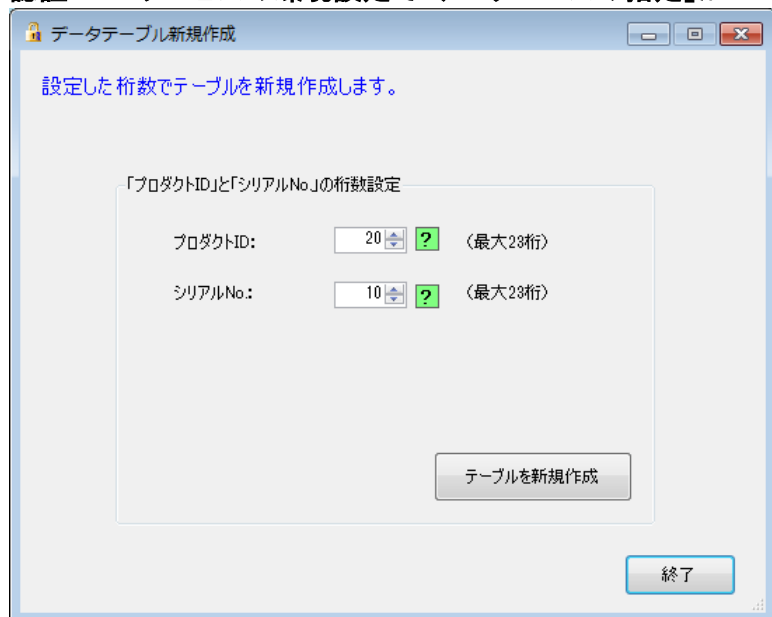
また、Web サーバー用 PC の SQL Server を利用するが、たとえばインスタンス名をデフォルトの「Newtone」から別のものに変更する場合などにもこの「任意接続文字列」を利用できます。この任意接続文字列の使用時は、テキストボックスに正しい接続文字列を入力する必要があります。

上記の設定をしたら「データベース接続確認」ボタンを押して、データベースに接続できることを確認してください。

■データテーブル新規作成

次に、データベースのテーブルの新規作成をします。
環境設定で「データテーブル新規作成」ボタンを押します。

認証 Web サービスの環境設定で「データベースの指定」が「NR2 規定」の場合



「テーブルを新規作成」ボタンを押すとデータベース内に**データの無いテーブル**が作成されます。実際のプロダクト ID やシリアル No.を含むキーデータは、「認証管理システム」のメニューの「認証キー作成」(自動ナンバリング)処理などで作成します。

以下にプロダクト ID やシリアル No.を簡単に説明しています。

・プロダクト ID

プロダクト ID は、製品を表わす任意の文字列です。桁数は運用開始時に設定し、以降は変更できません。

桁数の初期値は 17 桁です。最大 23 桁です。

なお、データベースの選択で「NR2 サンプルデータベース」をお試しいただく場合の桁数は 17 桁固定となり設定できません。

このプロダクト ID には、製品分類やバージョン、ライセンス数などを識別できる桁取りがあると管理しやすくなります。

たとえば、0000A-00002-00001 で製品 A のバージョン 2 の 1 ライセンスを、0000A-00002-00004 で製品 A のバージョン 2 の 4 ライセンスを、それぞれ示すように設定します。

プロダクト ID は、半角の次の文字が使用できます。

- ・大文字の英字(A～Z)
- ・小文字の英字(a～z)
- ・数字(0～9)
- ・記号(+-*!/#\$%&()=¥@<>?)

このプロダクト ID と「シリアル No.」で出荷時の一意の製品を示すキーとなります。

・シリアル No.

シリアル No.は、同一製品の識別連番を表わす文字列です。桁数は運用開始時に設定し、以降は変更できません。桁数の初期値は 8 桁です。最大 23 桁です。

なお、データベースの選択で「NR2 サンプルデータベース」をお試しいただく場合の桁数は 8 桁固定となり設定できません。

「認証キー作成」(自動ナンバリング)処理では、任意の上位桁数の文字列を指定することができます。

たとえば、8 桁中、上位 3 桁を“ABC”と指定した場合は次のような

シリアル番号の自動作成が可能です。

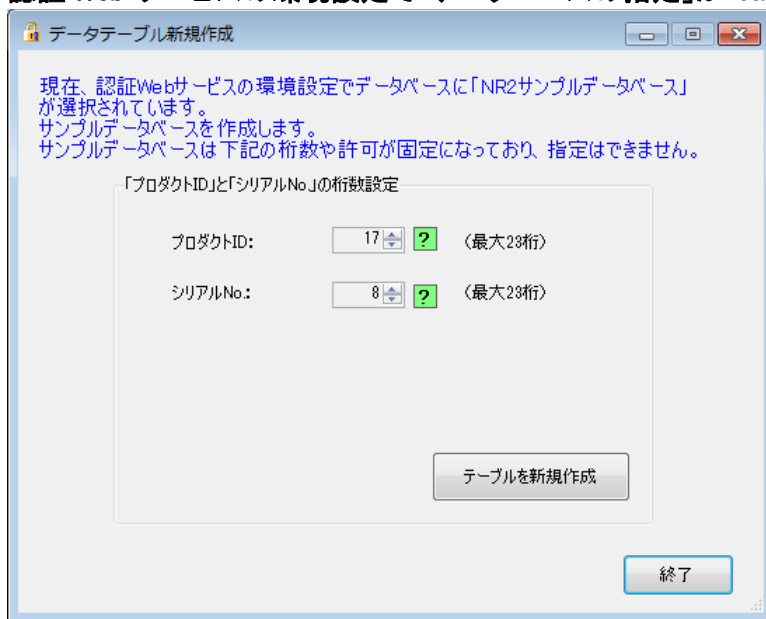
ABC00001
 ABC00002
 ABC00003
 ABC00004

シリアル No. は、半角の次の文字が使用できます。

- ・大文字の英字(A～Z)
- ・小文字の英字(a～z)
- ・数字(0～9)
- ・記号(+*!/#\$%&()=¥@<>?)

「プロダクト ID」とこのシリアル No. で出荷時の一意の製品を示すキーとなります。

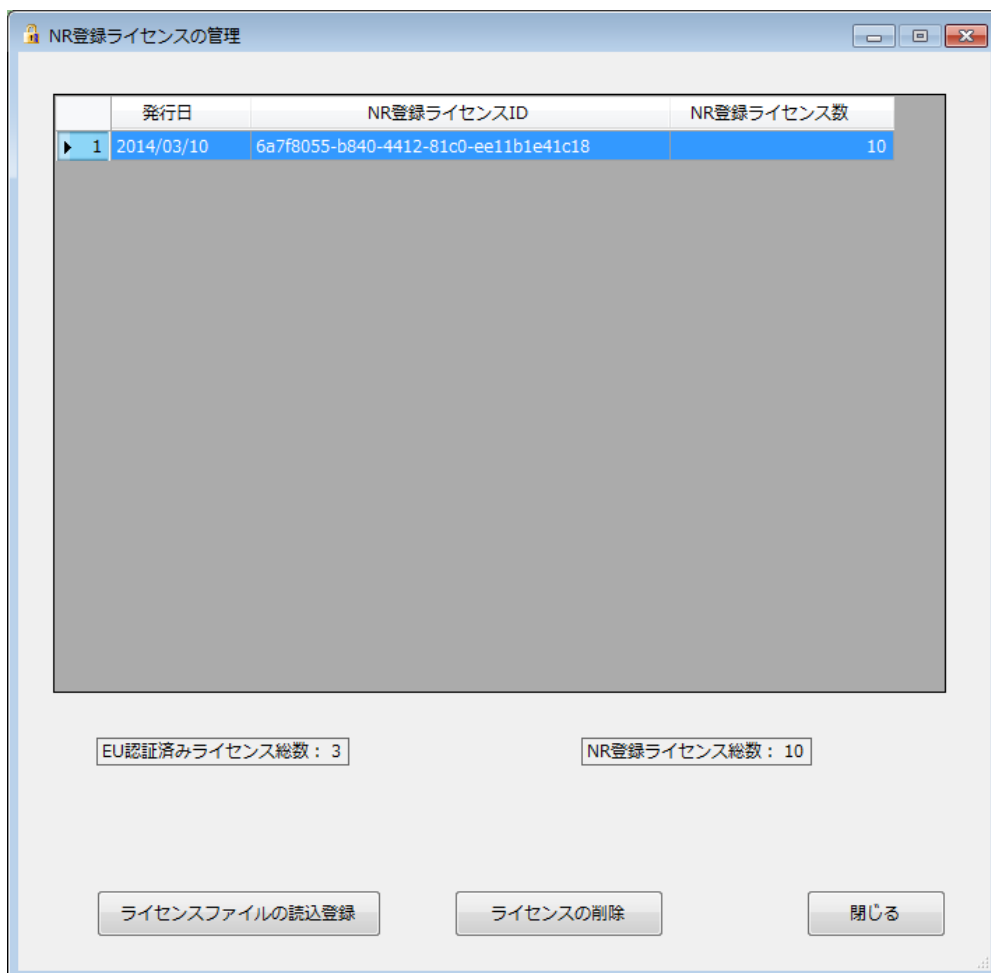
認証 Web サービスの環境設定で「データベースの指定」が「NR2 サンプルデータベース」の場合



「テーブルを新規作成」ボタンを押すとデータベース内にサンプルデータが入ったテーブルが作成されます。

■NR 登録ライセンスの管理

環境設定の「NR 登録ライセンスの管理」ボタンを押すと次のような画面が表示されます。



この「ライセンスの管理」処理では、お客様(エンドユーザ、以降 EU)が認証登録してきたライセンス総数の確認と弊社(ニュートン)から入手いただいた NR(認証レスキュー!)登録ライセンス数を管理できます。

「EU 認証済みライセンス総数」に EU が認証登録してきたライセンスの総数が、「NR 登録ライセンス総数」に現在登録されている貴社分の NR 登録ライセンスの総数がそれぞれ表示されます。

「ライセンスファイルの読込登録」ボタンで入手した NR 登録ライセンスを登録します。「ライセンスの削除」ボタンで登録してあった NR 登録ライセンスを削除します。

貴社のパッケージソフトを EU に配布して EU がライセンス登録をした場合、貴社 Web サーバーなどの認証レスキュー!用データベースの認証データテーブルにその登録ライセンスがレコード数として記録されます。

認証レスキュー!の運用では、その「EU 登録済みライセンス」の総数以上の NR 登録ライセンスを貴社にご用意いただく必要があります。

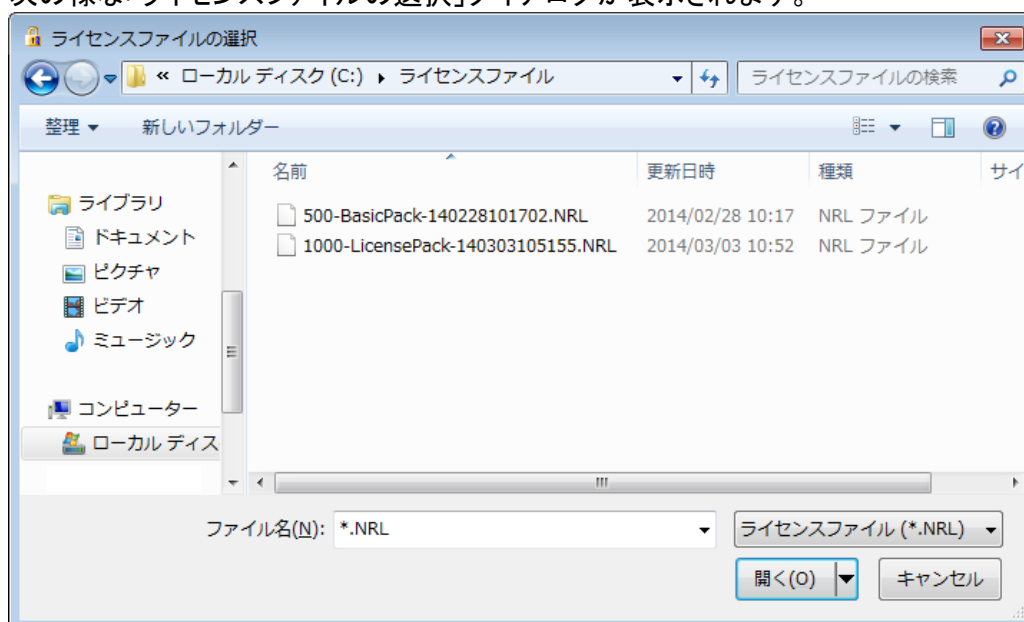
以下に NR 登録ライセンスの一覧表を示します。

<NR 登録ライセンス一覧>

| 種類 | 提供形態 | 想定 EU 登録済み ライセンス総数 |
|----------------------|-----------|-----------------------|
| 体験版ライセンス(10 登録ライセンス) | 体験版に付属 | 10 ライセンス分 |
| 基本パック(500 登録ライセンス付属) | 製品に付属 | 500 ライセンス分 |
| 1,000 登録ライセンスパック | オプション(別売) | 1,000 ライセンス分 |
| 5,000 登録ライセンスパック | オプション(別売) | 5,000 ライセンス分 |
| 10,000 登録ライセンスパック | オプション(別売) | 10,000 ライセンス分 |
| 50,000 登録ライセンスパック | オプション(別売) | 50,000 ライセンス分 |
| 100,000 登録ライセンスパック | オプション(別売) | 100,000 ライセンス分 |
| 無制限登録ライセンスパック | オプション(別売) | 制限なし |

それではここで実際に、「ライセンスファイルの読込登録」ボタンを押して、NR 登録ライセンスを登録します。

次の様な「ライセンスファイルの選択」ダイアログが表示されます。



ここで、貴社が入手した NR 登録ライセンスファイルを指定します。
基本パックに付属している 500 登録ライセンスファイルであれば、ディスクやメールで入手した 500-BasicPack-xxxxxxxxxxxx.NRL といったファイルを指定します。

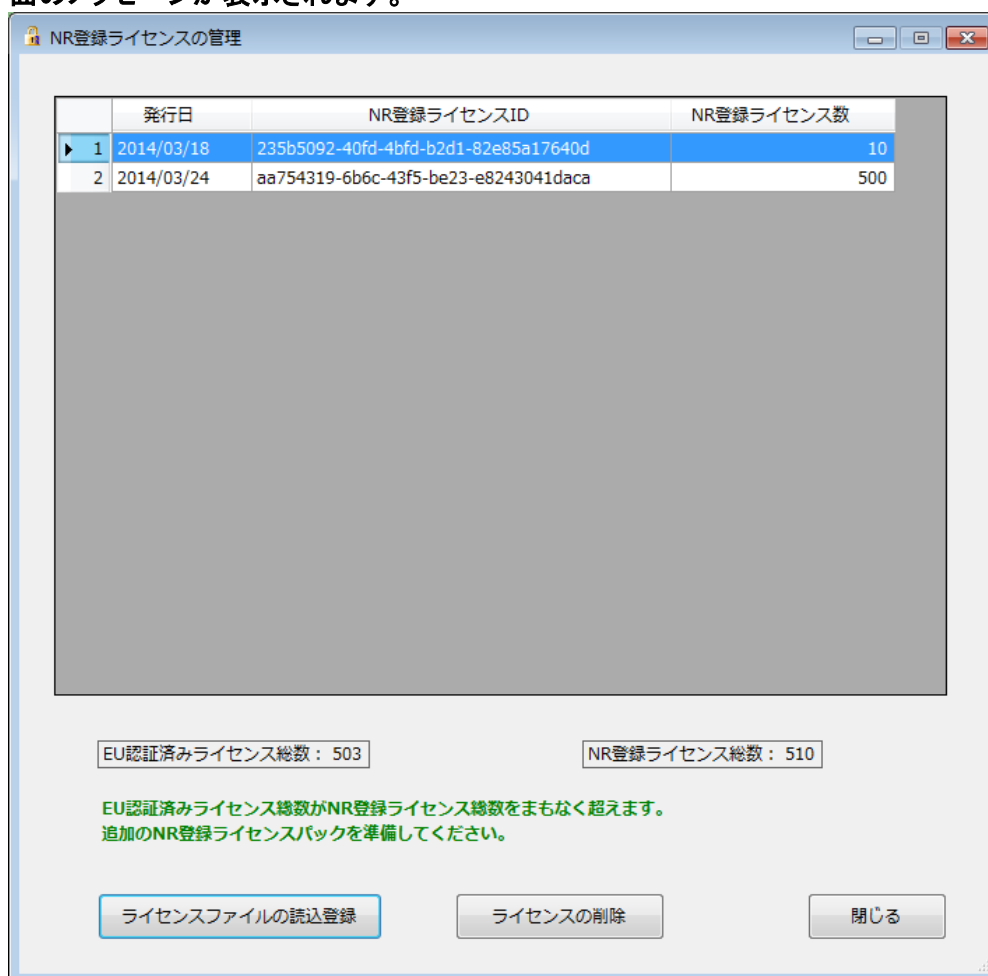
また、体験版や基本パックの場合は、インストール先がデフォルトなら、
<32bitOS の場合> C:\Program Files\Newtone\NR2\NR2Web
<64bitOS の場合> C:\Program Files (x86)\Newtone\NR2\NR2Web
に、「10-TrialLicense-xxxxxxxxxxxx.NRL」といった 10 登録分の NR 登録ライセンスファイルがあるのでそれを利用できます。

このダイアログで「開く」ボタンを押すと指定した NR 登録ライセンスファイルが読み込まれ、「NR 登録ライセンスの管理」画面に反映されます。

■NR 登録ライセンスの警告

この「ライセンスの管理」処理で、NR 登録ライセンスに関する警告が表示される場合があります。

[1] NR 登録ライセンス総数が EU 登録済みライセンス総数の 90%を超えた場合に次のような画面のメッセージが表示されます。



この場合、各 PC での処理の動作は次のようになります。

・認証業務用社内 PC での「認証管理システム」

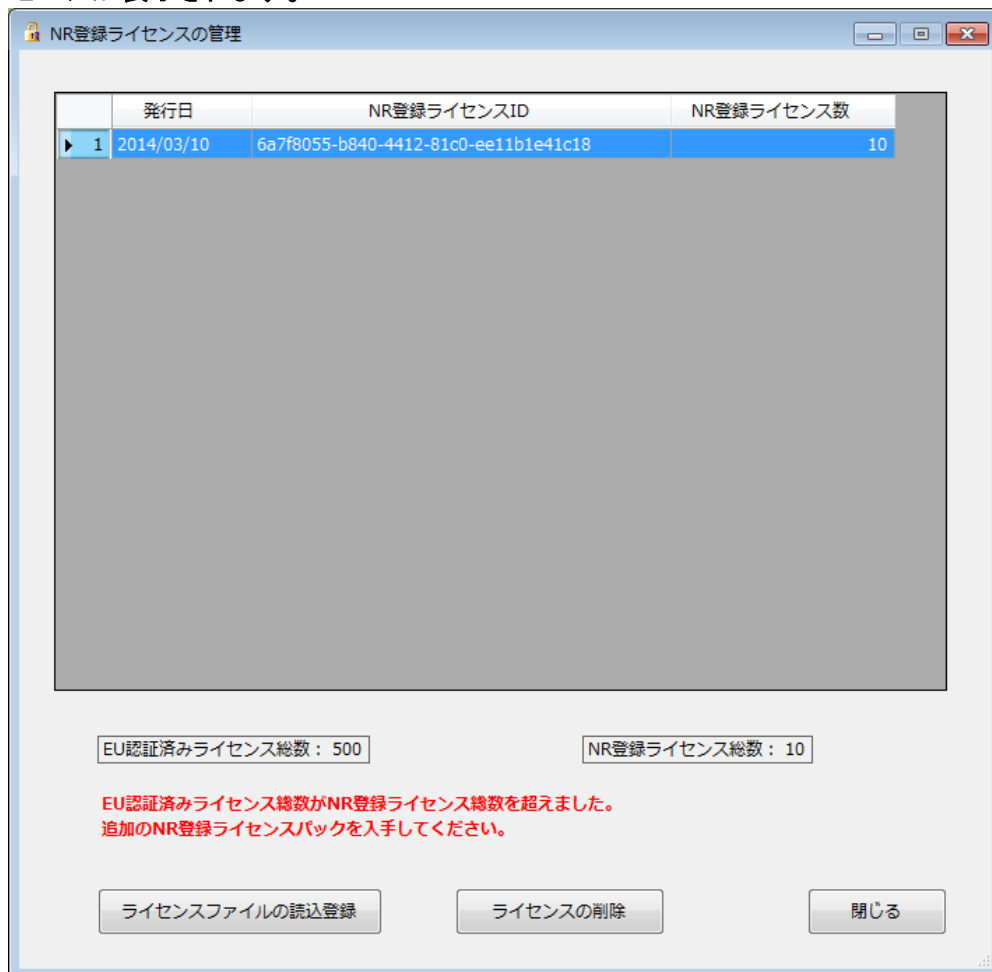
認証レスキュー！用のデータベースのデータアクセス箇所では次のメッセージを表示し処理は続行します。

”EU 認証済みライセンス総数が NR 登録ライセンス総数をまもなく超えます。追加の NR 登録ライセンスパックを準備をしてください。”

・認証 UI ライブラリ(DLL)を使用する貴社アプリケーションが動作する EU の PC

認証レスキュー！用のデータベースのデータアクセス箇所ではメッセージを表示せず、処理は続行されます。

[2] NR 登録ライセンス総数が EU 登録済みライセンス総数を超えた場合は次のような画面のメッセージが表示されます。



この場合、各 PC での処理の動作は次のようになります。

・**認証業務用社内 PC での「認証管理システム」**

認証レスキュー！用のデータベースのデータアクセス箇所では次のメッセージを表示し処理を中断します。

”EU 認証済みライセンス総数が NR 登録ライセンス総数を超えました。追加の NR 登録ライセンスパックを入手してください”

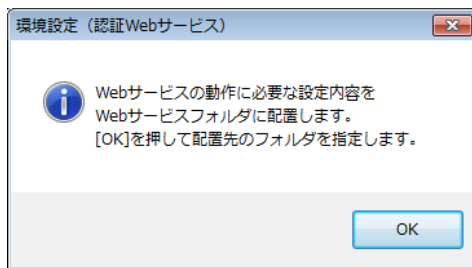
・**認証 UI ライブラリ(DLL)を使用する貴社アプリケーションが動作する EU の PC**

認証レスキュー！用のデータベースのデータアクセス箇所では次のメッセージを表示し処理を中断します。

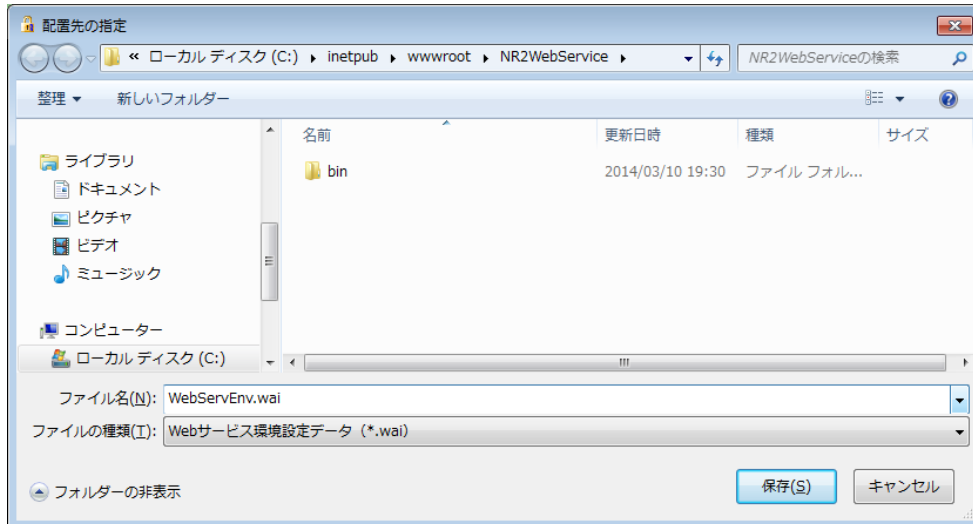
”NR 登録ライセンスに問題があります。アプリケーションベンダにご連絡ください。”

■ **「登録」ボタン**

環境設定が終了したら「登録」ボタンを押します。次のようなメッセージが表示されます。



ここで、「OK」ボタンを押すと、次のような「配置先の指定」ダイアログが表示されます。



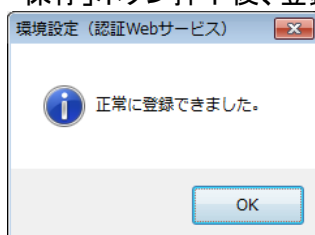
環境設定で入力した情報の内、Web サービスの動作に必要な情報を Web サービス環境設定データ (WebServEnv.wai) ファイルとして、この「配置先の指定」ダイアログで IIS の認証レスキュー！用の Web サービスが配置してあるフォルダにも出力します。通常は、自動的に(例: C:\inetpub\wwwroot\NR2WebService などの)適切な出力先が表示されますので、確認してそのまま「保存」ボタンを押します。

マイクロソフト社のクラウドサービス Microsoft Azure の「Web アプリ (旧 Web サイト)」に Web サービスを配置する場合は、この Web サービス環境設定データ (WebServEnv.wai) ファイルも Web サービスと同じフォルダにコピーする必要があります。この後の「NR 登録ライセンスの管理」でライセンスを登録した場合も Web サービス環境設定データ (WebServEnv.wai) ファイルが更新されますが、更新されるたびに Microsoft Azure の「Web アプリ (旧 Web サイト)」に上書き配置 (コピー) する必要があります。

Microsoft Azure の「Web アプリ (旧 Web サイト)」への配置については詳しくは、後述の「Web サービスとデータベースを Microsoft Azure で利用する方法」をご覧ください。

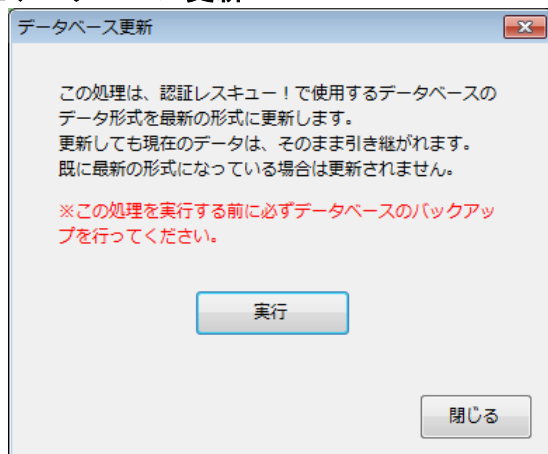
なお、環境設定で入力した情報で Web サービスの動作と関係しない情報は、Web サービス環境設定データ (WebServEnv.wai) ファイルではなく、この「環境設定」を実行している PC のレジストリに記録されます。

「保存」ボタン押下後、登録に成功すると次のメッセージが表示されます。



以下は必要に応じて行う処理を説明します。

■ データベース更新



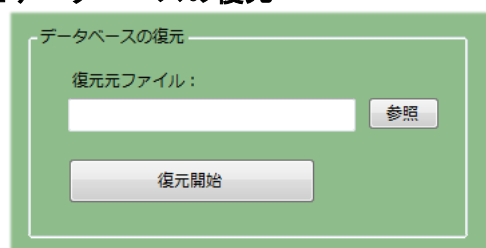
この処理は、認証レスキュー！で使用するデータベースのデータ形式を最新の形式に更新します。更新しても現在のデータは、そのまま引き継がれます。既に最新の形式になっている場合は更新されません。

■ データベースのバックアップ



この処理は、認証レスキュー！で使用するデータベースをバックアップします。「バックアップ先ファイル」を指定して「バックアップ開始」ボタンを押すとすぐにバックアップが実行されます。定期的なバックアップを設定するには、「スケジュール化する」をチェックしてバックアップする間隔を指定します。

■ データベースの復元



この処理は、バックアップしてあるデータベースを復元します。「復元元ファイル」を指定して「復元開始」ボタンを押すとすぐに復元が実行されます。

2. 認証業務用社内 PC の「認証管理システム」

デスクトップ上の「認証レスキュー！ 認証管理システム」へのショートカットを起動すると次の画面が表示されます。

この実行ファイルは、インストール先がデフォルトなら、

<32bitOS の場合> C:\Program Files\Newtone\NR2\NR2InsideSystem\InsideSystem.exe、

<64bitOS の場合> C:\Program Files (x86)\Newtone\NR2\NR2InsideSystem\InsideSystem.exe
です。



「認証管理システム」のメニューが表示されますが、最初に行うのは「環境設定」処理です。初期状態では他の処理は選択できません。

■環境設定

「環境設定」ボタンを押すと次のような画面が表示されます。

ここでは、設定が必須で省略できない項目についてだけ説明します。

•Web サービス/URL

認証に関するシステムを Web サービスとして提供する Web サーバーの URL を指定します。
以下にいくつか例を示します。

★自 PC のローカルホストの Web サーバー (IIS) にアクセスする場合：

`http://localhost/Nr2WebService/Service.aspx`

★自社 Web サーバー (IIS) にアクセスする場合：

例えば、URL は

`http://www.newtone.co.jp/Nr2WebService/Service.aspx`

といったものになります。

この、「www.newtone.co.jp」部分が貴社の Web サイトになります。

★クラウドサービス Microsoft Azure の仮想マシンを利用する場合：

例えば、URL は

`http://newtone-nr-2019-1.westus2.cloudapp.azure.com/Nr2WebService/Service.aspx`

といったものになります。

この、「newtone-nr-2019-1.westus2.cloudapp.azure.com」部分が、貴社の Azure ポータルの「仮想マシン」の「基本」の「DNS 名」に表示されているものとなります。

★クラウドサービス Microsoft Azure の Web アプリ (旧 Web サイト) に配置した Web サービスを利用する場合：

例えば、URL は

`http://Newtonejp.azurewebsites.net/Service.aspx`

といったものになります。

この、「http://Newtonejp.azurewebsites.net」部分が、貴社の Azure ポータルの「Web アプリ」の「基本」の「URL」に表示されているものとなります。

・Web サービス/確認パスワード

Web サービスを利用する場合の確認用のパスワードを設定します。ここでは、Web サーバー側設定アプリケーション「認証 Web サービス」の環境設定処理の Web サービスの確認パスワードと同じものを設定します。

確認パスワードは必須項目です、省略はできません。

8 文字以上で半角の次の文字が使用できます。

- ・大文字の英字(A～Z)
- ・小文字の英字(a～z)
- ・数字(0～9)
- ・記号(+*!/#\$%&()=¥@<>?)

上記の設定をしたら「Web サービス接続確認」ボタンを押して、Web サービスに接続できることを確認してください。

「登録」ボタンを押して設定を保存します。設定内容はこの「環境設定」を実行しているPCのレジストリに保存されます。

■認証キー作成（自動ナンバリング）

次に、認証キーを作成します。3 種類の作成方法がありますが、ここでは自動ナンバリングで作成します。

「認証管理システム」のメニューで「認証キー作成(自動ナンバリング)」ボタンを押します。

次の画面が表示されます。

「作成」ボタンを押すと認証キーが作成されます。

「認証キー一覧」ボタンを押すと作成した認証キーの一覧が次のように表示されます。

検索

日付: 指定する 2016/03/28 ~ 2016/03/28

プロダクトID: ? (未入力:指定なし)

シリアルNo. 先頭指定文字列: ? (未入力:指定なし)

検索実行

| | プロダクトID | シリアルNo. | ライセンス数 | プラス許可数 | 有効期限利用 | 有効期限 | 日時 |
|-----|-------------------|----------|--------|--------|-------------------------------------|------------|---------------------|
| ▶ 1 | 00001-00001-00001 | A0000001 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 2 | 00001-00001-00001 | A0000002 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 3 | 00001-00001-00001 | A0000003 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 4 | 00001-00001-00001 | A0000004 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 5 | 00001-00001-00001 | A0000005 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 6 | 12345-12345-12345 | 1234ABCD | 5 | 0 | <input checked="" type="checkbox"/> | 2018/03/31 | 2014/03/25 18:17:00 |

終了

以下に各項目を説明します。

・上位固定文字列

上位固定文字列は、シリアル No.において任意の上位桁数の文字列を指定する場合に指定します。

たとえば、8 桁中、上位 3 桁を“ABC”と指定した場合は次のようなシリアル番号の自動作成が可能です。

ABC00001
 ABC00002
 ABC00003
 ABC00004

・開始番号

開始番号は、シリアル No.において連番を自動付番する場合の最初の数を指定します。「間隔(ステップ)数」や「ナンバリング数」とともにシリアル No.の自動付番時の必須項目です。

たとえば、8 桁中、上位 3 桁を“ABC”と指定し、開始番号を 1、間隔(ステップ)数を 2、ナンバリング数を 5 とそれぞれ指定した場合、次のようなシリアル番号が自動作成されます。

ABC00001
 ABC00003
 ABC00005
 ABC00007
 ABC00009

・間隔(ステップ)数

間隔(ステップ)数は、シリアル No.において連番を自動付番する場合の付番間隔を数で指定します。「開始番号」や「ナンバリング数」とともにシリアル No.の自動付番時の必須項目です。

たとえば、8桁中、上位3桁を“ABC”と指定し、開始番号を1、間隔(ステップ)数を2、ナンバリング数を5とそれぞれ指定した場合、次のようなシリアル番号が自動作成されます。

ABC00001
ABC00003
ABC00005
ABC00007
ABC00009

・ナンバリング数

ナンバリング数は、シリアル No.において自動付番で作成する(シリアル No.の)数を指定します。「開始番号」や「間隔(ステップ)数」とともにシリアル No.の自動付番時の必須項目です。

たとえば、8桁中、上位3桁を“ABC”と指定し、開始番号を1、間隔(ステップ)数を2、ナンバリング数を5とそれぞれ指定した場合、次のようなシリアル番号が自動作成されます。

ABC00001
ABC00003
ABC00005
ABC00007
ABC00009

・ライセンス数

ライセンス数は、製品のライセンス数を指定します。

たとえば、1ライセンスなら1、5ライセンスなら5と指定します。「認証キー作成」(自動ナンバリング)処理でライセンス数を指定する場合、1度に作成する自動付番内に異なるライセンスを指定することはできません。

ライセンス数は、「プロダクト ID」や「シリアル No.」などととも認証キーテーブルの項目のひとつです。

・有効期限設定

製品の認証登録後、無期限で使用できるライセンスではなく、特定の有効期限まで使用可能とするライセンス形態にする場合はこの設定を行います。

有効期限後も継続して使用可能とする場合の有効期限の再設定は、認証管理システムの「認証キー編集(表形式)」処理を利用します

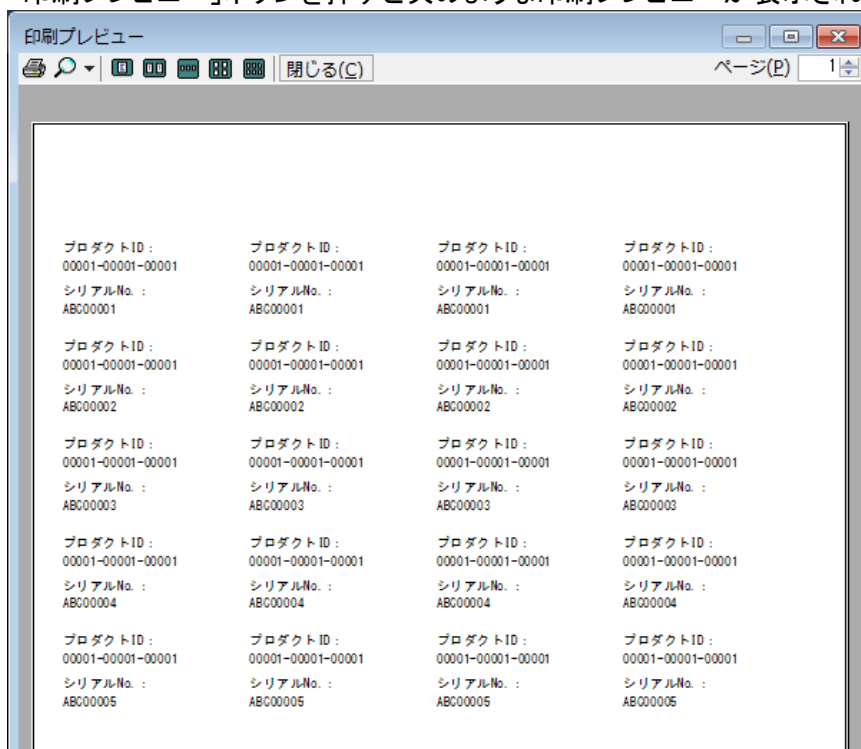
■ラベル印刷

次に、お客様(エンドユーザ)にアプリケーションを配布するために、前工程で作成した認証キーからプロダクト ID とシリアル No.のラベルシールを印刷します。

「認証管理システム」のメニューで「ラベル印刷」ボタンを押します。



ここでは、必要に応じて検索条件を入力して「検索実行」ボタンを押します。
表示された認証キーの一覧に対して印刷する対象にチェックをつけます。
「印刷プレビュー」ボタンを押すと次のような印刷プレビューが表示されます。



ラベルシールは、このようにプロダクトIDとシリアルNo.が横それぞれ4枚ずつ印刷されます。
その1行分にはすべて同じ内容が印刷されます。
奇数行目のプロダクトIDと次の行の偶数行目のシリアルNo.で対となります。

同じラベルシールが 4 枚あるのは、それぞれの利用目的があるからです。たとえば、次のような用途です。

- 1 枚目: パッケージ内の CD (DVD) ジャケット (ケース) 添付 (貼付) 用
- 2 枚目: パッケージ内のユーザ登録用紙添付用
- 3 枚目: 出荷指示書添付用
- 4 枚目: 注文書控え添付用

・用紙の仕様

このラベルシールの印刷用紙は、一般的な市販のラベルシール用紙を想定しています。具体的には、次の仕様です。

用紙サイズ: A4

ラベル数: 横 4 × 縦 9 = 36 面

ラベルサイズ: 45.7 × 25.4mm

余白: 上 31.8mm / 左 10.2mm / 右 9.5mm / 下 36.6mm

メーカー例: ヒサゴ「A4 タックシール」GB871 など

●Microsoft Azure で「認証レスキュー！」を利用する方法

ここでは、「認証レスキュー！」をマイクロソフト社のクラウドサービス Microsoft Azure で利用する方法について説明します。

「認証レスキュー！」を Microsoft Azure で利用するには、次の 2 種類の利用方法があります。Microsoft Azure の仮想マシンを利用する方法と Microsoft Azure の Web アプリと SQL データベースを利用する方法です。

前者の仮想マシンを利用する場合は、自社サーバーへのインストールとほぼ同様の手順で「認証レスキュー！」の利用開始ができます。

後者の Web アプリと SQL データベースを利用する場合は、前者に比べ稼働前の手順が少し複雑です。

Microsoft Azure での運用コストは、両者とも各環境のスペックの選択やオプションによりますが、例えば、最安価の選択をした場合は下表の通りほぼ同様と思われます。

Microsoft Azure 運用コストの最安価の例：月額

| 利用方法 | リソース | | 計 |
|------------------------|-------------------------------------|----------------------------------------|---------|
| 仮想マシンを利用 | 仮想マシン | | \$13.19 |
| | レベル: Basic インスタンス: A0 \$13.19 | | |
| Web アプリと SQL データベースを利用 | App Service | Azure SQL Database | \$14.39 |
| | レベル: 共有 インスタンス: D1 \$9.49 | サービスレベル: Basic 購入モデル: DTU \$4.90 | |

マイクロソフト社 Azure 料金計算ツールサイトより (2022 年 2 月時点)

■Microsoft Azure の仮想マシンを利用する

まず、貴社がマイクロソフト社に申込み、Microsoft Azure の「仮想マシン」機能が使用できる状態が必要です。Microsoft Azure の申込みなどにつきましては、マイクロソフト社の情報をご覧ください。

ここでは、貴社がすでに Microsoft Azure の「仮想マシン」を使用できる状態を前提としています。

1.仮想マシンの作成（参考）

Microsoft Azure ポータルサイト

The screenshot shows the Microsoft Azure portal interface. At the top, there is a search bar with the text "リソース、サービス、ドキュメントの検索 (G+)". Below the search bar, the page title is "リソースの作成". On the left side, there is a navigation menu with categories like "AI + Machine Learning", "分析", "ブロックチェーン", "Compute", "コンテナ", "データベース", "開発者ツール", and "DevOps". The "Compute" category is selected. In the main content area, there is a search bar with the text "サービスとマーケットプレースを検索してください". Below the search bar, there is a section titled "人気のある製品 Marketplace でさらに表示". The "仮想マシン" (Virtual Machines) item is highlighted with a red box. Other items listed include "仮想マシン スケール セット", "Kubernetes Service", and "関数アプリ".

The screenshot shows the Microsoft Azure portal interface for creating a virtual machine. The page title is "仮想マシンの作成". Below the title, there are tabs for "基本", "ディスク", "ネットワーク", "管理", "詳細", "タグ", and "確認および作成". The "基本" tab is selected. The main content area contains the following sections:

- 基本**: A paragraph explaining that Linux or Windows virtual machines can be created from the Azure Marketplace or custom images. It includes a link to "詳細情報".
- プロジェクトの詳細**: A paragraph explaining that a subscription and resource group must be selected for deployment and cost management.
- サブスクリプション ***: A dropdown menu showing the selected subscription.
- リソース グループ ***: A dropdown menu showing the selected resource group, with a link to "新規作成".
- インスタンスの詳細**: A section with the following fields:
 - 仮想マシン名 ***: A text input field.
 - 地域 ***: A dropdown menu showing "(US) West US 2".
 - 可用性オプション**: A dropdown menu showing "インフラストラクチャ冗長は必要ありません".

At the bottom of the page, there are three buttons: "確認および作成", "< 前へ", and "次: ディスク >".

2.作成した仮想マシン（例）

The screenshot shows the Microsoft Azure portal interface. At the top, there is a search bar with the text 'リソース、サービス、ドキュメントの検索 (G+/)'. Below the search bar, the breadcrumb 'ホーム >' is visible. The main content area displays the virtual machine 'VNWinServer2019' with a status of '仮想マシン'. A toolbar contains several action buttons: '接続' (Connect), '開始' (Start), '再起動' (Restart), '停止' (Stop), 'キャプチャ' (Capture), and '削除' (Delete). Below the toolbar, a '基本' (Basic) section lists various properties:

- リソース グループ (移動) : VNWinServer2019_group
- 状態 : 停止済み (割り当て解除)
- 場所 : West US 2
- サブスクリプション (移動) : [Redacted]
- サブスクリプション ID : [Redacted]
- オペレーティング システム : Windows
- サイズ : Standard DS1 v2 (1 vcpu, 3.5 GiB メモリ)
- パブリック IP アドレス : [Redacted]
- 仮想ネットワーク/サブネット : VNWinServer2019_group-vnet/default
- DNS 名 : newtone-nr-2019-1.westus2.cloudapp.azure.com
- タグ (編集) : タグを追加するにはここをクリック

At the bottom, there are tabs for 'プロパティ', '監視', '機能 (8)', '推奨事項', and 'チュートリアル'. A summary card below the tabs shows the VM icon, name '仮想マシン', computer name 'VNWinServer2019', and status '正常性の状態'.

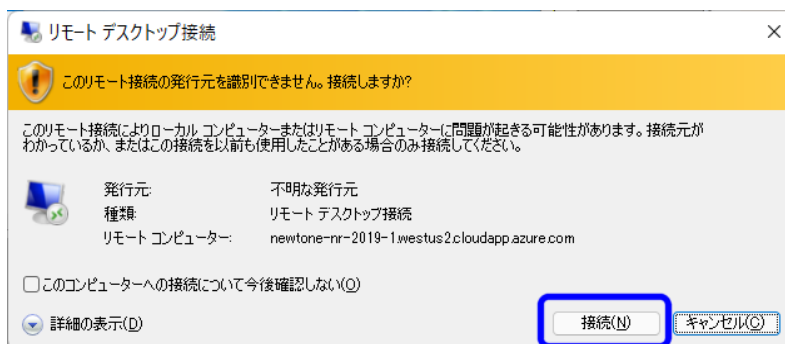
3.仮想マシンの開始（例）

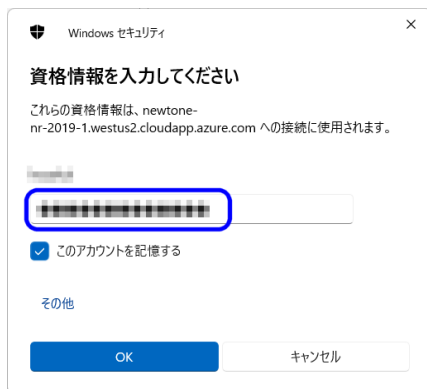
仮想マシンを開始します。

This screenshot is similar to the previous one, showing the details of the 'VNWinServer2019' virtual machine. The '開始' (Start) button in the toolbar is highlighted with a blue square, indicating the action to be performed. The rest of the page content, including the search bar, breadcrumb, and property list, remains the same as in the previous screenshot.

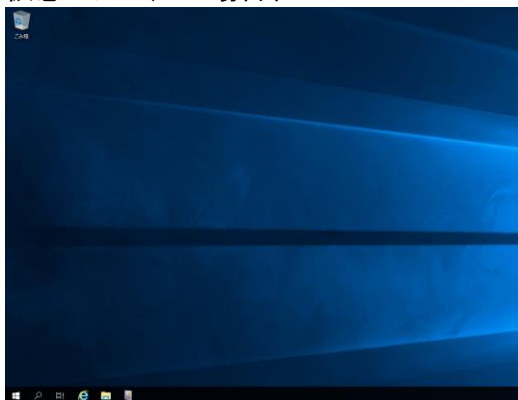
4.仮想マシンの接続（例）

仮想マシンに接続をします。
ここでは、RDP（リモート デスクトップ）を選択します。





仮想マシン(この場合、Windows Server 2019)のデスクトップが表示されます。



5.仮想マシンへの認証レスキュー！のインストール

入手した認証レスキュー！がパッケージの場合は、ディスク内の全フォルダの全ファイル、ダウンロードなどの場合は解凍したフォルダにある全フォルダの全ファイルを、仮想マシンの任意のフォルダにコピーしてください。全体で 2GB 以上ありますので通信環境によっては時間がかかる場合があります。

次に、コピーしたルートフォルダにある「NR2InstallMenu.exe」を実行してください
通常通り、認証レスキュー！のインストーラで「Web サーバー用 PC」へのインストールを行います。
以降の手順は、本操作ガイドの[「インストール」](#)の[「Web サーバー用 PC へのインストール」](#)と同様ですので、リンク先をご覧ください。

「Web サーバー用 PC へのインストール」が完了するとデスクトップに「認証レスキュー！ 2 Web 環境設定」のショートカットが表示されます。



これで、Microsoft Azure の仮想マシンへのインストールは終了です。

■Microsoft Azure の Web アプリ（旧 Web サイト）と SQL データベースを利用する

まず、貴社がマイクロソフト社に申込み、Microsoft Azure の「Web アプリ（旧 Web サイト）」機能と「SQL データベース」機能が使用できる状態が必要です。

Microsoft Azure の申込みなどにつきましては、マイクロソフト社の情報をご覧ください。

ここでは、貴社がすでに Microsoft Azure の「Web アプリ（旧 Web サイト）」機能と「SQL データベース」機能を使用できる状態を前提としています。

下図は、Microsoft Azure の「Web アプリ（旧 Web サイト）」機能と「SQL データベース」機能を使用している場合の Web 上の管理サイト画面の一例です。

Microsoft Azure ポータルサイト

The image shows two overlapping screenshots of the Microsoft Azure portal. The top screenshot displays the 'SQL データベース' (SQL Databases) management page. The left sidebar lists navigation options: ホーム (Home), 通知 (Notifications), 参照 (References), アクティブ (Active), 課金 (Billing), and 新規 (New). The main content area shows a table of databases:

| データベース | 状態 |
|-----------|-------|
| NR2 | オンライン |
| NR2SAMPLE | オンライン |

The bottom screenshot displays the 'Web アプリ' (Web Apps) management page. The left sidebar is similar to the top screenshot. The main content area shows a table of web applications:

| 名前 | 状態 |
|-----------|-----|
| NewtoneJP | 実行中 |

Microsoft Azure ポータルサイト(旧)



以下の手順を行います。

1. 認証レスキュー! の「Web サーバー用 PC」へのインストールを行う

仮のサーバーPC、またはローカル PC に、前述の「Web サーバー用 PC」へのインストールを行います。

2.認証レスキュー！用のデータベースを Microsoft Azure の「SQL データベース」に配置する

認証レスキュー！用のデータベースは前述の「Web サーバー用 PC」へのインストールで「データベースのインストール」を行うと(まだテーブルデータの無い)データベースが作成されます。「データベースのインストール」では SQL Server 2012 Express + Management Studio がインストールされます。Microsoft Azure の「SQL データベース」への認証レスキュー！用のデータベースの配置には、この Management Studio を利用すると簡単です。

以下にその手順を示します。

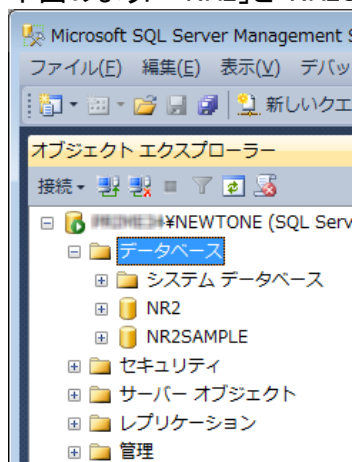
スタートメニューから Microsoft SQL Server 2012→SQL Server Management Studio を選択します。



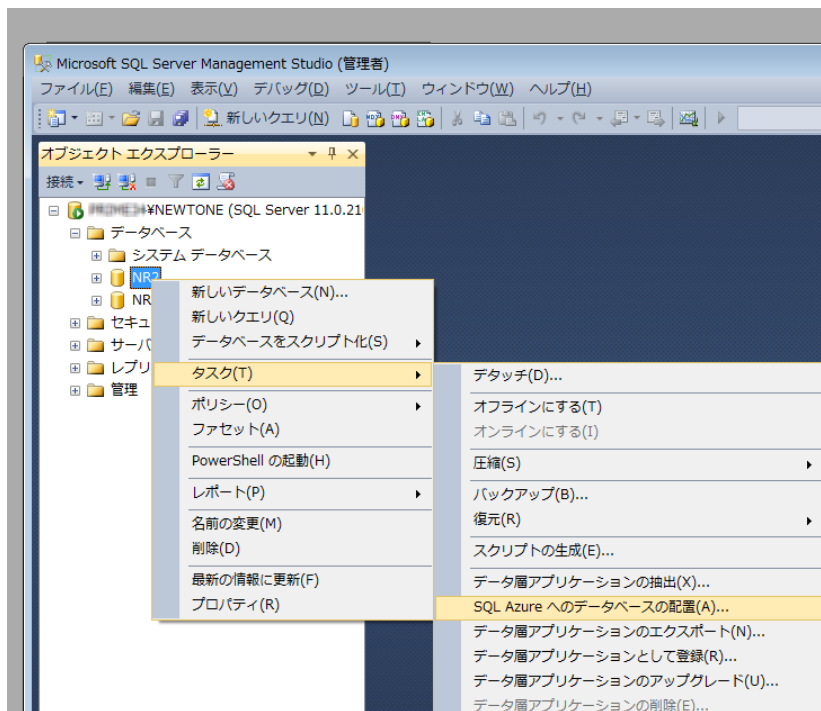
「サーバーへの接続」ダイアログで NEWTONE インスタンスを選択して、接続ボタンを押します。



下図のように「NR2」と「NR2SAMPLE」という 2 つのデータベースがあります。



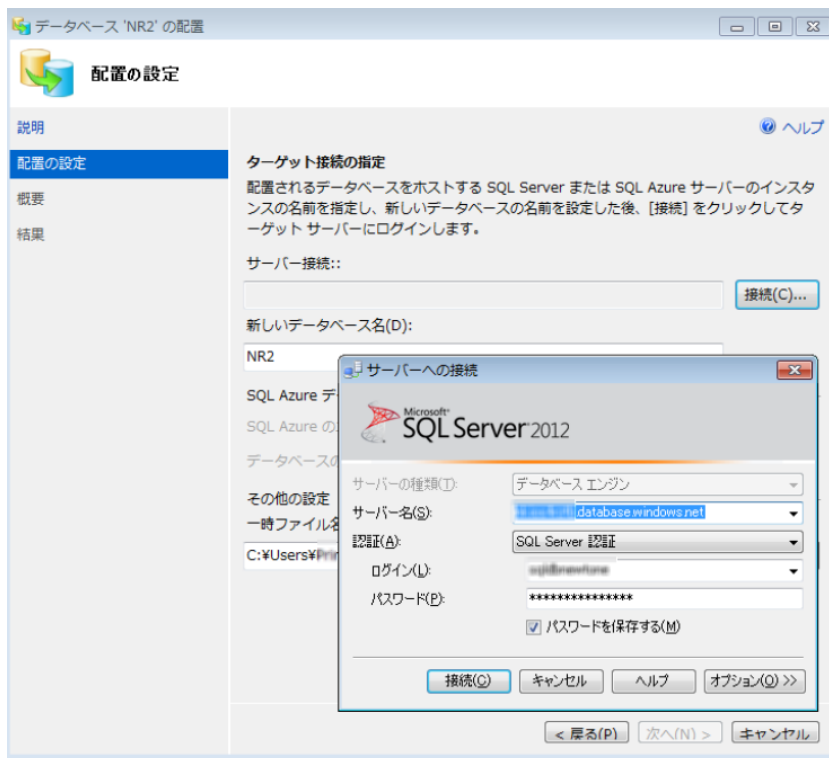
たとえば、データベース「NR2」を Microsoft Azure に配置(コピー)する場合は、オブジェクトエクスプローラーの「NR2」上で右クリックしてコンテキストメニュー内の「タスク」→「SQL Azure へのデータベースの配置」を選択します。



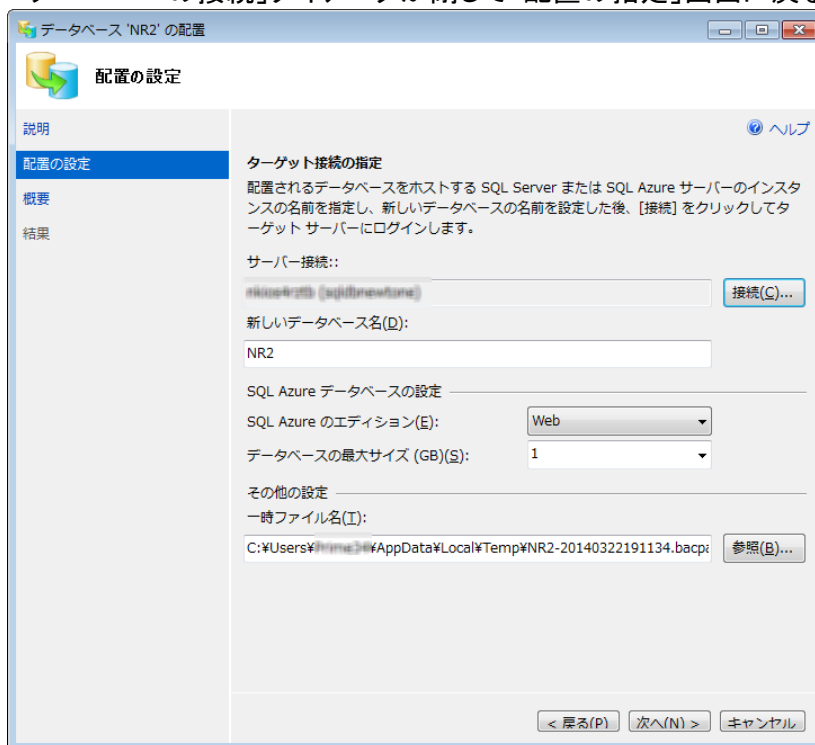
下図のようにデータベースの配置ウィザードが表示されます。
この画面の説明にあるように Microsoft Azure の SQL データベースのアカウント情報を準備します。「次へ」を選択します。



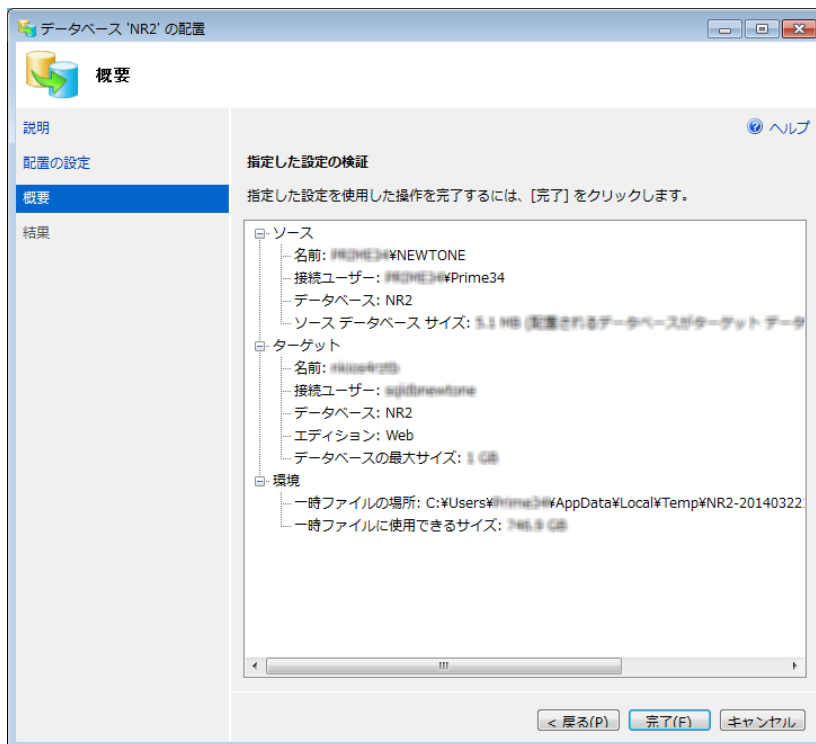
下図で、サーバー接続の「接続」ボタンを押して、「サーバーへの接続」ダイアログが表示されるので、サーバー名や認証(方法)、ログイン(名)、パスワードを Microsoft Azure の SQL データベースのアカウント情報から入力して、接続ボタンを押します。



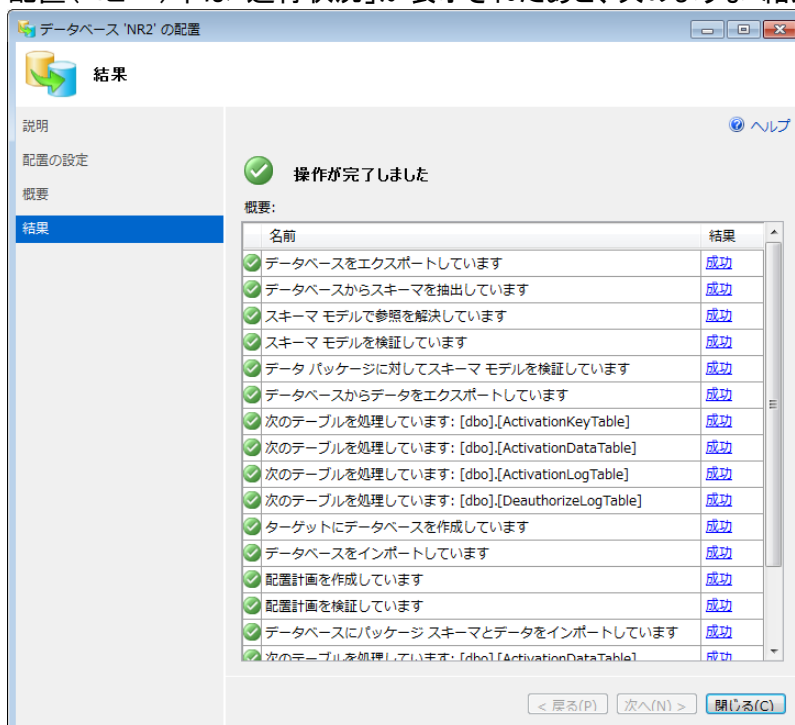
「サーバーへの接続」ダイアログが閉じて「配置の指定」画面に戻るので「次へ」を選択します。



次に表示される「概要」画面で「完了」ボタンを押します。



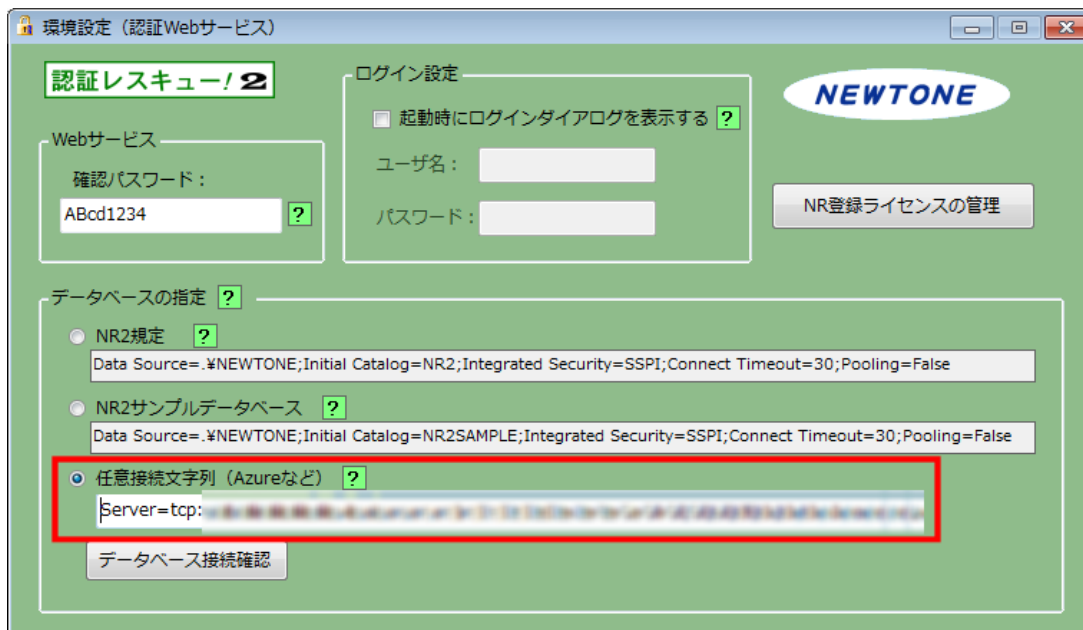
配置(コピー)中は「進行状況」が表示されたあと、次のような「結果」画面が表示されます。



「閉じる」ボタンを押して、認証レスキュー！用のデータベース「NR2」の Microsoft Azure の「SQL データベース」への配置(コピー)は終了です。

3. 認証レスキュー！の Web サーバー用 PC の「環境設定」処理を完了する

認証レスキュー！の Web サーバー用 PC の「環境設定」のデータベースの指定で「任意接続文字列(Azure など)」を選択して、下のテキストボックスに Azure の「SQL データベース」への接続文字列を入力します。



この接続文字列の確認方法は次の通りです。

Microsoft Azure の SQL データベースの管理ページで、「接続文字列の表示」リンクを押すといくつかの接続文字列が表示されます。その中で、「ADO.NET」用の接続文字列を使用します。その接続文字列の（{ここにパスワードを挿入}となっている）パスワード部分を Microsoft Azure の SQL データベースで設定した実際のパスワードに変えます。

<Microsoft Azure の SQL データベースの接続文字列の表示例>

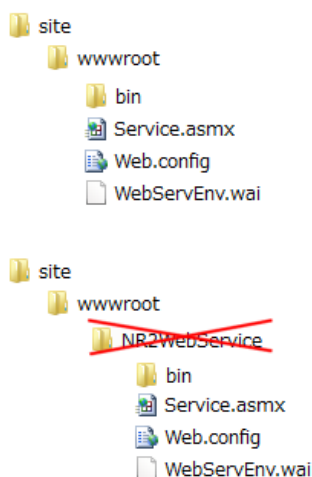


設定したら必ず「データベース接続確認」ボタンで接続を確認してください。

4.Microsoft Azure の「Web アプリ」に認証レスキュー！の Web サービスを配置する

仮のサーバーPC、またはローカル PC に前述の「Web サーバー用 PC」へのインストールを行います。その結果、その PC の IIS のサイトに「NR2WebService」仮想サイトがインストールされます。エクスプローラでそのサイトの実態フォルダ（例：C:\inetpub\wwwroot\NR2WebService など）内のすべてのファイルとフォルダとを FTP などを利用して Microsoft Azure の「Web アプリ（旧 Web サイト）」のルート（例：xxxxxxx/site/wwwroot/）に配置（コピー）します。

この際、Microsoft Azure の「Web アプリ（旧 Web サイト）」の仕様で、Web サービスページ（Service.asmx）は必ず Microsoft Azure の「Web アプリ（旧 Web サイト）」のルート（例：xxxxxxx/site/wwwroot/）である必要があります。
wwwroot の下に作ったフォルダに配置してはいけません。
たとえば次の場合です。



上図では、上が正しい配置です。下の配置例は URL をそのように指定しても Web サービス実行時に Microsoft Azure からエラーが返りますので注意してください。

Microsoft Azure の「Web アプリ（旧 Web サイト）」側の FTP ホスト名は、たとえば ftp://xxxx-xxxx-xxx-001.azurewebsites.windows.net/site/wwwroot/ などと表します。これは、Microsoft Azure の Web アプリ（旧 Web サイト）の管理ページで確認できます。

この配置するファイルの中には、認証レスキュー！の Web サービス環境設定データ（WebServEnv.wai）が含まれます。

このファイルは、「Web サーバー用 PC」へのインストール直後には存在しません。前述の Web サーバー用 PC の「環境設定」処理を完了すると上記の IIS のフォルダ（例：C:\inetpub\wwwroot\NR2WebService など）内に保存されます。このため、上記 3 の「Web サーバー用 PC の「環境設定」処理を完了する」のステップは省略できませんので、ご注意ください。

5. Microsoft Azure 上で認証レスキュー！を運用している場合の Web アプリと SQL データベースのバージョンアップの方法

Microsoft Azure 上で認証レスキュー！2 の Web サービスとデータベースを既に運用されている場合の更新方法は次の通りです。

【手順 1】(「Web サーバー用 PC」のインストール)

[1] 仮のサーバー PC、またはローカル PC に認証レスキュー！2 の新しいバージョンの「Web サーバー用 PC」のインストールを行います。
新しいバージョンをインストールした PC の IIS のサイトに「NR2WebService」仮想サイトがインストールされます。

【手順 2】(Web サービスの更新)

[1] エクスプローラで上記の IIS のサイトの実態フォルダ
(例: C:\inetpub\wwwroot\NR2WebService など)内のすべてのファイルとフォルダとを FTP などを利用して Microsoft Azure の「Web アプリ (旧 Web サイト)」のルート(例: xxxxxx/site/wwwroot/)に配置(上書きコピー)します。

ただし、これらのファイルの中で Web サービスの動作に必要な情報が記録されている「WebServEnv.wai」はコピーの対象から外します。インストーラによりインストールされた WebServEnv.wai ファイルを上書きせず、Microsoft Azure 上の現行の貴社の WebServEnv.wai ファイルをそのまま利用します。

Microsoft Azure への配置に関しましては、「[4. Microsoft Azure の「Web アプリ」に認証レスキュー！の Web サービスを配置する](#)」をご参照ください。

【手順 3】(データベースの更新)

[1] Microsoft Azure 上のデータベースを更新する前に、バックアップ(エクスポート)することをお勧めいたします。

(例)

SQL Server 2012 の Management Studio で Microsoft Azure 上のデータベースに接続してデータベースを選択後、右クリックのコンテキストメニューより「タスク」→「データ層アプリケーションのエクスポート」を実行

[2] Microsoft Azure 上の現行の貴社の WebServEnv.wai ファイルを FTP などを利用して新しいバージョンをインストールした PC の IIS のサイト(例: C:\inetpub\wwwroot\NR2WebService など)にコピーします。WebServEnv.wai ファイルには Azure への接続文字列なども格納されていますので必ず新しいバージョンをインストールした PC の IIS のサイトへコピーしてください。

[3] 新しいバージョンをインストールした PC のデスクトップ上の「認証レスキュー！ Web 環境設定」ショートカットで環境設定(認証 Web サービス)「WebAdmin.exe」を起動します。

この起動時、データベースが古い形式の場合は自動的に「データベース更新の確認」ダイアログが表示されます。「今すぐ、更新しますか」といったメッセージが表示されますので「今すぐ実行する」ボタンを選択しデータベースを更新します。

なお、既存のデータはそのまま引き継がれます。

環境設定の画面の「終了」ボタンを選択して環境設定を終了します。

以上で、Web サービスとデータベースの新しいバージョンへの更新作業は終了です。

【ご注意】

上記の Web サービスとデータベース以外の「認証管理システム」や「認証 UI ライブラリ(DLL)」も最新版をインストールし、更新してください。「認証UIライブラリ(DLL)」は、エンドユーザに配布する必要がありますが、エンドユーザが古いバージョンの DLL を使用していて、Web サービスが新しいバージョンの場合でも互換を保ち動作いたしますが、適時更新してください。

■ Azure でのシステム標準時刻に関する注意点

Microsoft Azure の仕様でシステム標準時刻は UTC(世界協定時)になっています。これにより、「認証管理システム」の「認証状況」処理などで日付を指定して検索する場合、日本では 9 時間を引いた日付で検索する必要があります。例えば、2014 年 4 月 1 日の 15:00 のタイムスタンプのデータを検索する場合は、9 時間引いても 2014 年 4 月 1 日の 6:00 で、同じ日付なので 2014/04/01 で検索できます。しかし、2014 年 4 月 1 日の 6:00 のタイムスタンプのデータを検索する場合は、9 時間引くと 2014 年 3 月 31 日の 21:00 となり、2014/03/31 で検索しないとヒットしませんので、ご注意ください。なお、次の方法で Azure のシステム標準時刻を日本時間に設定できます。

Azure のポータルサイトで設定する App Service を選択し、設定グループの「構成」を選択します。次に「アプリケーション設定」を選択して「新しいアプリケーション設定」を選択します。



「アプリケーション設定の追加/編集」画面で、次の内容で入力します。

名前: WEBSITE_TIME_ZONE

値: Tokyo Standard Time

アプリケーション設定の追加/編集

名前

値

デプロイ スロットの設定

更新

キャンセル

「更新」ボタンを押します。

次のように設定されたのが確認できます。

ホーム > NewtoneJP - 構成

NewtoneJP - 構成 ×
App Service

検索 (Ctrl+/)

設定

- 構成
- アプリケーション設定 (クラシック)
- 認証/承認
- Application Insights
- ID
- バックアップ
- カスタムドメイン
- TLS/SSL の設定
- ネットワーク

保存 破棄

アプリケーション設定 全般設定 既定のドキュメント バスのマッピング

アプリケーション設定

アプリケーション設定は保存時に暗号化され、暗号化されたチャネルで送信されます。以下のコントロールを使用して、お使いのブラウザーにプレーンテキストで表示するよう選択することができます。アプリケーション設定は、実行時にアプリケーションでアクセスするための環境変数として公開されます。 [詳細情報](#)

+ 新しいアプリケーション設定 値を非表示にする 高度な編集 フィルター

| 名前 | 値 | デプロイス... |
|-------------------|---------------------|--------------------------------|
| WEBSITE_TIME_ZONE | Tokyo Standard Time | 🗑️ ✎ |

●アプリケーション開発用 PC での「認証 UI ライブラリ」の利用

■認証 UI ライブラリ関連ファイル

デスクトップ上の「認証レスキュー！ 認証 UI ライブラリ」へのショートカットを実行すると認証 UI ライブラリが格納されている（インストール先がデフォルトの場合）次のフォルダが開きます。

<32bitOS の場合> C:\Program Files\Newtone\NR2\NR2DLL

<64bitOS の場合> C:\Program Files (x86)\Newtone\NR2\NR2DLL

このフォルダに格納されているファイルは次の通りです。

| フォルダ名 | ファイル名/フォルダ名 | 内容 |
|------------------------------------------------------------------------------------|----------------------|--------------------------------|
| NRDLL/Framework4.0 および NRDLL/Framework3.5 | Newtone.NR.dll | 認証 UI ライブラリ(DLL)本体 |
| | Newtone.NR.tlb | タイプライブラリ(VisualBasic6.0 使用時必要) |
| | NewtoneNRvcpp.dll | VC++用ラッパーDLL(VC++使用時必要) |
| | NewtoneNRvcpp.tlb | タイプライブラリ(VC++使用時必要) |
| SampleProject (UI 系サンプルプロジェクト) または SampleProject_API (API 系サンプルプロジェクト) | VisualBasic2010 フォルダ | Visual Basic 2010 用のサンプルプロジェクト |
| | CSharp2010 フォルダ | C# 2010 用のサンプルプロジェクト |
| | VisualBasic6.0 フォルダ | Visual Basic 6.0 用のサンプルプロジェクト |
| | VC++2010 フォルダ | Visual C++ 2010 用のサンプルプロジェクト |
| SampleProject_Web (ASP.NET 系サンプルプロジェクト) | VisualBasic2010 フォルダ | Visual Basic 2010 用のサンプルプロジェクト |
| | CSharp2010 フォルダ | C# 2010 用のサンプルプロジェクト |

このライブラリを利用して、お客様(エンドユーザ)へ配布する貴社のアプリケーションにアクティブーション機能を組み込みます。

まずは、サンプルプロジェクトをお試しになり、ソースファイルをご確認ください。

なお、UI 系機能と API 系機能の利用形態の選択については次ページをご覧ください。

■認証 UI ライブラリの利用形態による選択

認証レスキュー！の Ver.2.5.0 以降の認証 UI ライブラリ(DLL)では、従来の UI 系機能のプロパティやメソッドとは別に、API 系機能のプロパティやメソッドを利用できるラインナップ(+API)が追加されました。

UI 系機能と API 系機能の大きな違いは、UI 系が認証レスキュー！の既定の UI(ユーザインターフェイス)を利用する前提であることに対し、API 系は貴社オリジナルの UI を作成し利用できる点です。

たとえば、「認証登録/インターネット」処理を考えた場合、UI 系では ActivateRegisterInternet メソッドを呼び出すだけで、「認証登録/インターネット」処理の UI が表示され UI 上の項目の細かいプログラムの制御などは全く必要ありません。

それに対し、API 系では最終的には APIActivateRegisterInternet メソッドを呼び出しますが、UI にあたる Form のデザインや Form 上の項目に関する細かい制御や APIActivateRegisterInternet メソッドを呼び出すための必須プロパティの設定などのコードを貴社で作成する必要があります。API 系では各メソッド実行結果の細かいステータスは返しますが、メッセージの表示を含め一切の画面表示は行いません。

認証 UI ライブラリ(DLL)の利用形態によるメリットとデメリットは次表の通りです。

| 利用形態 | メリット | デメリット |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI 系機能 | <ul style="list-style-type: none"> ●認証レスキュー！既定の UI を利用するため、貴社アプリケーションにアクティベーション機能を短時間で実装することが可能 ●API 系製品(+API)に比べ安価である | <ul style="list-style-type: none"> ●貴社オリジナルの UI を構築できないため、貴社アプリケーションとのシームレスな UI デザインや UI 上の(認証レスキュー！既定 UI には用意されていない)新規項目などは利用できない ●認証レスキュー！既定の UI を利用するため日本語のみの UI となる |
| API 系機能 | <ul style="list-style-type: none"> ●貴社オリジナルの UI を構築できるため、貴社アプリケーションとのシームレスな UI デザインや UI 上の(認証レスキュー！既定 UI には用意されていない)新規項目などを作成できる ・貴社オリジナルの UI を構築できるため日本語以外の多言語の UI が作成可能 ●プロダクト ID やシリアル No.を(レジストリより)取得できるメソッドがあるため、その文字列を定義することで、異なる製品や同一製品の追加オプションのライセンスを同一 PC 内で識別できる。 ※次ページの(2)に例を掲載、UI 系にはそれらを取得する機能はない | <ul style="list-style-type: none"> ●貴社オリジナルの UI を構築するため UI 系に比べ、貴社アプリケーションのアクティベーション機能実装に時間がかかる ●UI 系製品に比べ高価である |

■同一 PC 内で異なるアプリケーションやオプションのライセンスを識別する方法

2つの方法があります。

(1) 同一 PC 内で貴社の異なるアプリケーションのライセンスを識別する場合

この方法は、同一 PC 内の異なるアプリケーションのライセンスの識別ができますが、同一アプリケーション内の異なるオプションなどについてはライセンスの識別はできません。

ベンダアプリケーション開始レジストリキーパス設定用のプロパティにアプリケーションごとに異なるレジストリパスを設定します。

UI 系:

[VendorsProductStartRegistryKeyPath](#) プロパティ

アプリケーションA内での設定コード例:

VendorsProductStartRegistryKeyPath = "Software¥Company¥A"

アプリケーションB内での設定コード例:

VendorsProductStartRegistryKeyPath = "Software¥Company¥B"

API 系:

[APIVendorsProductStartRegistryKeyPath](#) プロパティ

アプリケーションA内での設定コード例:

APIVendorsProductStartRegistryKeyPath = "Software¥Company¥A"

アプリケーションB内での設定コード例:

APIVendorsProductStartRegistryKeyPath = "Software¥Company¥B"

(2) 同一 PC 内で貴社の異なるアプリケーションやオプションのライセンスを識別する場合

この方法は、同一 PC 内の同一アプリケーション内の異なるオプションについてもライセンスの識別ができます。この方法は、API 系限定の機能を使用します。UI 系では利用できません。

例えば、「プロダクト ID」9 桁とし先頭 4 桁をアプリケーションの製品コード、最後の 3 桁をオプションコードとします。

[プロダクト ID の桁の定義]

| | | | | | | | | |
|-------|---|---|---|----------|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 製品コード | | | | オプションコード | | | | |

[製品コードの定義]

0001: アプリケーション A

0002: アプリケーション B

0003: アプリケーション C

....

[オプションコードの定義]

オプションコード1桁目(プロダクト ID 7 桁目): 追加機能 1 がある場合 1、ない場合 0

オプションコード 2 桁目(プロダクト ID 8 桁目):追加機能 2 がある場合 1、ない場合 0

オプションコード 3 桁目(プロダクト ID 9 桁目):追加機能 3 がある場合 1、ない場合 0

[貴社でのプログラム]

ライセンスを確認したいタイミングで [APIGetRegisteredInfoFromRegistry](#) メソッドを実行して、認証後のレジストリより「プロダクト ID」を取得します。その値の先頭の 4 桁の製品コードでアプリケーションを識別し、最後の 3 桁でオプションを識別できます。

例えば、取得したプロダクト ID が次の通りであれば

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 2 | 1 | 2 | 1 | 0 | 1 |

エンドユーザは、アプリケーション B の追加機能 1 と追加機能 3 のライセンスを保持していることとなります。

■UI系サンプルプロジェクトとAPI系サンプルプロジェクトについて

Ver.2.5.0以降の認証UIライブラリ(DLL)用のサンプルプロジェクトは次の2種類です。

- ・SampleProject フォルダ (UI系サンプルプロジェクト)
- ・SampleProject_API フォルダ (API系サンプルプロジェクト)

これらのサンプルプロジェクトは、実際に実行してみると一見同じように見えますが、ソースコードを確認するとその相違点がわかります。

UI系サンプルプロジェクトが認証レスキュー！既定のUIを呼び出すサンプルプロジェクトになっているのに対し、API系サンプルプロジェクトではUI用のFormは全て貴社でカスタマイズ可能です。

■ASP.NET系サンプルプロジェクトについて

Ver.2.6.0以降で利用できるASP.NET(Webアプリケーション)用のサンプルプロジェクトは次の通りです。

- ・SampleProject_Web フォルダ (ASP.NET系サンプルプロジェクト)

このサンプルプロジェクトは、ASP.NETで作成された貴社のWebページから利用することができる、ASP.NET系のDLLの利用方法が含まれています。

■認証 UI ライブラリ機能一覧

【UI 系、API 系 DLL】

アセンブリ:Newton.NR (Newton.NR.dll 内)

名前空間:Newton.NR

Visual C++での利用時は、次の VC++用ラッパーDLL を呼び出すようになります。

アセンブリ:NewtonNRvcpp (NewtonNRvcpp.dll 内)

名前空間:NewtonNRvcpp

【ASP.NET 系 DLL】

アセンブリ:Newton.NR.ASPNET (Newton.NR.ASPNET.dll 内)

名前空間:Newton.NR.ASPNET

<クラス>

| 内容 | クラス名 |
|--------------------------|----------------|
| 認証に関する各種機能の提供(UI 系) | Activation |
| 認証に関する各種機能の提供(API 系) | APIActivation |
| 認証に関する各種機能の提供(ASP.NET 系) | APIActivation2 |

<UI系プロパティ>

プロパティ一覧

| プロパティ | 機能 |
|-------------------------------------------------------|---------------------------|
| EncryptionPassword | 暗号化時のパスワードを設定 |
| EncryptionSaltString | 暗号化時の Salt 文字列を設定 |
| ProductIdNumberOfDigits | プロダクト ID の桁数を設定 |
| RentalPeriod | レンタル日数を設定 |
| RentalPeriodName | レンタル期間の名称を設定 |
| SerialNoNumberOfDigits | シリアル No.の桁数を設定 |
| SetProductID | 認証登録時の設定プロダクト ID を設定 |
| SetSerialNo | 認証登録時の設定シリアル No.を設定 |
| TelephoneNumber | 電話で認証時の電話番号を設定 |
| TrialPeriod | 猶予(試用)日数を設定 |
| TrialPeriodName | 猶予(試用)期間の名称を設定 |
| UseCpuInfo | CPU 情報の使用を設定 |
| UseMacAddress | MAC アドレスの使用を設定 |
| VendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパスを設定 |
| WebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワードを設定 |
| WebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名を設定 |
| WebServiceCheckPassword | Web サービス確認パスワードを設定 |
| WebServiceTimeout | Web サービスのタイムアウトを設定 |
| WebServiceURL | Web サービスの URL を設定 |
| WebServiceUseBasicAuthentication | Web サービス時の基本認証の使用を設定 |

| |
|--------------------------|
| EncryptionPassword プロパティ |
|--------------------------|

【機能】

暗号化時のパスワードを設定します。

【構文】

<VB2010/VB6.0>

Public Property **EncryptionPassword** As **String**

<C#>

```
public string EncryptionPassword { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation::EncryptionPassword
```

【解説】

認証 UI ライブラリ(DLL)はエンドユーザ PC のレジストリやファイルにデータを出力する際に必要に応じてデータを暗号化しています。その際の暗号化の方法は貴社では指定できませんが、暗号化する時の2つのパラメータ、パスワードと Salt 文字列は貴社で指定できます。

EncryptionPassword プロパティには暗号化時のパスワードを指定します。全角でも指定できます。

(例)“認証レスキュー!”

この EncryptionPassword プロパティの文字数は、空文字列は不可で1~65535文字ですが、8文字から15文字程度が妥当と思われます。

| |
|----------------------------|
| EncryptionSaltString プロパティ |
|----------------------------|

【機能】

暗号化時の Salt 文字列(8 文字以上)を設定します。

【構文】

<VB2010/VB6.0>

Public Property **EncryptionSaltString** As **String**

<C#>

```
public string EncryptionSaltString { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation::EncryptionSaltString
```

【解説】

認証 UI ライブラリ(DLL)はエンドユーザ PC のレジストリやファイルにデータを出力する際に必要に応じてデータを暗号化しています。その際の暗号化の方法は貴社では指定できませんが、暗号化する時の 2 つのパラメータ、パスワードと Salt 文字列は貴社で指定できます。

EncryptionSaltString プロパティには暗号化時の Salt 文字列を指定します。

必ず 8 文字以上で指定します。

(例) "12345678ABCDEFGH"

| |
|-------------------------------|
| ProductIdNumberOfDigits プロパティ |
|-------------------------------|

【機能】

プロダクト ID の桁数(デフォルト:17)を設定します。

【構文】

<VB2010>

Public Property **ProductIdNumberOfDigits** As **Integer**

<VB6.0>

Public Property **ProductIdNumberOfDigits** As **Long**

<C#>

```
public int ProductIdNumberOfDigits { set; get; }
```

<VC++>

```
public : long NewtonNrvcpp::IActivation::ProductIdNumberOfDigits
```

【解説】

認証業務用社内 PC の「認証管理システム」の「データテーブル新規作成」処理時に指定したプロダクト ID の桁数を設定します。

RentalPeriod プロパティ**【機能】**

レンタル日数(デフォルト:0日、設定可能範囲:1~1100)を設定します。

【構文】

<VB2010>

Public Property **RentalPeriod** As **Integer**

<VB6.0>

Public Property **RentalPeriod** As **Long**

<C#>

```
public int RentalPeriod { set; get; }
```

<VC++>

```
public : long NewtonenRvcpp::IActivation::RentalPeriod
```

【解説】

エンドユーザが初めて認証してからアプリケーションが動作する期間を指定します。

1(日)~1100(日)の間で設定できます。

この RentalPeriod プロパティを 0 に設定すると、レンタル期間は無いことになります。

※お客様のパッケージ製品にマルチライセンス(例:10ライセンス)が含まれる場合はレンタル機能は使用できません。レンタル機能を使用するにはシングルライセンス(1ライセンス)の製品である必要があります。また、レンタル機能を使用する場合は「電話で認証登録」機能は利用できません。

RentalPeriodName プロパティ**【機能】**

レンタル期間の名称(デフォルト:“レンタル”)を設定します。

【構文】

<VB2010/VB6.0>

Public Property **RentalPeriodName** As **String**

<C#>

```
public string RentalPeriodName { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation::RentalPeriodName
```

【解説】

たとえば、ActivateStatusDisp メソッドでは認証状態が表示されます。現在、レンタル中ならば、「レンタル期間残日数:100日」といったように表示されます。その中の「レンタル」という文字列を別の文字列で指定することができます。それがこの RentalPeriodName プロパティです。

(例)“使用可能”

SerialNoNumberOfDigits プロパティ**【機能】**

シリアル No.の桁数(デフォルト:8)を設定します。

【構文】

<VB2010>

Public Property **SerialNoNumberOfDigits** As **Integer**

<VB6.0>

Public Property **SerialNoNumberOfDigits** As **Long**

<C#>

```
public int SerialNoNumberOfDigits { set; get; }
```

<VC++>

```
public : long NewtonenRvcpp::IActivation::SerialNoNumberOfDigits
```

【解説】

認証業務用社内 PC の「認証管理システム」の「データテーブル新規作成」処理時に指定したシリアル No.の桁数を設定します。

| |
|--------------------|
| SetProductID プロパティ |
|--------------------|

【機能】

認証登録時の設定プロダクト ID (デフォルト: 空文字列) を設定します。

【構文】

<VB2010/VB6.0>

Public Property **SetProductID** As **String**

<C#>

public **string** **SetProductID** { set; get; }

<VC++>

public : **_bstr_t** **NewtoneNRvcpp::IActivation::SetProductID**

【解説】

このプロパティが空文字列の場合は、認証登録系のメソッドで表示されるダイアログのプロダクトID用のテキストボックスは、エンドユーザによる入力が可能となります。

このプロパティが空文字列ではない場合は、認証登録系のメソッドで表示されるダイアログ中のプロダクトID用のテキストボックスにこのプロパティ値がグレーアウト表示され文字列コピーはできるが入力できません。

このプロパティは認証登録系のメソッドに対し、プロダクト ID を渡すためだけに利用され、認証登録処理後に実際に登録されたプロダクト ID で自動的に書き換えられるものではありません。

対象となる認証登録系メソッドは、次の通りです。

「認証登録/インターネット」処理の呼び出し

(ActivateRegisterInternetメソッド)

「認証登録/電話」処理の呼び出し

(ActivateRegisterTelephoneメソッド)

「認証登録状態回復」処理の呼び出し

(RestoreRegisterStatusメソッド)

「代理認証登録/実行」処理の呼び出し

(ProxyActivateRegisterExecute メソッド)

SetProductID、SetSerialNoプロパティの目的

プロダクトIDやシリアルNo.に貴社が取り決めたある識別を設けた場合に、意図したように認証登録を行うため。

SetProductID、SetSerialNoプロパティの利用例

プロダクトIDやシリアルNo.にレンタル製品とそうではない通常(売切り)製品の識別を持たせるとします。

プロダクトIDとシリアルNo.は、認証レスキュー！のメソッドのUIではなく、貴社側独自のプログラムでエンドユーザの入力を受け必要なチェック後プロダクトIDやとシリアルNo.をこのプロパティに設定したあとレンタル製品かどうかに応じてレンタル日数(RentalPeriod)プロパティを設定してから認証登録系のメソッドを呼び出すといった流れになります。

SetSerialNo プロパティ

【機能】

認証登録時の設定シリアル No. (デフォルト: 空文字列) を設定します。

【構文】

<VB2010/VB6.0>

Public Property **SetSerialNo** As **String**

<C#>

```
public string SetSerialNo { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation::SetSerialNo
```

【解説】

このプロパティが空文字列の場合は、認証登録系のメソッドで表示されるダイアログのシリアル No. 用のテキストボックスは、エンドユーザによる入力が可能となります。

このプロパティが空文字列ではない場合は、認証登録系のメソッドで表示されるダイアログ中のシリアル No. 用のテキストボックスにこのプロパティ値がグレーアウト表示され文字列コピーはできるが入力できません。

このプロパティは認証登録系のメソッドに対し、シリアル No. を渡すためだけに利用され、認証登録処理後に実際に登録されたシリアル No. で自動的に書き換えられるものではありません。

対象となる認証登録系メソッドは、次の通りです。

「認証登録/インターネット」処理の呼び出し

(ActivateRegisterInternetメソッド)

「認証登録/電話」処理の呼び出し

(ActivateRegisterTelephoneメソッド)

「認証登録状態回復」処理の呼び出し

(RestoreRegisterStatusメソッド)

「代理認証登録/実行」処理の呼び出し

(ProxyActivateRegisterExecuteメソッド)

このプロパティの目的と利用例は、SetProductID プロパティの説明をご覧ください。

| |
|-----------------------|
| TelephoneNumber プロパティ |
|-----------------------|

【機能】

電話で認証時の電話番号を設定します。

【構文】

<VB2010/VB6.0>

Public Property **TelephoneNumber** As **String**

<C#>

```
public string TelephoneNumber { set; get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation::TelephoneNumber
```

【解説】

エンドユーザがインターネットを使わずに電話での認証を行う「認証登録/電話」処理で、エンドユーザがかける電話の番号を指定します。

TrialPeriod プロパティ**【機能】**

猶予(試用)日数(デフォルト:0日、設定可能範囲:1~365)を設定します。

【構文】

<VB2010>

Public Property **TrialPeriod** As **Integer**

<VB6.0>

Public Property **TrialPeriod** As **Long**

<C#>

```
public int TrialPeriod { set; get; }
```

<VC++>

```
public : long NewtonERvcpp::IActivation::TrialPeriod
```

【解説】

エンドユーザがライセンス認証登録をしなくてもアプリケーションが動作する期間を指定します。

1(日)~365(日)の間で設定できます。

この **TrialPeriod** プロパティを 0 に設定すると、猶予期間は無いことになり、ライセンス認証登録をしない限りアプリケーション(またはアプリケーションの主機能など)が動作しないようにできます。

TrialPeriodName プロパティ**【機能】**

猶予(試用)期間の名称(デフォルト:“猶予(試用)”)を設定します。

【構文】

<VB2010/VB6.0>

Public Property TrialPeriodName As **String**

<C#>

```
public string TrialPeriodName { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation::TrialPeriodName
```

【解説】

たとえば、ActivateStatusDisp メソッドでは認証状態が表示されます。現在、猶予(試用)中ならば、「猶予(試用)期間残日数:6 日」といったように表示されます。その中の「猶予(試用)」という文字列を別の文字列で指定することができます。それがこの TrialPeriodName プロパティです。
(例)“体験版”

UseCpuInfo プロパティ**【機能】**

CPU 情報の使用 (デフォルト: True) を設定します。

次の 3 つのメソッドの処理において、認証情報とエンドユーザ PC 内の CPU 情報とを照合するかどうかを設定します。

- ・[ActivateStatusCheck](#) メソッド (エンドユーザ PC 内での認証状態の確認)
- ・[ActivateStatusCheckOnline](#) メソッド (オンラインで認証状態の確認)
- ・[RestoreRegisterStatus](#) メソッド (認証登録状態回復) 処理の呼び出し)

【構文】

<VB2010/VB6.0>

Public Property **UseCpuInfo** As **Boolean**

<C#>

```
public bool UseCpuInfo { set; get; }
```

<VC++>

```
public : VARIANT_BOOL NewtonenRvcpp::IActivation::UseCpuInfo
```

【解説】

認証レスキュー！の認証 UI ライブラリでは、認証登録など各処理時にエンドユーザ PC の CPU 情報を記録します。等プロパティを True にするとエンドユーザ PC の CPU 情報を認証識別情報として利用します。これを利用することでエンドユーザ PC の不正利用時の認証識別情報の精度を上げます。

※[RestoreRegisterStatus](#) メソッド (認証登録状態回復) 処理の呼び出し) を使用する際は、当プロパティを必ず True に設定してください。

UseMacAddress プロパティ

【機能】

MAC アドレスの使用(デフォルト: True)を設定します。

次の 3 つのメソッドの処理において、認証情報とエンドユーザ PC 内の MAC アドレスとを照合するかどうかを設定します。

- ・[ActivateStatusCheck](#) メソッド(エンドユーザ PC 内での認証状態の確認)
- ・[ActivateStatusCheckOnline](#) メソッド(オンラインで認証状態の確認)
- ・[RestoreRegisterStatus](#) メソッド(「認証登録状態回復」処理の呼び出し)

【構文】

<VB2010/VB6.0>

Public Property UseMacAddress As Boolean

<C#>

```
public bool UseMacAddress { set; get; }
```

<VC++>

```
public : VARIANT_BOOL NewtonenRvcpp::IActivation::UseMacAddress
```

【解説】

認証レスキュー！の認証 UI ライブラリでは、認証登録など各処理時にエンドユーザ PC の MAC アドレスを最大で 5 個記録します。MAC アドレスは LAN ポートなどのネットワーク機器に固有な物理アドレスです。

当プロパティを True にするとエンドユーザ PC の MAC アドレスを認証識別情報として利用します。これを利用することでエンドユーザ PC の不正利用時の認証識別情報の精度を上げます。

※[RestoreRegisterStatus](#) メソッド(「認証登録状態回復」処理の呼び出し)を使用する際は、当プロパティを必ず True に設定してください。

VendorsProductStartRegistryKeyPath プロパティ

【機能】

ベンダアプリケーション開始レジストリキーパスを設定します。

【構文】

<VB2010/VB6.0>

Public Property VendorsProductStartRegistryKeyPath As **String**

<C#>

```
public string VendorsProductStartRegistryKeyPath { set; get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation::VendorsProductStartRegistryKeyPath
```

【解説】

エンドユーザに配布したアプリケーションが認証 UI ライブラリ(DLL)の機能を使う場合、その各種情報の記録先として使うエンドユーザ PC 上のレジストリの開始キーのパスを指定します。レジストリ内の「HKEY_LOCAL_MACHINE」以降を指定します。

(例) "Software¥Newtone¥NinshoRescue¥NR-200¥SampleProject"

なお、物理的なレジストリの位置は動作する貴社のアプリケーションが32bitなのか64bitなのかと、動作するPCのOSが32bitなのか64bitなのかによって異なります。

たとえば、HKEY_LOCAL_MACHINE¥SOFTWARE¥ABCというレジストリパスは次のようになります。32bitOS上で32bitアプリケーションが動作する場合、または64bitOS上で64bitアプリケーションが動作する場合は、

HKEY_LOCAL_MACHINE¥SOFTWARE¥ABC

そのままです。

しかし、64bitOS上で32bitアプリケーションが動作する場合は、

HKEY_LOCAL_MACHINE¥SOFTWARE¥Wow6432Node¥ABC

となります。

また、貴社の異なる複数のアプリケーションをエンドユーザの同一PC上で使用させるには、この VendorsProductStartRegistryKeyPath プロパティにアプリケーションごとのそれぞれ異なるレジストリパスを設定してください。たとえば、次のように設定します。

アプリケーションA内での設定コード例:

```
VendorsProductStartRegistryKeyPath = "Software¥Company¥A"
```

アプリケーションB内での設定コード例:

```
VendorsProductStartRegistryKeyPath = "Software¥Company¥B"
```


WebServiceBasicAuthenticationPassword プロパティ**【機能】**

Web サービス時の基本認証パスワードを設定します。

【構文】

<VB2010/VB6.0>

Public Property **WebServiceBasicAuthenticationPassword** As **String**

<C#>

```
public string WebServiceBasicAuthenticationPassword { set; get; }
```

<VC++>

```
public : _bstr_t
```

```
Newtonsoft::IActivation::WebServiceBasicAuthenticationPassword
```

【解説】

Web サーバーで基本認証を使用する場合に、そのパスワードを指定します。
基本認証に関しては、**WebServiceUseBasicAuthentication** プロパティを参照してください。

WebServiceBasicAuthenticationUserName プロパティ**【機能】**

Web サービス時の基本認証ユーザ名を設定します。

【構文】

<VB2010/VB6.0>

Public Property **WebServiceBasicAuthenticationUserName** As **String**

<C#>

```
public string WebServiceBasicAuthenticationUserName { set; get; }
```

<VC++>

```
public : _bstr_t
```

```
NewtonNrvcpp::IActivation::WebServiceBasicAuthenticationUserName
```

【解説】

Web サーバーで基本認証を使用する場合に、そのユーザ名を指定します。

基本認証に関しては、**WebServiceUseBasicAuthentication** プロパティを参照してください。

WebServiceCheckPassword プロパティ

【機能】

Web サービス確認パスワード(8 文字以上)を設定します。

【構文】

<VB2010/VB6.0>

Public Property **WebServiceCheckPassword** As **String**

<C#>

```
public string WebServiceCheckPassword { set; get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation::WebServiceCheckPassword
```

【解説】

Web サービスを利用する場合の確認用のパスワードを設定します。ここでは、Web サーバー側設定アプリケーション「認証 Web サービス」の環境設定処理の Web サービスの確認パスワードと同じものを設定します。

確認パスワードは必須項目です、省略はできません。8 文字以上で半角の次の文字が使用できません。

- ・大文字の英字 (A～Z)
- ・小文字の英字 (a～z)
- ・数字 (0～9)
- ・記号 (+-*/!#\$%&()=¥@<>?)

WebServiceTimeout プロパティ**【機能】**

Web サービスのタイムアウト(デフォルト: 60 秒)を設定します。

【構文】

<VB2010>

Public Property **WebServiceTimeout** As **Integer**

<VB6.0>

Public Property **WebServiceTimeout** As **Long**

<C#>

```
public int WebServiceTimeout { set; get; }
```

<VC++>

```
public : long NewtoneNRvcpp::IActivation::WebServiceTimeout
```

【解説】

Web サービスに接続して応答を待つ最大時間を秒単位で指定します。初期値は 60 秒です。

WebServiceURL プロパティ**【機能】**

Web サービスの URL を設定します。

【構文】

<VB2010/VB6.0>

Public Property **WebServiceURL** As **String**

<C#>

public **string** **WebServiceURL** { set; get; }

<VC++>

public : **_bstr_t** **NewtoneNRvcpp::IActivation::WebServiceURL**

【解説】

認証に関するシステムを Web サービスとして提供する Web サーバーの URL を指定します。
以下に例を示します。

自 PC のローカルホストの Web サーバー(IIS)にアクセスする例:

“http://localhost/Nr2WebService/Service.asmx”

(注)この例は実際の運用ではありえません。貴社のアプリケーションで認証 UI ライブラリ(DLL)を使用した開発時に、テスト用に使用される URL です。

自社 Web サーバー(IIS)にアクセスさせる例:

“http://www.newtone.co.jp/Nr2WebService/Service.asmx”

クラウドサービス Microsoft Azure の Web アプリ (旧 Web サイト)に配置した Web サービスを利用する例:

“http://newtonecojp.azurewebsites.net/Service.asmx”

| |
|----------------------------------------|
| WebServiceUseBasicAuthentication プロパティ |
|----------------------------------------|

【機能】

Web サービス時の基本認証の使用(デフォルト:False)を設定します。

【構文】

<VB2010/VB6.0>

Public Property **WebServiceUseBasicAuthentication** As **Boolean**

<C#>

```
public bool WebServiceUseBasicAuthentication { set; get; }
```

<VC++>

```
public : VARIANT_BOOL
```

```
NewtonenRvcpp::IActivation::WebServiceUseBasicAuthentication
```

【解説】

Web サーバーで基本認証を使用する場合は、この **WebServiceUseBasicAuthentication** プロパティを True にします。

初期値は、False (基本認証を使用しない) です。

この **WebServiceUseBasicAuthentication** プロパティは、Web サーバー (IIS) 側で特定のアカウント (ユーザ名とパスワード) でアクセスできるフォルダにこの Web サービスが配置してある場合に使用できるセキュリティ設定です。

Web サーバーで基本認証を使用する一般的な手順は次の通りです。

1. サーバー PC 上でユーザを作成。

この際のユーザ名とパスワードがそのまま基本認証に使われます。

2. 基本認証フォルダのセキュリティ設定

フォルダのプロパティを開き、セキュリティタブで上記 1 のユーザ名を 追加し、「読み取り」権限を付与します。

3. IIS でのセキュリティ設定

IIS で該当フォルダに対し「認証」の設定で「匿名認証」を無効にして 「基本認証」を有効にします。

なお、Web サーバーでの基本認証の詳細につきましてはマイクロソフト社の関連ドキュメントをご覧ください。

<UI系メソッド>

メソッド一覧

| メソッド | 機能 |
|------------------------------------------------------------|-------------------------------|
| ActivateRegisterInternet | 「認証登録/インターネット」処理の呼び出し |
| ActivateRegisterTelephone | 「認証登録/電話」処理の呼び出し |
| ActivateRemoveInternet | 「認証解除/インターネット」処理の呼び出し |
| ActivateRemoveTelephone | 「認証解除/電話」処理の呼び出し |
| ActivateStatusCheck | エンドユーザ PC 内での認証状態の確認 |
| ActivateStatusCheckOnline | オンラインで認証状態の確認 |
| ActivateStatusDisp | 「認証状態表示」処理の呼び出し |
| DeterminationOfProxyUpdateOfExpirationDate | 「代理有効期限更新/確定」処理の呼び出し |
| GetProxyUpdateOfExpirationDate | 「代理有効期限/取得」処理の呼び出し(代理 PC で利用) |
| PreparationOfProxyUpdateOfExpirationDate | 「代理有効期限更新/準備」処理の呼び出し |
| ProxyActivateRegisterExecute | 「代理認証登録/実行」処理の呼び出し(代理 PC で利用) |
| ProxyActivateRegisterFix | 「代理認証登録/確定」処理の呼び出し |
| ProxyActivateRegisterPrepare | 「代理認証登録/準備」処理の呼び出し |
| ProxyActivateRemoveExecute | 「代理認証解除/実行」処理の呼び出し(代理 PC で利用) |
| ProxyActivateRemovePrepare | 「代理認証解除/準備」処理の呼び出し |
| RestoreCancelStatus | 「認証解除状態回復」処理の呼び出し |
| RestoreRegisterStatus | 「認証登録状態回復」処理の呼び出し |
| UpdateOfExpirationDate | 「有効期限の更新」処理の呼び出し |

ActivateRegisterInternet メソッド

【機能】

「認証登録/インターネット」処理の呼び出しを行います。

【構文】

<VB2010/VB6.0>

Public Function **ActivateRegisterInternet()** As **Boolean**

<C#>

public **bool** **ActivateRegisterInternet()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::ActivateRegisterInternet()**

【引数】

なし

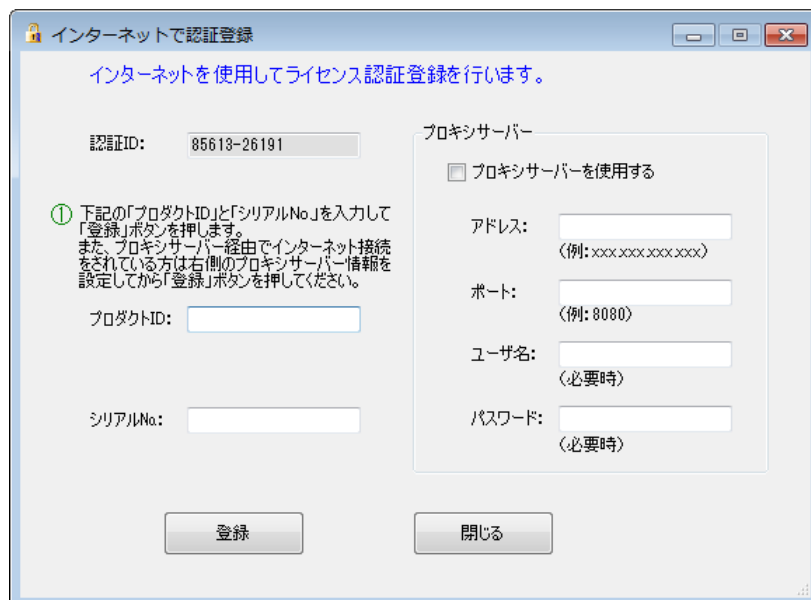
【戻り値】

True: 正常

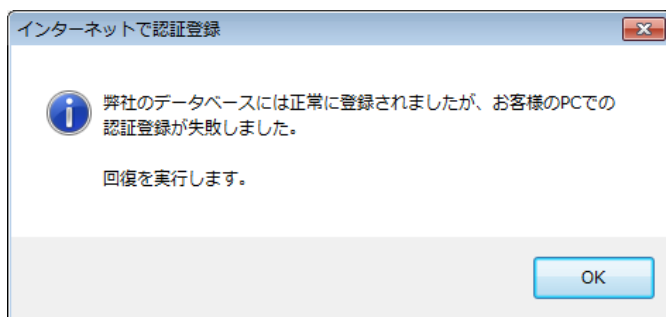
False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

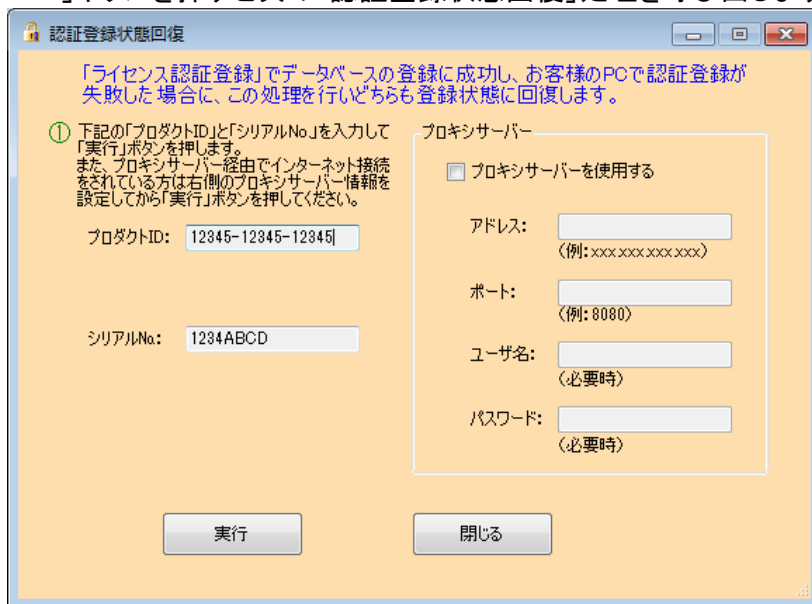
「認証登録/インターネット」処理の呼び出しを行います。



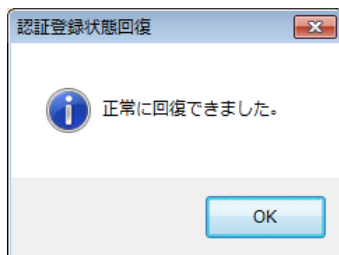
なお、何らかの原因で、データベースは登録済、エンドユーザ PC のレジストリが解除済の状態になった場合は、次のようなダイアログが表示されます。



「OK」ボタンを押すと次の「認証登録状態回復」処理を呼び出します。



「実行」ボタンを押して回復が成功すると次のダイアログが表示されます。



ActivateRegisterTelephone メソッド

【機能】

「認証登録/電話」処理の呼び出しを行います。

【構文】

<VB2010/VB6.0>

Public Function **ActivateRegisterTelephone()** As **Boolean**

<C#>

public **bool** **ActivateRegisterTelephone()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::ActivateRegisterTelephone()**

【引数】

なし

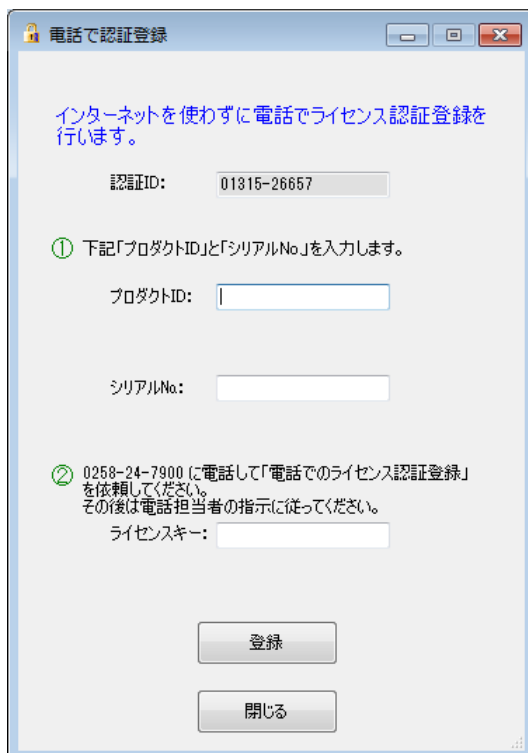
【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

「認証登録/電話」処理の呼び出しを行います。



ActivateRemoveInternet メソッド

【機能】

「認証解除/インターネット」処理の呼び出しを行います。

【構文】

<VB2010/VB6.0>

Public Function **ActivateRemoveInternet()** As **Boolean**

<C#>

public **bool** **ActivateRemoveInternet()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::ActivateRemoveInternet()**

【引数】

なし

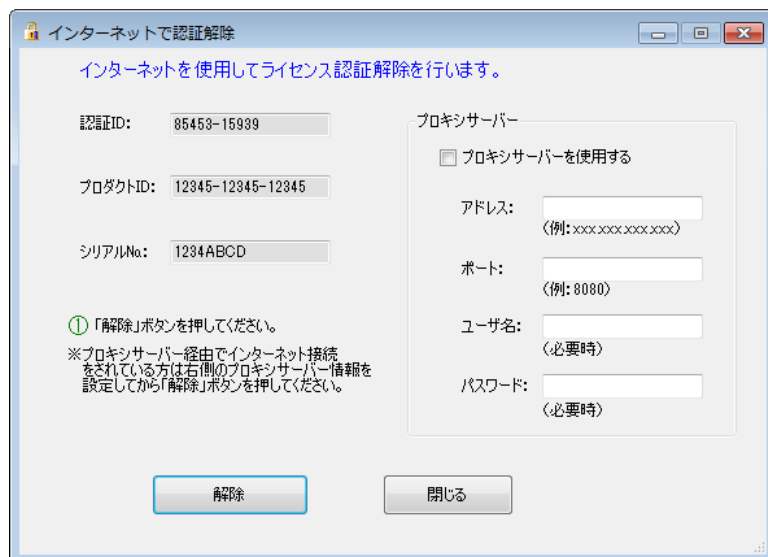
【戻り値】

True: 正常

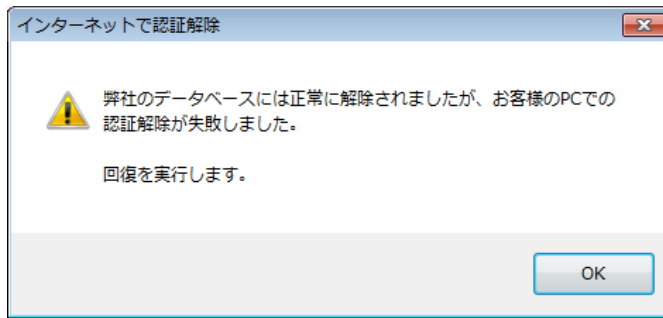
False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

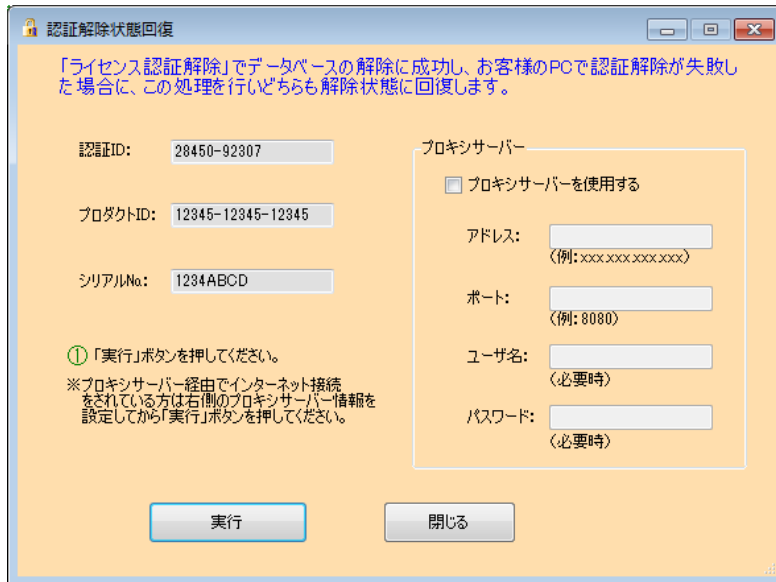
「認証解除/インターネット」処理の呼び出しを行います。



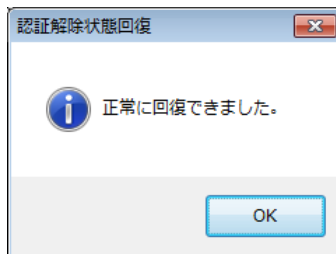
なお、何らかの原因で、データベースは解除済、エンドユーザ PC のレジストリが登録状態になった場合は、次のようなダイアログが表示されます。



「OK」ボタンを押すと次の「認証解除状態回復」処理を呼び出します。



「実行」ボタンを押して回復が成功すると次のダイアログが表示されます。



ActivateRemoveTelephone メソッド

【機能】

「認証解除/電話」処理の呼び出しを行います。

【構文】

<VB2010/VB6.0>

Public Function **ActivateRemoveTelephone()** As **Boolean**

<C#>

public **bool** **ActivateRemoveTelephone()**

<VC++>

public : **VARIANT_BOOL** **NewtonENRvcpp::IActivation::ActivateRemoveTelephone()**

【引数】

なし

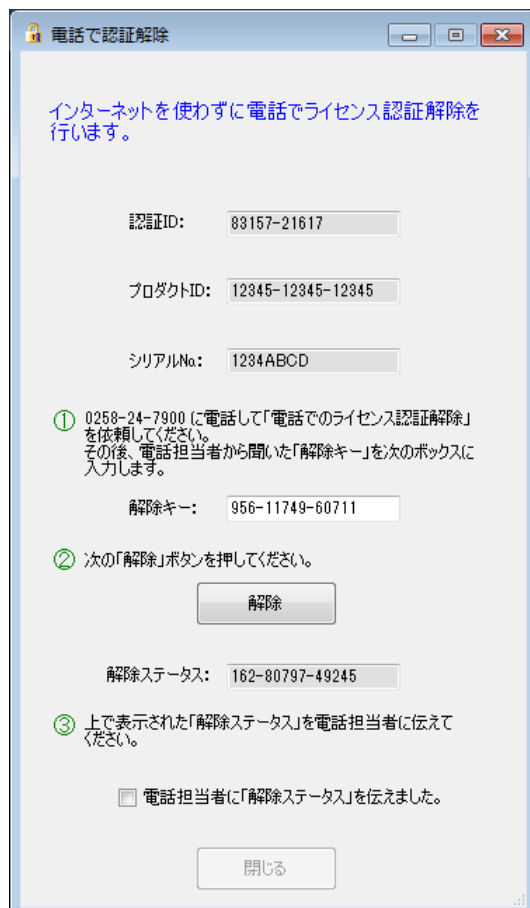
【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

「認証解除/電話」処理の呼び出しを行います。



| |
|--------------------------|
| ActivateStatusCheck メソッド |
|--------------------------|

【機能】

エンドユーザ PC 内での認証状態の確認を行います。

【構文】

<VB2010>

Public Function **ActivateStatusCheck()** As **Integer**

<VB6.0>

Public Function **ActivateStatusCheck()** As **Long**

<C#>

```
public int ActivateStatusCheck()
```

<VC++>

```
public : long NewtonenNRvcpp::IActivation::ActivateStatusCheck()
```

【引数】

なし

【戻り値】

| | |
|----------------------|-------------------------------------|
| 0: | 猶予期限切れ(猶予有効時) |
| 1~365: | 猶予日数有 |
| 366: | 日付データの取得失敗(猶予) |
| 400: | 未認証(猶予無効時) |
| 500: | 認証済み |
| 1000: | レンタル期限切れ(レンタル有効時) |
| 1001~2100: | レンタル残日数 1~1100(戻り値から 1000 を引いて使用する) |
| 2101: | 日付データの取得失敗(レンタル) |
| -999: | 認証済ハードウェア情報不一致 |
| -1: | エラー(未設定や範囲を超えているプロパティがある) |
| -21001231~-20000101: | 終了した有効期限(2000/01/01~2100/12/31) |
| -21001232: | 日付データの取得失敗(有効期限) |
| -3: | PC の日付が変更されました。 |
| 20000101~21001231: | まだ有効な有効期限(2000/01/01~2100/12/31) |

<戻り値が-1 の場合の補足説明>

本来設定が必須であるプロパティに値がセットされていない場合やセットした値が無効な場合などに(-1)が返ります。たとえば、「猶予(試用)期間の名称」用の TrialPeriodName プロパティに(猶予・試用期間機能を使用しないということで未設定にして結果として)空文字列を設定した場合や電話で認証時の電話番号用の TelephoneNumber プロパティに(電話対応機能を使用しないということで未設定にして結果として)空文字列を設定した場合などにこの戻り値(-1)が返ります。DLL のプロパティは機能の使用有無に関係なく DLL の起動時にそれらの値をチェックするようになっていて、上記のような空文字列などの場合その機能の利用時に問題が発生することをあらかじめ防止する目的で戻り値(-1)が返ります。

【解説】

この ActivateStatusCheck メソッドと ActivateStatusCheckOnline メソッドの相違点は ActivateStatusCheck メソッドがエンドユーザ PC 内での認証状況を返すのに対し、ActivateStatusCheckOnline メソッドはインターネットを介し貴社のデータベースにアクセスして取得した情報とエンドユーザ PC 内の情報とを照合して認証状況を返す点です。

| |
|--------------------------------|
| ActivateStatusCheckOnline メソッド |
|--------------------------------|

【機能】

オンラインで認証状態の確認を行います。

【構文】

<VB2010>

Public Function **ActivateStatusCheckOnline()** As **Integer**

<VB6.0>

Public Function **ActivateStatusCheckOnline()** As **Long**

<C#>

public **int** **ActivateStatusCheckOnline()**

<VC++>

public : **long** **NewtonenRvcpp::IActivation::ActivateStatusCheckOnline()**

【引数】

なし

【戻り値】

- 0: PC レベルで認証されていない(PC のレジストリには認証登録情報がない)
- 1: OK(PC レベルと DB で認証登録情報が一致した)
- 2: NG(PC レベルと一致する認証登録情報が DB がない)
- 3: NG(認証済ハードウェア情報不一致)
- 4: NG(日付データの取得失敗(猶予))
- 5: NG(日付データの取得失敗(レンタル))
- 6: NG(日付データの取得失敗(有効期限))
- 11: 接続できない(認証時は電話)
- 12: 接続できない(認証時は代理)
- 999: その他エラー(接続できないなど)
- 1: エラー(未設定や範囲を超えているプロパティがある)
- 3: PC の日付が変更されました。

<戻り値が-1 の場合の補足説明>

本来設定が必須であるプロパティに値がセットされていない場合やセットした値が無効な場合などに(-1)が返ります。たとえば、「猶予(試用)期間の名称」用の TrialPeriodName プロパティに(猶予・試用期間機能を使用しないということで未設定にして結果として)空文字列を設定した場合や電話で認証時の電話番号用の TelephoneNumber プロパティに(電話対応機能を使用しないということで未設定にして結果として)空文字列を設定した場合などにこの戻り値(-1)が返ります。DLL のプロパティは機能の使用有無に関係なく DLL の起動時にそれらの値をチェックするようになっていて、上記のような空文字列などの場合その機能の利用時に問題が発生することをあらかじめ防止する目的で戻り値(-1)が返ります。

【解説】

この ActivateStatusCheckOnline メソッドと ActivateStatusCheck メソッドとの相違点は ActivateStatusCheck メソッドがエンドユーザ PC 内での認証状況を返すのに対し、ActivateStatusCheckOnline メソッドはインターネットを介し貴社のデータベースにアクセスして取得

した情報とエンドユーザ PC 内の情報とを照合して認証状況を返す点です。

この ActivateStatusCheckOnline メソッドは、なんらかの理由で現在エンドユーザが使用している貴社のアプリケーションを使用不可としたい場合などに利用できます。

通常の(レンタル期間がない)認証登録の場合、レジストリとハード情報だけの確認で OK となってしまう貴社がデータベース(DB)上の当該情報を削除してもエンドユーザの PC ではアプリケーションが継続して使用できてしまいます。そこで、任意のタイミングでインターネットを介し貴社のデータベースにアクセスしてそれらの情報を確認できる機能がこのメソッドです。貴社はたとえば、アプリケーションの起動時にそのメソッドを利用するコードを記述し、その戻り値によってアプリケーションを(強制的に)終了する、といった挙動を制御できます。

インターネットを介し Web サービスに接続できなかった場合は、次の「インターネットに接続」ダイアログを表示します。

このダイアログでのエンドユーザの選択肢は次の通りです。

- ・「接続」ボタン→現在の内容で再度接続する。
- ・「キャンセル」ボタン→このダイアログを閉じて、当該メソッドの戻り値として「-999:その他エラー(接続できないなど)」を返してメソッド終了

なお、このダイアログ内で必要に応じてプロキシサーバーの接続設定が可能です。

| |
|-------------------------|
| ActivateStatusDisp メソッド |
|-------------------------|

【機能】

「認証状態表示」処理の呼び出しを行います。

【構文】

<VB2010/VB6.0>

Public Function **ActivateStatusDisp**() As **Boolean**

<C#>

public **bool** **ActivateStatusDisp**()

<VC++>

public : **VARIANT_BOOL** **NewtonenRvcpp::IActivation::ActivateStatusDisp**()

【引数】

なし

【戻り値】

True: 正常

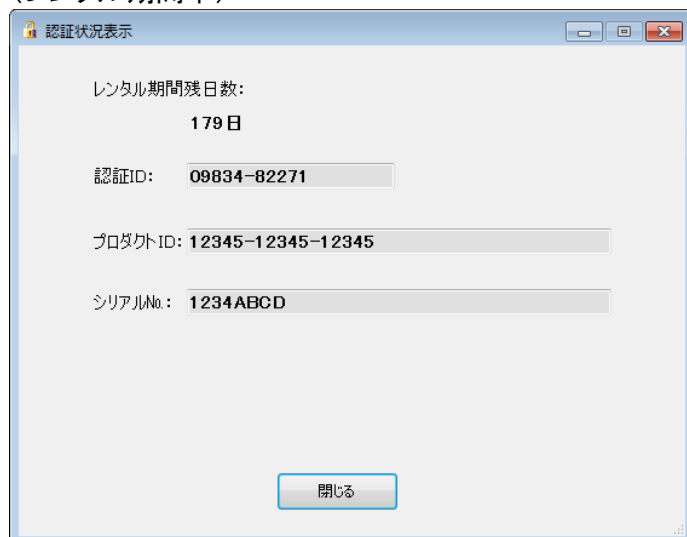
False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

エンドユーザ PC 内での認証状態の表示を行います。

呼び出したダイアログの例を以下に示します。

(レンタル期間中)



DeterminationOfProxyUpdateOfExpirationDate メソッド

【機能】

「代理有効期限更新/確定」処理の呼び出しを行います。
 (代理有効期限更新機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function **DeterminationOfProxyUpdateOfExpirationDate()** As **Boolean**

<C#>

public **bool** **DeterminationOfProxyUpdateOfExpirationDate()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::**
DeterminationOfProxyUpdateOfExpirationDate()

【引数】

なし

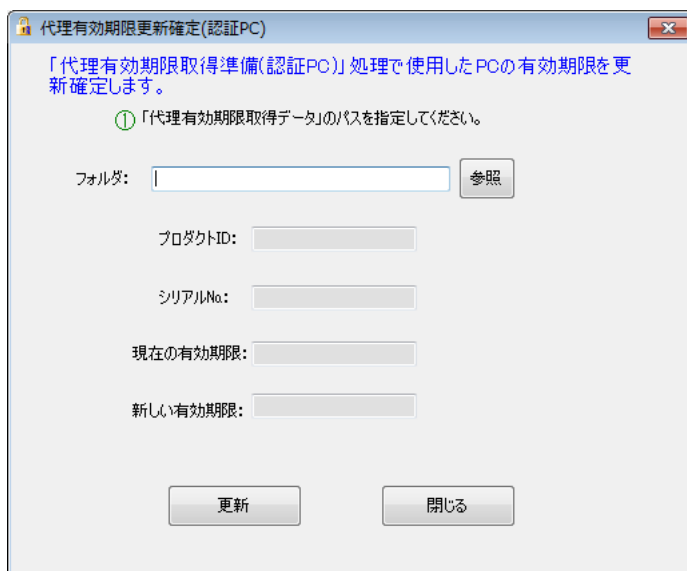
【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

「代理有効期限更新/確定」処理の呼び出しを行います。



GetProxyUpdateOfExpirationDate メソッド

【機能】

「代理有効期限/取得」処理の呼び出し(代理 PC で利用)を行います。
(代理有効期限更新機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function GetProxyUpdateOfExpirationDate () As Boolean

<C#>

public bool GetProxyUpdateOfExpirationDate ()

<VC++>

public : VARIANT_BOOL NewtonenRvcpp::IActivation::

GetProxyUpdateOfExpirationDate ()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

「代理有効期限/取得」処理の呼び出し(代理 PC で利用)を行います。

PreparationOfProxyUpdateOfExpirationDate メソッド

【機能】

「代理有効期限更新/準備」処理の呼び出しを行います。
 (代理有効期限更新機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function PreparationOfProxyUpdateOfExpirationDate() As **Boolean**

<C#>

public **bool** PreparationOfProxyUpdateOfExpirationDate ()

<VC++>

public : **VARIANT_BOOL** NewtonenRvcpp::IActivation::

PreparationOfProxyUpdateOfExpirationDate()

【引数】

なし

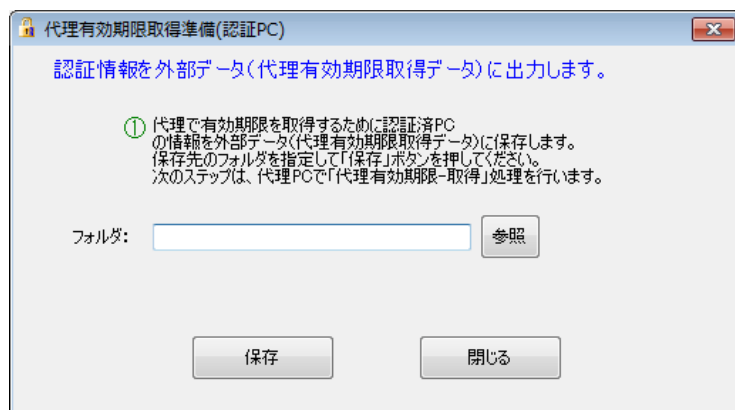
【戻り値】

True: 正常

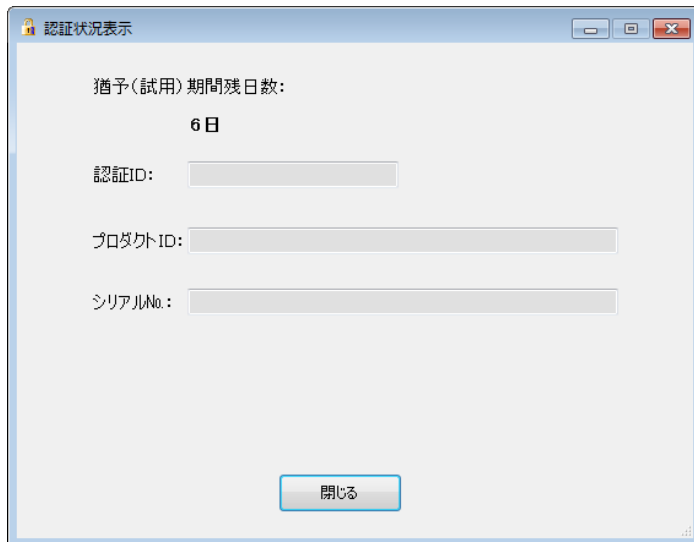
False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

「代理有効期限更新/準備」処理の呼び出しを行います。



(猶予期間中)



認証状況表示

猶予(試用)期間残日数:
6日

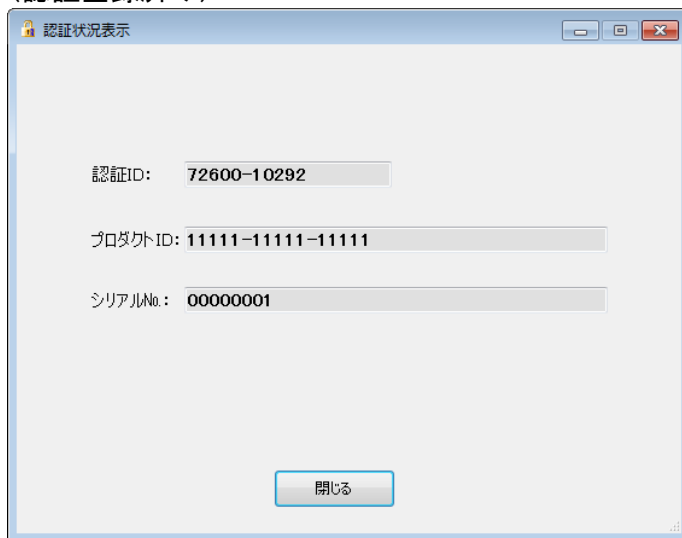
認証ID:

プロダクトID:

シリアルNo.:

閉じる

(認証登録済み)



認証状況表示

認証ID: 72600-10292

プロダクトID: 11111-11111-11111

シリアルNo.: 00000001

閉じる

(有効期限内)



認証状況表示

有効期限:
2016年05月30日まで

認証ID: 20333-47848

プロダクトID: 12345-12345-12345

シリアルNo.: 1234ABCD

閉じる

ProxyActivateRegisterExecute メソッド

【機能】

「代理認証登録/実行」処理の呼び出し(代理 PC で利用)を行います。
(代理認証機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function ProxyActivateRegisterExecute() As **Boolean**

<C#>

public **bool** ProxyActivateRegisterExecute()

<VC++>

public : **VARIANT_BOOL**

NewtonenRvcpp::IActivation::ProxyActivateRegisterExecute()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

「代理認証登録/実行」処理の呼び出し(代理 PC で利用)を行います。

ProxyActivateRegisterFix メソッド

【機能】

「代理認証登録/確定」処理の呼び出しを行います。
 (代理認証機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function ProxyActivateRegisterFix() As Boolean

<C#>

public bool ProxyActivateRegisterFix()

<VC++>

public : VARIANT_BOOL NewtonenRvcpp::IActivation::ProxyActivateRegisterFix()

【引数】

なし

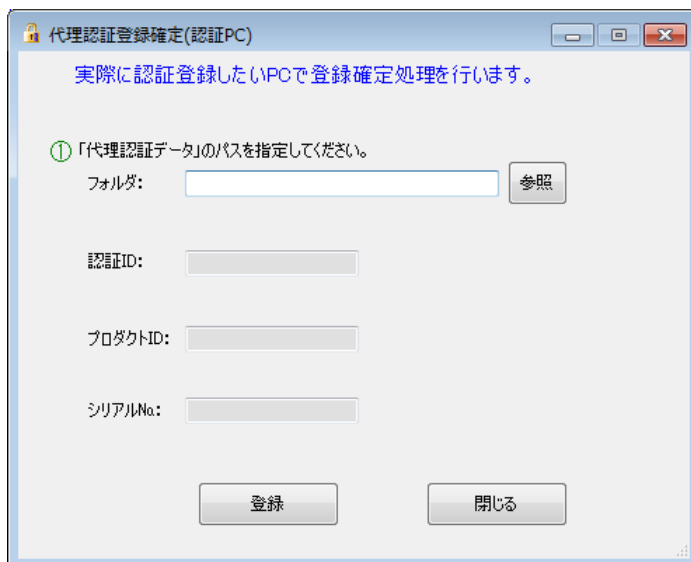
【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

「代理認証登録/確定」処理の呼び出しを行います。



ProxyActivateRegisterPrepare メソッド

【機能】

「代理認証登録/準備」処理の呼び出しを行います。
 (代理認証機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function ProxyActivateRegisterPrepare() As **Boolean**

<C#>

public **bool** ProxyActivateRegisterPrepare()

<VC++>

public : **VARIANT_BOOL**

NewtonenRvcpp::IActivation::ProxyActivateRegisterPrepare()

【引数】

なし

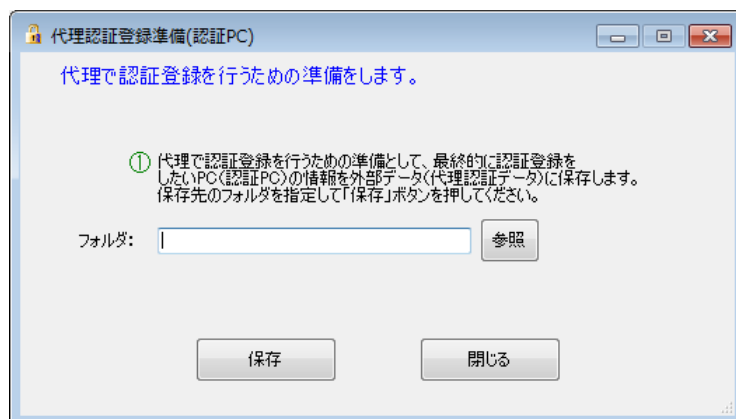
【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

「代理認証登録/準備」処理の呼び出しを行います。



ProxyActivateRemoveExecute メソッド

【機能】

「代理認証解除/実行」処理の呼び出し(代理 PC で利用)を行います。
(代理認証機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function ProxyActivateRemoveExecute() As Boolean

<C#>

public bool ProxyActivateRemoveExecute()

<VC++>

public : VARIANT_BOOL

NewtonNrvcpp::IActivation::ProxyActivateRemoveExecute()

【引数】

なし

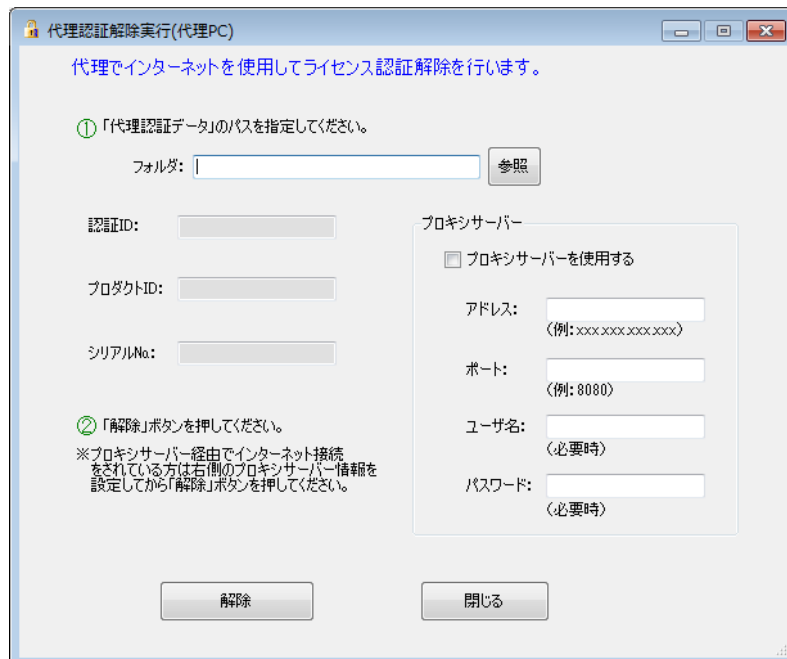
【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

「代理認証解除/実行」処理の呼び出し(代理 PC で利用)を行います。



ProxyActivateRemovePrepare メソッド

【機能】

「代理認証解除/準備」処理の呼び出しを行います。
 (代理認証機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function ProxyActivateRemovePrepare() As Boolean

<C#>

public bool ProxyActivateRemovePrepare()

<VC++>

public : VARIANT_BOOL

NewtonenRvcpp::IActivation::ProxyActivateRemovePrepare()

【引数】

なし

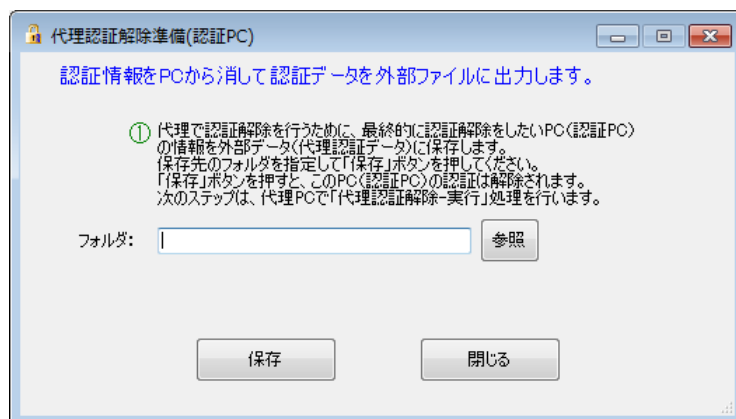
【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

「代理認証解除/準備」処理の呼び出しを行います。



RestoreCancelStatus メソッド

【機能】

「認証解除状態回復」処理の呼び出しを行います。

【構文】

<VB2010/VB6.0>

Public Function **RestoreCancelStatus()** As **Boolean**

<C#>

public **bool** RestoreCancelStatus()

<VC++>

public : **VARIANT_BOOL** NewtonenRvcpp::IActivation::RestoreCancelStatus()

【引数】

なし

【戻り値】

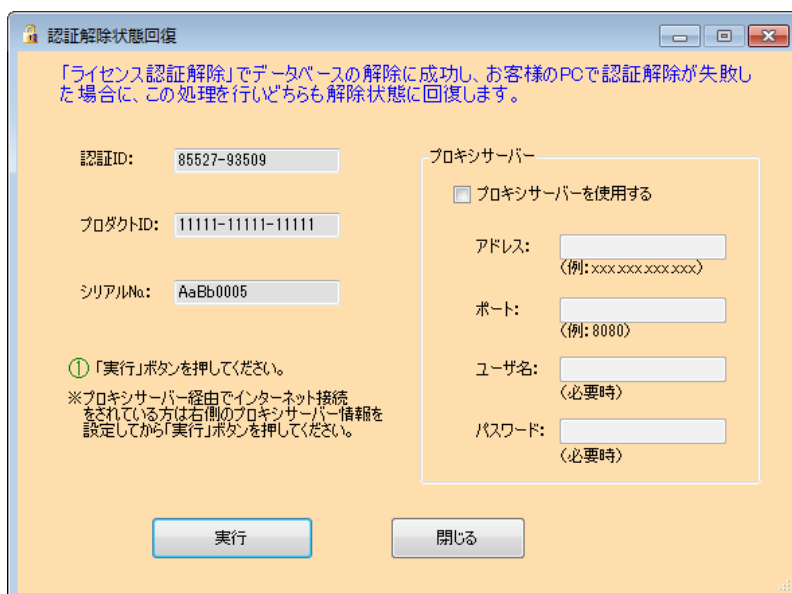
True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

「認証解除状態回復」処理の呼び出しを行います。

「ライセンス認証解除」でデータベースの解除に成功したが、エンドユーザ PC での認証解除が失敗した状態になった場合に、このメソッドによる処理でエンドユーザの操作でその状態を回復しエンドユーザ PC を認証解除状態とします。



RestoreRegisterStatus メソッド

【機能】

「認証登録状態回復」処理の呼び出しを行います。

【構文】

<VB2010/VB6.0>

Public Function **RestoreRegisterStatus()** As **Boolean**

<C#>

public **bool** RestoreRegisterStatus()

<VC++>

public : **VARIANT_BOOL** NewtonenNRvcpp::IActivation::RestoreRegisterStatus()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

「認証登録状態回復」処理の呼び出しを行います。

「ライセンス認証登録」でデータベースの登録に成功したが、エンドユーザ PC での認証登録が失敗した状態になった場合に、このメソッドによる処理でエンドユーザの操作でその状態を回復しエンドユーザ PC を認証登録状態とします。

UpdateOfExpirationDate メソッド

【機能】

「有効期限の更新」処理の呼び出しを行います。

【構文】

<VB2010/VB6.0>

Public Function **UpdateOfExpirationDate()** As **Boolean**

<C#>

public **bool** UpdateOfExpirationDate()

<VC++>

public : **VARIANT_BOOL** NewtonenRvcpp::IActivation::UpdateOfExpirationDate()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

【解説】

「有効期限の更新」処理の呼び出しを行います。

プロダクト ID とシリアル No.が有効期限によるライセンスの場合、当処理を利用してエンドユーザに有効期限を更新させることができます。

この処理をエンドユーザに実行してもらう前に、貴社で新しい有効期限の設定をする必要があります。

有効期限の設定は認証管理システムの「認証キー編集(表形式)」処理を利用します。

なお、エンドユーザに有効期限を更新させる方法として当処理を実行させる他に、エンドユーザに一度認証の解除後、再度認証の登録をしてもらうことでも更新が完了します。

エンドユーザが代理認証を利用している場合で、有効期限によるライセンスを更新する場合はその方法を使います。

<API 系プロパティ>

プロパティ一覧

| プロパティ | 機能 |
|----------------------------------------------------------|-------------------------------------------------------------------------------------|
| APIError 列挙体 | エラー内容を表示します。 |
| APICertificationID | 認証 ID (取得専用) |
| APICurrentExpirationDate | 現在の有効期限(取得専用) |
| APIEncryptionPassword | 暗号化時のパスワードを設定 |
| APIEncryptionSaltString | 暗号化時の Salt 文字列を設定 |
| APIErrorStatus | API 系のメソッドを使用した際のエラーの内容を返す。(取得専用) |
| APIExternalLinkKey | APIGetProductIdSerialNoList メソッド 実行時の外部データベースとのリンク用キー項目を設定 |
| APIFreeItem1~5 | APIGetFreeItem メソッド 実行後に自由入力項目 1~5 が設定される(取得専用) |
| APILicenseKey | ライセンスキー |
| APINewExpirationDate | 新しい有効期限を取得します。(取得専用) |
| APIOverwriteModeOfExpirationDateUpdate | 「有効期限の更新」実行時の有効期限新旧上書きの設定 |
| APIProductID | プロダクト ID |
| APIProductIdSerialNoList | APIGetProductIdSerialNoList メソッド実行後にプロダクト ID とシリアル No.のペアをデリミタで列挙したの文字列が設定される(取得専用) |
| APIProxyDataPath | 代理認証データパス |
| APIProxyServerAddress | プロキシサーバーのアドレス |
| APIProxyServerPassword | プロキシサーバーのパスワード |
| APIProxyServerPort | プロキシサーバーのポート |
| APIProxyServerUserName | プロキシサーバーのユーザ名 |
| APIReleaseKey | 解除キー |
| APIReleaseStatus | 解除ステータス(取得専用) |
| APIRentalPeriod | レンタル日数を設定 |
| APISelectRunAppDatePathFlag | 「アプリ起動日」の読み込み/書き込み場所の切り替えフラグ |
| APISerialNo | シリアル No. |
| APITrialPeriod | 猶予(試用)日数を設定 |
| APIUseCpuInfo | CPU 情報の使用を設定 |
| APIUseMacAddress | MAC アドレスの使用を設定 |
| APIUseProxyServer | プロキシサーバーの使用区分 |
| APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパスを設定 |
| APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワードを設定 |
| APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名を設定 |
| APIWebServiceCheckPassword | Web サービス確認パスワードを設定 |
| APIWebServiceTimeout | Web サービスのタイムアウトを設定 |
| APIWebServiceURL | Web サービスの URL を設定 |
| APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用を設定 |

APIError 列挙体

エラー内容を表示します。

```
public enum APIError
パブリックメンバ
```

| メンバ名 | 値 | 内容 | 関連するメソッド |
|----------------------------------|-----|-------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| None | 0 | エラーなし | |
| OutOfMemoryException | 11 | プログラムの実行を継続するためのメモリが不足している場合にスローされる例外。 | |
| StackOverflowException | 12 | 入れ子になったメソッド呼び出しが多くなりすぎ、実行スタックがオーバーフローした場合にスローされる例外。このクラスは継承できません。 | |
| UnauthorizedAccessException | 13 | オペレーティング システムが I/O エラーまたは特定の種類のセキュリティエラーのためにアクセスを拒否する場合、スローされる例外。 | |
| IoDirectoryNotFoundExcep tion | 14 | ファイルまたはディレクトリの一部が見つからない場合にスローされる例外。 | |
| IoDriveNotFoundException | 15 | 使用できないドライブまたは共有にアクセスしようとするとスローされる例外。 | |
| IoEndOfStreamException | 16 | ストリームの末尾を越えて読み取ろうとしたときにスローされる例外。 | |
| IoFileLoadException | 17 | マネージ アセンブリが見つかったが、読み込むことができない場合にスローされる例外。 | |
| IoFileNotFoundException | 18 | ディスク上に存在しないファイルにアクセスしようとして失敗したときにスローされる例外。 | |
| IoIOException | 19 | I/O エラーが発生したときにスローされる例外。 | |
| IoPathTooLongException | 20 | パス名またはファイル名がシステム定義の最大長を超えている場合にスローされる例外。 | |
| BadStrInProductID | 101 | プロダクト ID に不正な文字が含まれています。 | APIActivateRegisterInternet APIActivateRegisterTelephone APIProxyActivateRegisterExecute APIRestoreRegisterStatus APIUpdateOfExpirationDate APIGetFreeItem APIGetRegisteredI |

| | | | |
|------------------|-----|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | nfoFromWeb APIGetProxyUpdateOfExpirationDate |
| BadStrInSerialNo | 102 | シリアル No.に不正な文字が含まれています。 | APIActivateRegisterInternet APIActivateRegisterTelephone APIProxyActivateRegisterExecute APIRestoreRegisterStatus APIUpdateOfExpirationDate APIGetFreeItem APIGetRegisteredInfoFromWeb APIGetProxyUpdateOfExpirationDate |
| BadInputData | 103 | 入力されたデータが不正です。 | APIActivateRegisterInternet APIActivateRegisterTelephone APIProxyActivateRegisterExecute APIProxyActivateRegisterFix APIProxyActivateRemoveExecute |
| FailedEnableNIC | 104 | NIC の設定に失敗しました。(有効化) | APIActivateRegisterInternet APIActivateRegisterTelephone APIActivateRemoveInternet APIProxyActivateRegisterFix APIProxyActivateRegisterPrepare APIRestoreRegisterStatus |
| FailedDisableNIC | 105 | NIC の設定に失敗しました。(無効化) | APIActivateRegisterInternet APIActivateRegisterTelephone APIActivateRemoveInternet APIProxyActivateRegisterFix APIProxyActivateRegisterPrepare |

| | | | |
|---------------------------|-----|------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | APIRestoreRegisterStatus |
| NoMACAddress | 106 | MAC アドレスが1つも取得できませんでした。 | APIActivateRegisterInternet APIActivateRegisterTelephone APIActivateRemoveInternet APIProxyActivateRegisterFix APIProxyActivateRegisterPrepare APIRestoreRegisterStatus |
| ActivationFailedException | 107 | 弊社のデータベースには正常に登録されましたが、お客様の PC での認証登録が失敗しました。 | APIActivateRegisterInternet |
| ExceedLicense | 108 | ライセンス数を超過しています。 | APIActivateRegisterInternet APIProxyActivateRegisterExecute |
| UnregisteredException | 109 | 指定されたプロダクト ID とシリアルNo. は弊社データベースに登録されていません。 | APIActivateRegisterInternet APIProxyActivateRegisterExecute APIUpdateOfExpirationDate APIGetFreeItem APIGetRegisteredInfoFromWeb APIGetProxyUpdateOfExpirationDate |
| NotRegisteredException | 110 | 何らかの原因で登録できませんでした。 | APIActivateRegisterInternet APIProxyActivateRegisterExecute APIUpdateOfExpirationDate APIGetRegisteredInfoFromWeb APIGetProxyUpdateOfExpirationDate |
| AlreadyRegistered | 111 | そのプロダクト ID + シリアルNo. + 認証 ID は既に登録されているので登録できませんでした。 | APIActivateRegisterInternet APIProxyActivateRegisterExecute |
| BadCheckPassword | 112 | 確認パスワードが不正です。 | APIActivateRegisterInternet APIProxyActivateRegisterExecute |

| | | | |
|--------------------------|-----|--------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | <p>APIActivateRemoveInternet APIProxyActivateRemoveExecute APIRestoreCancelStatus APIRestoreRegisterStatus APIUpdateOfExpirationDate APIGetProductIdSerialNoList APIGetFreeItem APIGetRegisteredInfoFromWeb APIGetProxyUpdateOfExpirationDate</p> |
| BadWaiFile | 113 | <p>WebServEnv.wai ファイルに問題があります。 認証Web サービス実行PC上で「認証レスキュー！2 Web 環境設定」(WebAdmin.exe)が行われていない可能性があります。</p> | <p>APIActivateRegisterInternet APIProxyActivateRegisterExecute APIActivateRemoveInternet APIProxyActivateRemoveExecute APIRestoreCancelStatus APIRestoreRegisterStatus APIUpdateOfExpirationDate APIGetProductIdSerialNoList APIGetFreeItem APIGetRegisteredInfoFromWeb APIGetProxyUpdateOfExpirationDate</p> |
| NotRetrievedTotalLicense | 114 | <p>EU 認証済みライセンス総数が取得できませんでした。</p> | <p>APIActivateRegisterInternet APIProxyActivateRegisterExecute APIGetRegisteredInfoFromWeb</p> |
| BadNRRegistrationLicense | 115 | <p>NR 登録ライセンスに問題があります。 弊社までご連絡ください。</p> | <p>APIActivateRegisterInternet APIProxyActivateRegisterExecute APIGetRegisteredInfoFromWeb</p> |

| | | | |
|----------------------------|-----|-------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ExpirationDateIsOutOfRange | 116 | 弊社のデータベース上の「有効期限」が無効(範囲外など)になっていて更新できませんでした。 弊社にご連絡ください。 | APIActivateRegisterInternet APIProxyActivateRegisterExecute APIRestoreRegisterStatus APIUpdateOfExpirationDate APIGetProxyUpdateOfExpirationDate |
| ExpirationDateIsOldDate | 117 | 弊社のデータベース上の「有効期限」が無効(前日以前)になっていて更新できませんでした。 弊社にご連絡ください。 | APIActivateRegisterInternet APIProxyActivateRegisterExecute APIRestoreRegisterStatus APIUpdateOfExpirationDate APIGetProxyUpdateOfExpirationDate |
| Authenticated | 118 | 既に認証済のため、この処理は無効です。 | APIActivateRegisterInternet APIActivateRegisterTelephone APIProxyActivateRegisterFix APIProxyActivateRegisterPrepare APIRestoreRegisterStatus APIGenerationOfNewCertificationID |
| UnhandledProcess | 119 | 何らかの原因のため、この処理は無効です。 | APIActivateRegisterInternet APIActivateRegisterTelephone APIActivateRemoveInternet APIActivateRemoveTelephone APIProxyActivateRegisterFix APIProxyActivateRegisterPrepare APIProxyActivateRemovePrepare APIRestoreCancelStatus APIRestoreRegisterStatus |

| | | | |
|------------------------|-----|-----------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | <p>APIUpdateOfExpirationDate APIGenerationOfNewCertificationID APIGetRegisteredInfoFromRegistry APIGetProductIdSerialNoList APIGetFreeItem APIPreparationOfProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate</p> |
| UnexpectedError | 120 | <p>認証状況確認中に想定外のエラーが発生しました”</p> | <p>APIActivateRegisterInternet APIActivateRegisterTelephone APIActivateRemoveInternet APIActivateRemoveTelephone APIProxyActivateRegisterFix APIProxyActivateRegisterPrepare APIProxyActivateRemovePrepare APIRestoreCancelStatus APIRestoreRegisterStatus APIUpdateOfExpirationDate APIGenerationOfNewCertificationID APIGetRegisteredInfoFromRegistry APIPreparationOfProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate</p> |
| NoExpirationDateIsUsed | 121 | <p>このプロダクト ID とシリアル No.には有効期限の利用設定はありません。</p> | <p>APIUpdateOfExpirationDate APIPreparationOfProxyUpdateOfExpirationDate</p> |

| | | | |
|------------------------------------------------|-----|--------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | APIGetProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate |
| NotAllowedByRental | 122 | レンタルでご利用の場合は、この処理はできません。 | APIActivateRegisterTelephone |
| BadLicenseKey | 123 | 入力されたデータが不正です。 | APIActivateRegisterTelephone |
| ActivationFailed | 124 | お客様の PC での認証登録が失敗しました。 再度、当処理を実行してください。 | APIActivateRegisterTelephone APIProxyActivateRegisterFix |
| DeauthorizeFailedException | 125 | 弊社のデータベースには正常に解除されましたが、お客様の PC での認証解除が失敗しました。 | APIActivateRemoveInternet |
| ProductIDAndSerialNoAndCertificationIDNotFound | 126 | 指定されたプロダクト ID/シリアルNo./認証 ID は弊社データベースに登録されていません。 | APIActivateRemoveInternet APIProxyActivateRemoveExecute |
| NotDeauthorized | 127 | 何らかの原因で解除できませんでした。 | APIActivateRemoveInternet APIProxyActivateRemoveExecute |
| UnauthenticatedByTrialPeriodOut | 128 | 認証登録されていないため、この処理は無効です。 | APIActivateRemoveInternet APIActivateRemoveTelephone APIProxyActivateRemovePrepare APIRestoreCancelStatus APIUpdateOfExpirationDate APIGetRegisteredInfoFromRegistry APIPreparationOfProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate |
| UnauthenticatedByTrialPeriod | 129 | 認証登録されていないため、この処理は無効です。 | APIActivateRemoveInternet APIActivateRemoveTelephone APIProxyActivateRemovePrepare APIRestoreCancelStatus |

| | | | |
|-----------------------------|-----|-------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | <p>APIUpdateOfExpirationDate APIGetRegisteredInfoFromRegistry APIPreparationOfProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate</p> |
| Unauthenticated | 130 | 認証登録されていないため、この処理は無効です。 | <p>APIActivateRemoveInternet APIActivateRemoveTelephone APIProxyActivateRemovePrepare APIRestoreCancelStatus APIUpdateOfExpirationDate APIGetRegisteredInfoFromRegistry APIPreparationOfProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate</p> |
| HardwareInformationMismatch | 131 | 認証済ハードウェア情報が不一致のため、この処理は無効です。 | <p>APIActivateRemoveInternet APIActivateRemoveTelephone APIProxyActivateRemovePrepare APIRestoreCancelStatus APIUpdateOfExpirationDate APIGetRegisteredInfoFromRegistry APIPreparationOfProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate</p> |
| ReleasekeyIsEmpty | 132 | 解除キーが不正です。 | <p>APIActivateRemoveTelephone</p> |
| ReleasekeyDigitsProblem | 133 | 解除キーが不正です。 | <p>APIActivateRemoveTelephone</p> |

| | | | |
|-------------------------------------|-----|---------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| BadReleasekey | 134 | 解除キーが不正です。 | APIActivateRemoveTelephone |
| DeauthorizeFailed | 135 | お客様の PC での認証解除が失敗しました。 再度、当処理を実行してください。 | APIActivateRemoveTelephone |
| ProxyAuthenticationDataFileNotFound | 136 | 代理認証データファイルが存在しません。 | APIProxyActivateRegisterExecute APIProxyActivateRegisterFix APIProxyActivateRemoveExecute APIProxyActivateRegisterExecute APIProxyActivateRemoveExecute APIGetProxyDataForExpirationDate APIGetProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate |
| NoDataInProxydatafile | 137 | 代理認証データファイルにデータが存在しません。 | APIProxyActivateRegisterExecute APIProxyActivateRegisterFix APIProxyActivateRemoveExecute |
| IncorrectProxyData | 138 | 代理認証データファイルの内容が不正です。 | APIProxyActivateRegisterExecute APIProxyActivateRegisterFix |
| FailedReadProxydata | 139 | 代理認証データの読み込みに失敗しました。 | APIProxyActivateRegisterExecute APIProxyActivateRemoveExecute APIProxyActivateRegisterFix |
| FailedToWriteProxyData | 140 | 弊社のデータベースには正常に登録されましたが、代理認証データの書き込みが失敗しました。 弊社のデータベースの登録を解除した上で、再度、当処理を実行していただく必要がありますので弊社までご連絡ください。 | APIProxyActivateRegisterExecute |
| FailedWithEncryption | 141 | 暗号化で失敗しました。 | APIProxyActivateRegisterExecute APIProxyActivateRegisterPrepare APIProxyActivateR |

| | | | |
|-------------------------------------|-----|---------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | emoveExecute APIProxyActivateR emovePrepare APIWriteProxyServ erInfoToRegistry APIPreparationOfP roxyUpdateOfExpir ationDate APIGetProxyUpdat eOfExpirationDate |
| ProxyDataInconsistency | 142 | 「代理認証データ」と現在お使いの PC の内容が異なるので確定処理が実行できません。 | APIProxyActivateR egisterFix APIProxyActivateR egisterExecute |
| 欠番 | 143 | | |
| AlreadyUsedProxyData | 144 | この代理認証データファイルは、過去に使用されているので登録に利用できません。 | APIProxyActivateR egisterFix |
| FailToRegisterConfirm | 145 | 登録確定に失敗しました。 | APIProxyActivateR egisterFix |
| FailToRegisterPreparation | 146 | 登録準備に失敗しました。 | APIProxyActivateR egisterPrepare |
| UnableToWriteProxyData | 147 | 弊社のデータベースには正常に解除 されましたが、代理認証データの書き 込みが失敗しました。 「認証解除/電話」でお客様の PC で の認証解除を実行していただく必要 がありますので弊社までご連絡くださ い。 | APIProxyActivateR emoveExecute |
| ThePathHasNotBeenSet | 148 | パスが設定されていません。 | APIProxyActivateR egisterPrepare APIProxyActivateR emovePrepare APIPreparationOfP roxyUpdateOfExpir ationDate |
| PathNotFound | 149 | 設定されたパスが存在しません。 | APIProxyActivateR egisterPrepare APIProxyActivateR emovePrepare APIPreparationOfP roxyUpdateOfExpir ationDate |
| EmptyAnyOfTheInputData | 150 | この PC は認証登録されていません。 | APIProxyActivateR emovePrepare |
| FailedToRemoveFromRegistry | 151 | お客様の PC での認証解除準備が失 敗しました。 再度、当処理を実行してください。 | APIProxyActivateR emovePrepare |
| FailedToDeauthenticationPreparation | 152 | 解除準備が失敗しました。 | APIProxyActivateR emovePrepare |

| | | | |
|------------------------------------|-----|----------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------|
| NotRestoreException | 153 | 何らかの原因で回復できませんでした。 | APIRestoreCancel Status APIRestoreRegisterStatus |
| PleasePerformTheDeauthorizeProcess | 154 | 弊社のデータベースにもデータが存在しますので当処理ではなく、「認証解除」処理を行ってください。 | APIRestoreCancel Status |
| FailedToDeauthorizeRecovery | 155 | お客様の PC で認証解除の回復に失敗しました。 「認証解除/電話」でお客様の PC の認証解除を実行していただく必要がありますので弊社までご連絡ください。 | APIRestoreCancel Status |
| 欠番 | 156 | | |
| 欠番 | 157 | | |
| FailedToRestoreActivation | 158 | お客様の PC で認証登録の回復に失敗しました。 弊社のデータベースの登録を解除した上で、再度、認証登録を実行していただく必要がありますので弊社までご連絡ください。 | APIRestoreRegisterStatus |
| InputDataNotFound | 159 | 入力された「プロダクト ID」や「シリアル No.」、PC 情報などが弊社のデータベースに登録されていません。 弊社のデータベースの登録を解除した上で、再度、認証登録を実行していただく必要がありますので弊社までご連絡ください。 | APIRestoreRegisterStatus |
| LicensingIssues | 160 | このプロダクト ID とシリアル No.の組み合わせはレンタルとして登録できません。 もう一度、プロダクト ID とシリアル No.をご確認いただいてお間違いない場合は弊社までご連絡ください。 | APIActivateRegisterInternet APIProxyActivateRegisterExecute APIRestoreRegisterStatus |
| RentalHasExpired | 161 | レンタルのご利用期限が過ぎました。 | APIActivateRegisterInternet APIProxyActivateRegisterExecute APIRestoreRegisterStatus |
| InvalidUseOfMACAddressOrCPU | 162 | MACアドレスか CPU 情報の使用が無効なためこの処理は実行できません。 弊社までご連絡ください。 | APIActivateRegisterInternet APIRestoreRegisterStatus |
| FailedToUpdateTheExpirationDate | 163 | 「有効期限」の更新に失敗しました。 | APIUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate |
| CanNotUseTheString | 164 | 使用できない文字列が含まれています。 | APIActivateRegister |

| | | | |
|-----------------------|-----|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | す。 | erInternet APIActivateRemoveInternet APIProxyActivateRegisterExecute APIProxyActivateRemoveExecute APIRestoreCancelStatus APIRestoreRegisterStatus APIUpdateOfExpirationDate APIGetProductIdSerialNoList APIGetFreeItem APIGetRegisteredInfoFromWeb APIGetProxyUpdateOfExpirationDate |
| PropertyValueNotFound | 165 | 「必須設定プロパティ」に値が設定されていません。 | APIActivateRegisterInternet APIGetRegisteredInfoFromRegistry APIActivateRegisterTelephone APIActivateRemoveInternet APIActivateRemoveTelephone APIActivateStatusCheck APIActivateStatusCheckOnline APIGenerationOfNewCertificationID APIGetProxyDataForRemove APIProxyActivateRegisterExecute APIProxyActivateRegisterFix APIProxyActivateRegisterPrepare APIProxyActivateRemoveExecute APIProxyActivateRemovePrepare APIRestoreCancelStatus |

| | | | |
|-----------------------------|-----|-----------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | <p>APIRestoreRegisterStatus APIUpdateOfExpirationDate APIGetProductIdSerialNoList APIGetFreeItem APITrialStartDateRemove APIRunNR2AppDateRemove APIRunNR2AppDateRemove2 APIActivateStatusCheck2 APIGetRegisteredInfoFromRegistry2</p> |
| NotUpdateTheExpirationDate | 166 | 新しい有効期限が同じか古い場合「更新しない」ように設定されているので更新されませんでした。 | <p>APIUpdateOfExpirationDate APIGetProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate</p> |
| CanNotReadTheTrialStartDate | 167 | 日付データの取得に失敗しました。(猶予) | <p>APIActivateRegisterInternet APIActivateRegisterTelephone APIActivateRemoveInternet APIActivateRemoveTelephone APIGenerationOfNewCertificationID APIGetRegisteredInfoFromRegistry APIProxyActivateRegisterFix APIProxyActivateRegisterPrepare APIProxyActivateRemovePrepare APIRestoreCancelStatus APIRestoreRegisterStatus APIUpdateOfExpirationDate APIPreparationOfProxyUpdateOfExpir</p> |

| | | | |
|----------------------------------|-----|----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | <p>ationDate APIDetermination OfProxyUpdateOfE xpirationDate</p> |
| CanNotReadTheRentalStartDay | 168 | 日付データの取得に失敗しました。 (レンタル) | <p>APIActivateRegisterInternet APIActivateRegisterTelephone APIActivateRemoveInternet APIActivateRemoveTelephone APIGenerationOfNewCertificationID APIGetProxyDataForRegister APIGetRegisteredInfoFromRegistry APIProxyActivateRegisterFix APIProxyActivateRegisterPrepare APIProxyActivateRegisterExecute APIProxyActivateRemovePrepare APIRestoreCancelStatus APIRestoreRegisterStatus APIUpdateOfExpirationDate APIPreparationOfProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate</p> |
| CanNotReadTheExpirationStartDate | 169 | 日付データの取得に失敗しました。 (有効期限) | <p>APIActivateRegisterInternet APIActivateRegisterTelephone APIActivateRemoveInternet APIActivateRemoveTelephone APIGenerationOfNewCertificationID APIGetProxyDataForRegister</p> |

| | | | |
|---------------------------------------|-----|---------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | <p>APIGetRegisteredInfoFromRegistry APIProxyActivateRegisterFix APIProxyActivateRegisterPrepare APIProxyActivateRegisterExecute APIProxyActivateRemovePrepare APIRestoreCancelStatus APIRestoreRegisterStatus APIUpdateOfExpirationDate APIPreparationOfProxyUpdateOfExpirationDate APIPreparationOfProxyUpdateOfExpirationDate APIGetProxyDataForExpirationDate APIGetProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate</p> |
| CanNotReadProxyServerInfoFromRegistry | 170 | プロキシサーバー情報の読み込みに失敗しました。 | APIReadProxyServerInfoFromRegistry |
| CanNotWriteProxyServerInfoToRegistry | 171 | プロキシサーバー情報の書き込みに失敗しました。 | APIWriteProxyServerInfoToRegistry |
| ExternalLinkKeyNotFound | 172 | 指定されたリンク用キーは弊社データベースに登録されていません。 | APIGetProductIdSerialNoList |
| FailToGetRegisterInfo | 173 | 認証情報の取得に失敗しました。 | APIPreparationOfProxyUpdateOfExpirationDate |
| PCDateChanged | 174 | PC の日付が変更されました。 | <p>APIActivateRegisterInternet APIActivateRegisterTelephone APIActivateRemoveInternet APIActivateRemoveTelephone APIGenerationOfN</p> |

| | | | |
|--------------------------------------|-----|-----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | <p>ewCertificationID APIGetRegisteredInfoFromRegistry APIProxyActivateRegisterFix APIProxyActivateRegisterPrepare APIProxyActivateRemovePrepare APIRestoreCancelStatus APIRestoreRegisterStatus APIUpdateOfExpirationDate APIPreparationOfProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate</p> |
| ProxyExpirationDateDataFileNotFound | 175 | 代理有効期限取得データファイルが存在しません。 | <p>APIGetProxyDataForExpirationDate APIGetProxyDataForExpirationDate APIGetProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate</p> |
| NoDataInProxyExpirationDateDataFile | 176 | 代理有効期限取得データファイルにデータが存在しません。 | <p>APIGetProxyDataForExpirationDate APIGetProxyDataForExpirationDate APIGetProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate</p> |
| FailedReadProxyExpirationDateData | 177 | 代理有効期限取得データの読み込みに失敗しました。 | <p>APIGetProxyDataForExpirationDate APIGetProxyDataForExpirationDate APIGetProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate</p> |
| FailedToWriteProxyExpirationDateData | 178 | 代理有効期限取得データの書き込みが失敗しました。 | <p>APIGetProxyUpdateOfExpirationDate</p> |

| | | | |
|-------------------------------------------------------|-----|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | もう一度、「代理有効期限取得準備 (認証 PC)」から行ってください。 | |
| NoExpirationDateSettingForProxyExpirationDateDataFile | 179 | 代理有効期限取得データは有効期限の利用設定はありません。 | APIDeterminationOfProxyUpdateOfExpirationDate |
| HardwareInfoAndProxyExpirationDateDataIsMismatch | 180 | 「代理有効期限取得データ」と現在お使いの PC の内容が一致しません。 | APIDeterminationOfProxyUpdateOfExpirationDate |
| FailedRemoveValueFromRegistry | 181 | 削除処理が失敗しました。(レジストリ) | APITrialStartDateRemove APIRunNR2AppDateRemove APIRunNR2AppDateRemove2 |
| FailedRemoveValueFromFolder | 182 | 削除処理が失敗しました。(フォルダ) | APIRunNR2AppDateRemove2 |
| FailedReadRunAppDateValueFromRegistry | 183 | 「アプリ起動日」の読み込みに失敗しました。(レジストリ) | APIActivateRegisterInternet APIActivateRegisterTelephone APIActivateRemoveInternet APIActivateRemoveTelephone APIActivateStatusCheck APIActivateStatusCheck2 APIGenerationOfNewCertificationID APIGetRegisteredInfoFromRegistry APIProxyActivateRegisterFix APIProxyActivateRegisterPrepare APIProxyActivateRemovePrepare APIRestoreCancelStatus APIRestoreRegisterStatus APIUpdateOfExpirationDate APIPreparationOfProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate |

| | | | |
|----------------------------------------|-----|------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | <p>APIActivateStatus CheckOnline APIGetRegisteredI nfoFromRegistry2</p> |
| FailedWriteRunAppDateValueFromRegistry | 184 | 「アプリ起動日」の書き込みに失敗しました。(レジストリ) | <p>APIActivateRegisterInternet APIActivateRegisterTelephone APIActivateRemoveInternet APIActivateRemoveTelephone APIActivateStatusCheck APIActivateStatusCheck2 APIGenerationOfNewCertificationID APIGetRegisteredI nfoFromRegistry APIProxyActivateRegisterFix APIProxyActivateRegisterPrepare APIProxyActivateRemovePrepare APIRestoreCancelStatus APIRestoreRegisterStatus APIUpdateOfExpirationDate APIPreparationOfProxyUpdateOfExpirationDate APIDeterminationOfProxyUpdateOfExpirationDate APIActivateStatusCheckOnline APIGetRegisteredI nfoFromRegistry2</p> |
| FailedReadRunAppDateValueFromFolder | 185 | 「アプリ起動日」の読み込みに失敗しました。(フォルダ) | <p>APIActivateRegisterInternet APIActivateRegisterTelephone APIActivateRemoveInternet APIActivateRemoveTelephone</p> |

| | | | |
|--------------------------------------|-----|-----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | <p>APIActivateStatus Check APIActivateStatus Check2 APIGenerationOfNew CertificationID APIGetRegisteredInfo FromRegistry APIProxyActivateRegister Fix APIProxyActivateRegister Prepare APIProxyActivateRemove Prepare APIRestoreCancel Status APIRestoreRegister Status APIUpdateOfExpiration Date APIPreparationOfProxy UpdateOfExpirationDate APIDeterminationOf ProxyUpdateOfExpiration Date APIActivateStatus CheckOnline APIGetRegisteredInfo FromRegistry2</p> |
| FailedWriteRunAppDateValueFromFolder | 186 | 「アプリ起動日」の書き込みに失敗しました。(フォルダ) | <p>APIActivateRegister Internet APIActivateRegister Telephone APIActivateRemove Internet APIActivateRemove Telephone APIActivateStatus Check APIActivateStatus Check2 APIGenerationOfNew CertificationID APIGetRegisteredInfo FromRegistry APIProxyActivateRegister Fix APIProxyActivateRegister Prepare</p> |

| | | | |
|--|------|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | APIProxyActivateR emovePrepare APIRestoreCancel Status APIRestoreRegiste rStatus APIUpdateOfExpira tionDate APIPreparationOfP roxyUpdateOfExpir ationDate APIDetermination OfProxyUpdateOfE xpirationDate APIActivateStatus CheckOnline APIGetRegisteredI nfoFromRegistry2 |
| | 9999 | その他エラー | |

APICertificationID プロパティ**【機能】**

認証 ID を取得します。(取得専用)

【構文】

<VB2010/VB6.0>

Public Property **APICertificationID** As **String**

<C#>

```
public string APICertificationID { get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation:: APICertificationID
```

【解説】

認証 ID を取得します。(取得専用)

次のメソッドを実行すると当プロパティ(APICertificationID プロパティ)に()内の各値が設定されます。

- [APIGenerationOfNewCertificationID](#) メソッド(新しく生成された認証 ID)
- [APIGetRegisteredInfoFromRegistry](#) メソッド(認証登録済みの認証 ID)
- [APIGetProxyDataForRegister](#) メソッド(代理認証登録データから取得した認証 ID)
- [APIGetProxyDataForRemove](#) メソッド(代理認証解除データから取得した認証 ID)

APICurrentExpirationDate プロパティ**【機能】**

現在の有効期限を取得します。(取得専用)

【構文】

<VB2010/VB6.0>

Public Property **APICurrentExpirationDate** As **String**

<C#>

```
public string APICurrentExpirationDate { get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APICurrentExpirationDate
```

【解説】

現在の有効期限を取得します。(取得専用)

次のメソッドの成功時のみ、当プロパティに値が設定されます。

・[APIGetRegisteredInfoFromRegistry](#) メソッド(レジストリからの証登録済み情報の取得)

当プロパティの値を利用するメソッドは次の通りです。

・[APIUpdateOfExpirationDate](#) メソッド(「有効期限の更新」を実行)

APIEncryptionPassword プロパティ**【機能】**

暗号化時のパスワードを設定します。

【構文】

<VB2010/VB6.0>

Public Property **APIEncryptionPassword** As **String**

<C#>

```
public string APIEncryptionPassword { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation:: APIEncryptionPassword
```

【解説】

認証 UI ライブラリ(DLL)はエンドユーザ PC のレジストリやファイルにデータを出力する際に必要に応じてデータを暗号化しています。その際の暗号化の方法は貴社では指定できませんが、暗号化する時の 2 つのパラメータ、パスワードと Salt 文字列は貴社で指定できます。

当プロパティには暗号化時のパスワードを指定します。全角でも指定できます。

(例)“認証レスキュー！”

当プロパティの文字数は、空文字列は不可で 1~65535 文字ですが、8 文字から 15 文字程度が妥当と思われます。

APIEncryptionSaltString プロパティ**【機能】**

暗号化時の Salt 文字列(8 文字以上)を設定します。

【構文】

<VB2010/VB6.0>

Public Property **APIEncryptionSaltString** As **String**

<C#>

```
public string APIEncryptionSaltString { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation:: APIEncryptionSaltString
```

【解説】

認証 UI ライブラリ(DLL)はエンドユーザ PC のレジストリやファイルにデータを出力する際に必要に応じてデータを暗号化しています。その際の暗号化の方法は貴社では指定できませんが、暗号化する時の 2 つのパラメータ、パスワードと Salt 文字列は貴社で指定できます。

当プロパティには暗号化時の Salt 文字列を指定します。

必ず 8 文字以上で指定します。

(例) "12345678ABCDEFGH"

APIErrorStatus プロパティ**【機能】**

API 系のメソッドを使用した際のエラーの内容を返します。

【構文】

<VB2010>

Public ReadOnly Property **APIErrorStatus** As **Integer**

<VB6.0>

Public ReadOnly Property **APIErrorStatus** As **Long**

<C#>

```
public int APIErrorStatus { get; }
```

<VC++>

```
public : long NewtonenRvcpp::IActivation:: APIErrorStatus
```

【解説】

API 系のメソッドは戻り値として正常 (True) かエラー (False) を返しますが、この **APIErrorStatus** プロパティはそのエラーの内容を [APIError 列挙体](#) の参照で返します。この **APIErrorStatus** プロパティは取得専用です。

APIExternalLinkKey プロパティ**【機能】**

APIGetProductIdSerialNoList メソッド 実行時の外部データベースとのリンク用キー項目を設定します。

【構文】

<VB2010/VB6.0>

Public Property **APIExternalLinkKey** As **String**

<C#>

```
public string APIExternalLinkKey { set; get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APIExternalLinkKey
```

【解説】

認証レスキュー！の ActivationKey テーブルよりプロダクト ID とシリアル No.のペア文字列の列挙を取得する APIGetProductIdSerialNoList メソッドの実行時に、外部データベースとのリンク用キー項目を当プロパティ(APIExternalLinkKey プロパティ)設定します。

認証レスキュー！とは別の外部データベースとのリンク用キーを当プロパティ(APIExternalLinkKey プロパティ)に設定し、APIGetProductIdSerialNoList メソッドを実行すると、認証レスキュー！のデータベースの ActivationKey テーブルより該当するプロダクト ID とシリアル No.のペア文字列の列挙をデリミタ付きで APIProductIdSerialNoList プロパティに設定して返します。

APINewExpirationDate プロパティ**【機能】**

新しい有効期限を取得します。(取得専用)

【構文】

<VB2010/VB6.0>

Public Property **APINewExpirationDate** As **String**

<C#>

```
public string APINewExpirationDate { get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation:: APINewExpirationDate
```

【解説】

新しい有効期限を取得します。(取得専用)

次のメソッドの成功時のみ、当プロパティに値が設定されます。

- ・[APIGetProxyDataForExpirationDate](#) メソッド(代理有効期限取得データの取得)
- ・[APIGetProxyUpdateOfExpirationDate](#) メソッド(「代理有効期限/取得」を実行(代理 PC で利用))

当プロパティの値を利用するメソッドは次の通りです。

- ・[APIDeterminationOfProxyUpdateOfExpirationDate](#) メソッド(「代理有効期限更新/確定」を実行)

APIFreeItem1～5 プロパティ**【機能】**

APIGetFreeItem メソッドを実行して ActivationKey テーブルより取得した自由入力項目の値が当プロパティ(APIFreeItem1～APIFreeItem5 プロパティ)に設定されます。(取得専用)

【構文】

<VB2010/VB6.0>

Public Property **APIFreeItem1** As **String**

Public Property **APIFreeItem2** As **String**

Public Property **APIFreeItem3** As **String**

Public Property **APIFreeItem4** As **String**

Public Property **APIFreeItem5** As **String**

<C#>

```
public string APIFreeItem1 { get; }
```

```
public string APIFreeItem2 { get; }
```

```
public string APIFreeItem3 { get; }
```

```
public string APIFreeItem4 { get; }
```

```
public string APIFreeItem5 { get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APIFreeItem1
```

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APIFreeItem2
```

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APIFreeItem3
```

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APIFreeItem4
```

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APIFreeItem5
```

【解説】

APIGetFreeItem メソッドを実行して ActivationKey テーブルより取得した自由入力項目の値が当プロパティ(APIFreeItem1～APIFreeItem5 プロパティ)に設定されます。

APILicenseKey プロパティ**【機能】**

ライセンスキーを設定・取得します。

【構文】

<VB2010/VB6.0>

Public Property **APILicenseKey** As **String**

<C#>

```
public string APILicenseKey { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation:: APILicenseKey
```

【解説】

ライセンスキーを設定・取得します。

次のメソッドを実行すると当プロパティ(APILicenseKey プロパティ)に()内の値が設定されます。

・[APIGetRegisteredInfoFromRegistry](#) メソッド(認証登録済みのライセンスキー)

APIOverwriteModeOfExpirationDateUpdate プロパティ**【機能】**

「有効期限の更新」(APIUpdateOfExpirationDate)メソッドを実行する際に、データベース上の新しい有効期限が現在のエンドユーザ PC のレジストリ内有効期限が同じか古い場合の対応を設定します。

【構文】

<VB2010/VB6.0>

Public Property **APIOverwriteModeOfExpirationDateUpdate** As **Boolean**

<C#>

```
public bool APIOverwriteModeOfExpirationDateUpdate { set; get; }
```

<VC++>

```
public : VARIANT_BOOL NewtonenRvcpp::IActivation::
```

```
APIOverwriteModeOfExpirationDateUpdate
```

【解説】

当プロパティには次のいずれか値を設定します。

True: 新しい有効期限が現在と同じか古い場合でも更新します。

False: 新しい有効期限が現在と同じか古い場合は更新しません。

「有効期限の更新」の実行については、[APIUpdateOfExpirationDate](#) メソッドをご覧ください。

APIProductID プロパティ**【機能】**

プロダクト ID を設定・取得します。

【構文】

<VB2010/VB6.0>

Public Property **APIProductID** As **String**

<C#>

```
public string APIProductID { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation:: APIProductID
```

【解説】

プロダクト ID を設定・取得します。

次のメソッドを実行すると当プロパティ(APIProductID プロパティ)に()内の各値が設定されます。

- [APIGetRegisteredInfoFromRegistry](#) メソッド(認証登録済みのプロダクト ID)
- [APIGetProxyDataForRegister](#) メソッド(代理認証登録データから取得したプロダクト ID)
- [APIGetProxyDataForRemove](#) メソッド(代理認証解除データから取得したプロダクト ID)

| |
|--------------------------------|
| APIProductIdSerialNoList プロパティ |
|--------------------------------|

【機能】

APIGetProductIdSerialNoList メソッド実行後にプロダクトIDとシリアルNo.のペアをデリミタで列挙したの文字列が設定されます。(取得専用)

【構文】

<VB2010/VB6.0>

Public Property **APIProductIdSerialNoList** As **String**

<C#>

```
public string APIProductIdSerialNoList { get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APIProductIdSerialNoList
```

【解説】

APIGetProductIdSerialNoList メソッド実行後にプロダクトIDとシリアルNo.のペアをデリミタで列挙した文字列が当プロパティ(APIProductIdSerialNoList プロパティ)設定されます。

認証レスキュー！とは別の外部データベースとのリンク用キーを APIExternalLinkKey プロパティに設定し、APIGetProductIdSerialNoList メソッドを実行すると、認証レスキュー！のデータベースの ActivationKey テーブルより該当するプロダクトIDとシリアルNo.のペア文字列の列挙をデリミタ付きで当プロパティ(APIProductIdSerialNoList プロパティ)に設定して返します。

プロダクトIDとシリアルNo.のペア文字列の列挙は、たとえば次のような文字列です。
デリミタはカンマ(,)です。

```
1111-1111-1111,aaaa1111,22222-22222-22222,bbbb2222,33333-33333-33333,cccc3333
```

この例では次の3組のプロダクトIDとシリアルNo.が返されました。

プロダクトID ="11111-11111-11111" シリアルNo.="aaaa1111"

プロダクトID ="22222-22222-22222" シリアルNo.="bbbb2222"

プロダクトID ="33333-33333-33333" シリアルNo.="cccc3333"

APIProxyDataPath プロパティ**【機能】**

代理認証データパスを設定・取得します。

(代理認証機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Property **APIProxyDataPath** As **String**

<C#>

```
public string APIProxyDataPath { set; get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APIProxyDataPath
```

【解説】

代理認証データパスを設定・取得します。ファイルの拡張子は「.NRS」とします。
当プロパティを利用するメソッドは次の通りです。

- [APIGetProxyDataForRegister](#) メソッド(代理認証登録データの取得)
- [APIGetProxyDataForRemove](#) メソッド(代理認証解除データの取得)
- [APIProxyActivateRegisterExecute](#) メソッド(代理認証登録/実行)
- [APIProxyActivateRegisterFix](#) メソッド(代理認証登録/確定)
- [APIProxyActivateRegisterPrepare](#) メソッド(代理認証登録/準備)
- [APIProxyActivateRemoveExecute](#) メソッド(代理認証解除/実行)
- [APIProxyActivateRemovePrepare](#) メソッド(代理認証解除/準備)

APIProxyServerAddress プロパティ

【機能】

プロキシサーバーのアドレスを設定します。

【構文】

<VB2010/VB6.0>

Public Property **APIProxyServerAddress** As **String**

<C#>

```
public string APIProxyServerAddress { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation:: APIProxyServerAddress
```

【解説】

プロキシサーバーのアドレスを設定します。

インターネットに接続する際にプロキシサーバーを使用する場合のみ設定が必要です。

プロキシサーバーを使用する場合は、当プロパティを含む次のプロパティに適切な設定が必要です。

- ・[APIProxyServerAddress](#) プロパティ(プロキシサーバーのアドレス)→当プロパティ
- ・[APIProxyServerPort](#) プロパティ(プロキシサーバーのポート)
- ・[APIProxyServerUserName](#) プロパティ(プロキシサーバーのユーザ名)
- ・[APIProxyServerPassword](#) プロパティ(プロキシサーバーのパスワード)

プロキシサーバーを使用する場合、当プロパティの値を利用するメソッドは次の通りです。

- ・[APIActivateRegisterInternet](#) メソッド(インターネットによる認証登録を実行)
- ・[APIActivateRemoveInternet](#) メソッド(インターネットによる認証解除を実行)
- ・[APIProxyActivateRegisterExecute](#) メソッド(代理認証登録/実行)
- ・[APIProxyActivateRemoveExecute](#) メソッド(代理認証解除/実行)
- ・[APIRestoreRegisterStatus](#) メソッド(認証登録状態回復)
- ・[APIRestoreCancelStatus](#) メソッド(認証解除状態回復)
- ・[APIUpdateOfExpirationDate](#) メソッド(有効期限の更新)

APIProxyServerPassword プロパティ

【機能】

プロキシサーバーのパスワードを設定します。

【構文】

<VB2010/VB6.0>

Public Property **APIProxyServerPassword** As **String**

<C#>

```
public string APIProxyServerPassword { set; get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APIProxyServerPassword
```

【解説】

プロキシサーバーのパスワードを設定します。

インターネットに接続する際にプロキシサーバーを使用する場合のみ設定が必要です。

プロキシサーバーを使用する場合は、当プロパティを含む次のプロパティに適切な設定が必要です。

- ・[APIProxyServerAddress](#) プロパティ(プロキシサーバーのアドレス)
- ・[APIProxyServerPort](#) プロパティ(プロキシサーバーのポート)
- ・[APIProxyServerUserName](#) プロパティ(プロキシサーバーのユーザ名)
- ・[APIProxyServerPassword](#) プロパティ(プロキシサーバーのパスワード)→**当プロパティ**

プロキシサーバーを使用する場合、当プロパティの値を利用するメソッドは次の通りです。

- ・[APIActivateRegisterInternet](#) メソッド(インターネットによる認証登録を実行)
- ・[APIActivateRemoveInternet](#) メソッド(インターネットによる認証解除を実行)
- ・[APIProxyActivateRegisterExecute](#) メソッド(代理認証登録/実行)
- ・[APIProxyActivateRemoveExecute](#) メソッド(代理認証解除/実行)
- ・[APIRestoreRegisterStatus](#) メソッド(認証登録状態回復)
- ・[APIRestoreCancelStatus](#) メソッド(認証解除状態回復)
- ・[APIUpdateOfExpirationDate](#) メソッド(有効期限の更新)

APIProxyServerPort プロパティ

【機能】

プロキシサーバーのポートを設定します。

【構文】

<VB2010/VB6.0>

Public Property APIProxyServerPort As **String**

<C#>

```
public string APIProxyServerPort { set; get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APIProxyServerPort
```

【解説】

プロキシサーバーのポートを設定します。

インターネットに接続する際にプロキシサーバーを使用する場合のみ設定が必要です。

プロキシサーバーを使用する場合は、当プロパティを含む次のプロパティに適切な設定が必要です。

- ・[APIProxyServerAddress](#) プロパティ(プロキシサーバーのアドレス)
- ・[APIProxyServerPort](#) プロパティ(プロキシサーバーのポート)→**当プロパティ**
- ・[APIProxyServerUserName](#) プロパティ(プロキシサーバーのユーザ名)
- ・[APIProxyServerPassword](#) プロパティ(プロキシサーバーのパスワード)

プロキシサーバーを使用する場合、当プロパティの値を利用するメソッドは次の通りです。

- ・[APIActivateRegisterInternet](#) メソッド(インターネットによる認証登録を実行)
- ・[APIActivateRemoveInternet](#) メソッド(インターネットによる認証解除を実行)
- ・[APIProxyActivateRegisterExecute](#) メソッド(代理認証登録/実行)
- ・[APIProxyActivateRemoveExecute](#) メソッド(代理認証解除/実行)
- ・[APIRestoreRegisterStatus](#) メソッド(認証登録状態回復)
- ・[APIRestoreCancelStatus](#) メソッド(認証解除状態回復)
- ・[APIUpdateOfExpirationDate](#) メソッド(有効期限の更新)

APIProxyServerUserName プロパティ

【機能】

プロキシサーバーのユーザ名を設定します。

【構文】

<VB2010/VB6.0>

Public Property APIProxyServerUserName As **String**

<C#>

```
public string APIProxyServerUserName { set; get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APIProxyServerUserName
```

【解説】

プロキシサーバーのユーザ名を設定します。

インターネットに接続する際にプロキシサーバーを使用する場合のみ設定が必要です。

プロキシサーバーを使用する場合は、当プロパティを含む次のプロパティに適切な設定が必要です。

- ・[APIProxyServerAddress](#) プロパティ(プロキシサーバーのアドレス)
- ・[APIProxyServerPort](#) プロパティ(プロキシサーバーのポート)
- ・[APIProxyServerUserName](#) プロパティ(プロキシサーバーのユーザ名)→**当プロパティ**
- ・[APIProxyServerPassword](#) プロパティ(プロキシサーバーのパスワード)

プロキシサーバーを使用する場合、当プロパティの値を利用するメソッドは次の通りです。

- ・[APIActivateRegisterInternet](#) メソッド(インターネットによる認証登録を実行)
- ・[APIActivateRemoveInternet](#) メソッド(インターネットによる認証解除を実行)
- ・[APIProxyActivateRegisterExecute](#) メソッド(代理認証登録/実行)
- ・[APIProxyActivateRemoveExecute](#) メソッド(代理認証解除/実行)
- ・[APIRestoreRegisterStatus](#) メソッド(認証登録状態回復)
- ・[APIRestoreCancelStatus](#) メソッド(認証解除状態回復)
- ・[APIUpdateOfExpirationDate](#) メソッド(有効期限の更新)

APIReleaseKey プロパティ**【機能】**

解除キーを設定・取得します。

【構文】

<VB2010/VB6.0>

Public Property **APIReleaseKey** As **String**

<C#>

```
public string APIReleaseKey { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation:: APIReleaseKey
```

【解説】

解除キーを設定・取得します。

当プロパティは、電話による認証解除メソッド ([APIActivateRemoveTelephone](#) メソッド) で使用します。

解除キーは通常、たとえば“956-11749-60711”といったハイフン付きの数字列です。

APIReleaseStatus プロパティ**【機能】**

解除ステータスを取得します。(取得専用)

【構文】

<VB2010/VB6.0>

Public Property **APIReleaseStatus** As **String**

<C#>

```
public string APIReleaseStatus { get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation:: APIReleaseStatus
```

【解説】

解除ステータスを取得します。(取得専用)

当プロパティは、電話による認証解除メソッド([APIActivateRemoveTelephone](#) メソッド)の実行が成功した場合のみ自動的に設定されます。

解除ステータスは通常、たとえば"162-20797-49245"といったハイフン付きの数字列です。

APIRentalPeriod プロパティ**【機能】**

レンタル日数(デフォルト:0日、設定可能範囲:1~1100)を設定します。

【構文】

<VB2010>

Public Property **APIRentalPeriod** As **Integer**

<VB6.0>

Public Property **APIRentalPeriod** As **Long**

<C#>

```
public int APIRentalPeriod { set; get; }
```

<VC++>

```
public : long NewtonenRvcpp::IActivation:: APIRentalPeriod
```

【解説】

エンドユーザが初めて認証してからアプリケーションが動作する期間を指定します。

1(日)~1100(日)の間で設定できます。

当プロパティを0に設定すると、レンタル期間は無いことになります。

※お客様のパッケージ製品にマルチライセンス(例:10ライセンス)が含まれる場合はレンタル機能は使用できません。レンタル機能を使用するにはシングルライセンス(1ライセンス)の製品である必要があります。また、レンタル機能を使用する場合は「電話で認証登録」機能は利用できません。

APISelectRunAppDatePathFlag プロパティ**【機能】**

「アプリ起動日」の読み込み/書き込み場所の切り替えフラグを設定します。

[APIActivateStatusCheck2](#) メソッドと [APIActivateStatusCheckOnline2](#) メソッドで使用されます。

【構文】

<VB2010>

Public Property **APISelectRunAppDatePathFlag** As **Integer**

<VB6.0>

Public Property **APISelectRunAppDatePathFlag** As **Long**

<C#>

```
public int APISelectRunAppDatePathFlag { set; get; }
```

<VC++>

```
public : long NewtonNRvcpp::IActivation:: APISelectRunAppDatePathFlag
```

【解説】

このプロパティ値により「アプリ起動日」の読み込み/書き込み場所を切り替えます。

0 を指定した場合 : HKEY_CURRENT_USER (レジストリ)

1 を指定した場合 : ProgramData (フォルダ)

[APIActivateStatusCheck2](#) メソッドと [APIActivateStatusCheckOnline2](#) メソッドで使用されます。

APISerialNo プロパティ**【機能】**

シリアル No.を設定・取得します。

【構文】

<VB2010/VB6.0>

Public Property **APISerialNo** As **String**

<C#>

```
public string APISerialNo { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation:: APISerialNo
```

【解説】

シリアル No.を設定・取得します。

次のメソッドを実行すると当プロパティ(APIProductID プロパティ)に()内の各値が設定されます。

- ・[APIGetRegisteredInfoFromRegistry](#) メソッド(認証登録済みのシリアル No.)
- ・[APIGetProxyDataForRegister](#) メソッド(代理認証登録データから取得したシリアル No.)
- ・[APIGetProxyDataForRemove](#) メソッド(代理認証解除データから取得したシリアル No.)

APITrialPeriod プロパティ**【機能】**

猶予(試用)日数(デフォルト:0日、設定可能範囲:1~365)を設定します。

【構文】

<VB2010>

Public Property **APITrialPeriod** As **Integer**

<VB6.0>

Public Property **APITrialPeriod** As **Long**

<C#>

```
public int APITrialPeriod { set; get; }
```

<VC++>

```
public : long NewtonenRvcpp::IActivation:: APITrialPeriod
```

【解説】

エンドユーザがライセンス認証登録をしなくてもアプリケーションが動作する期間を指定します。

1(日)~365(日)の間で設定できます。

当プロパティを0に設定すると、猶予期間は無いことになり、ライセンス認証登録をしない限りアプリケーション(またはアプリケーションの主機能など)が動作しないようにできます。

APIUseCpuInfo プロパティ**【機能】**

CPU 情報の使用 (デフォルト: True) を設定します。

次の 3 つのメソッドの処理において、認証情報とエンドユーザ PC 内の CPU 情報とを照合するかどうかを設定します。

- ・[APIActivateStatusCheck](#) メソッド (エンドユーザ PC 内での認証状態の確認)
- ・[APIActivateStatusCheckOnline](#) メソッド (オンラインで認証状態の確認)
- ・[APIRestoreRegisterStatus](#) メソッド (「認証登録状態回復」処理の実行)

【構文】

<VB2010/VB6.0>

Public Property **APIUseCpuInfo** As **Boolean**

<C#>

```
public bool APIUseCpuInfo { set; get; }
```

<VC++>

```
public : VARIANT_BOOL NewtonenNRvcpp::IActivation:: APIUseCpuInfo
```

【解説】

認証レスキュー！の認証 UI ライブラリでは、認証登録など各処理時にエンドユーザ PC の CPU 情報を記録します。当プロパティを True にするとエンドユーザ PC の CPU 情報を認証識別情報として利用します。これを利用することでエンドユーザ PC の不正利用時の認証識別情報の精度を上げます。

※[APIRestoreRegisterStatus](#) メソッド (「認証登録状態回復」処理の実行) を使用する際は、当プロパティを必ず True に設定してください。

APIUseMacAddress プロパティ

【機能】

MAC アドレスの使用(デフォルト: True)を設定します。

次の 3 つのメソッドの処理において、認証情報とエンドユーザ PC 内の MAC アドレスとを照合するかどうかを設定します。

- ・[APIActivateStatusCheck](#) メソッド(エンドユーザ PC 内での認証状態の確認)
- ・[APIActivateStatusCheckOnline](#) メソッド(オンラインで認証状態の確認)
- ・[APIRestoreRegisterStatus](#) メソッド(「認証登録状態回復」処理の実行)

【構文】

<VB2010/VB6.0>

Public Property **APIUseMacAddress** As **Boolean**

<C#>

```
public bool APIUseMacAddress { set; get; }
```

<VC++>

```
public : VARIANT_BOOL NewtonenRvcpp::IActivation:: APIUseMacAddress
```

【解説】

認証レスキュー！の認証 UI ライブラリでは、認証登録など各処理時にエンドユーザ PC の MAC アドレスを最大で 5 個記録します。MAC アドレスは LAN ポートなどのネットワーク機器に固有な物理アドレスです。MAC アドレスは、たとえば“E840F260C430”といった文字列です

当プロパティを True にするとエンドユーザ PC の MAC アドレスを認証識別情報として利用します。これを利用することでエンドユーザ PC の不正利用時の認証識別情報の精度を上げます。

※[APIRestoreRegisterStatus](#)メソッド(「認証登録状態回復」処理の実行)を使用する際は、当プロパティを必ず True に設定してください。

APIUseProxyServer プロパティ

【機能】

プロキシサーバーの使用区分(デフォルト:False)を設定します。

【構文】

<VB2010/VB6.0>

Public Property **APIUseProxyServer** As **Boolean**

<C#>

```
public bool APIUseProxyServer { set; get; }
```

<VC++>

```
public : VARIANT_BOOL NewtonenRvcpp::IActivation:: APIUseProxyServer
```

【解説】

プロキシサーバーの使用区分(デフォルト:False)を設定します。

インターネットに接続する際にプロキシサーバーを使用するかどうかを設定します。

プロキシサーバーを使用する場合は当プロパティに True を、使用しない場合は False をそれぞれ設定します。

プロキシサーバーを使用する場合は、次のプロパティに適切な設定が必要です。

- ・[APIProxyServerAddress](#) プロパティ(プロキシサーバーのアドレス)
- ・[APIProxyServerPort](#) プロパティ(プロキシサーバーのポート)
- ・[APIProxyServerUserName](#) プロパティ(プロキシサーバーのユーザ名)
- ・[APIProxyServerPassword](#) プロパティ(プロキシサーバーのパスワード)

プロキシサーバーを使用する場合、当プロパティの値を利用するメソッドは次の通りです。

- ・[APIActivateRegisterInternet](#) メソッド(インターネットによる認証登録を実行)
- ・[APIActivateRemoveInternet](#) メソッド(インターネットによる認証解除を実行)
- ・[APIProxyActivateRegisterExecute](#) メソッド(代理認証登録/実行)
- ・[APIProxyActivateRemoveExecute](#) メソッド(代理認証解除/実行)
- ・[APIRestoreRegisterStatus](#) メソッド(認証登録状態回復)
- ・[APIRestoreCancelStatus](#) メソッド(認証解除状態回復)
- ・[APIUpdateOfExpirationDate](#) メソッド(有効期限の更新)

| |
|---------------------------------------------|
| APIVendorsProductStartRegistryKeyPath プロパティ |
|---------------------------------------------|

【機能】

ベンダアプリケーション開始レジストリキーパスを設定します。

【構文】

<VB2010/VB6.0>

Public Property **APIVendorsProductStartRegistryKeyPath** As **String**

<C#>

```
public string APIVendorsProductStartRegistryKeyPath { set; get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation::
```

APIVendorsProductStartRegistryKeyPath

【解説】

エンドユーザに配布したアプリケーションが認証 UI ライブラリ(DLL)の機能を使う場合、その各種情報の記録先として使うエンドユーザ PC 上のレジストリの開始キーのパスを当プロパティに指定します。レジストリ内の「HKEY_LOCAL_MACHINE」以降を指定します。

(例)“Software¥Newtone¥NinshoRescue¥NR-200¥SampleProject”

なお、物理的なレジストリの位置は動作する貴社のアプリケーションが32bitなのか64bitなのかと、動作するPCのOSが32bitなのか64bitなのかによって異なります。

たとえば、HKEY_LOCAL_MACHINE¥SOFTWARE¥ABCというレジストリパスは次のようになります。32bitOS上で32bitアプリケーションが動作する場合、または64bitOS上で64bitアプリケーションが動作する場合は、

HKEY_LOCAL_MACHINE¥SOFTWARE¥ABC

そのままです。

しかし、64bitOS上で32bitアプリケーションが動作する場合は、

HKEY_LOCAL_MACHINE¥SOFTWARE¥Wow6432Node¥ABC

となります。

また、貴社の異なる複数のアプリケーションをエンドユーザの同一PC上で使用させるには、当プロパティにアプリケーションごとのそれぞれ異なるレジストリパスを設定してください。たとえば、次のように設定します。

アプリケーションA内での設定コード例:

```
APIVendorsProductStartRegistryKeyPath = "Software¥Company¥A"
```

アプリケーションB内での設定コード例:

```
APIVendorsProductStartRegistryKeyPath = "Software¥Company¥B"
```

APIWebServiceBasicAuthenticationPassword プロパティ**【機能】**

Web サービス時の基本認証パスワードを設定します。

【構文】

<VB2010/VB6.0>

Public Property **APIWebServiceBasicAuthenticationPassword** As **String**

<C#>

```
public string APIWebServiceBasicAuthenticationPassword { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation::
```

```
APIWebServiceBasicAuthenticationPassword
```

【解説】

Web サーバーで基本認証を使用する場合に、そのパスワードを指定します。

基本認証に関しては、[APIWebServiceUseBasicAuthentication](#) プロパティを参照してください。

APIWebServiceBasicAuthenticationUserName プロパティ**【機能】**

Web サービス時の基本認証ユーザ名を設定します。

【構文】

<VB2010/VB6.0>

Public Property **APIWebServiceBasicAuthenticationUserName** As **String**

<C#>

```
public string APIWebServiceBasicAuthenticationUserName { set; get; }
```

<VC++>

```
public : _bstr_t NewtonenRvcpp::IActivation::
```

```
APIWebServiceBasicAuthenticationUserName
```

【解説】

Web サーバーで基本認証を使用する場合に、そのユーザ名を指定します。

基本認証に関しては、[APIWebServiceUseBasicAuthentication](#) プロパティを参照してください。

APIWebServiceCheckPassword プロパティ**【機能】**

Web サービス確認パスワード(8 文字以上)を設定します。

【構文】

<VB2010/VB6.0>

Public Property **APIWebServiceCheckPassword** As **String**

<C#>

```
public string APIWebServiceCheckPassword { set; get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APIWebServiceCheckPassword
```

【解説】

Web サービスを利用する場合の確認用のパスワードを設定します。ここでは、Web サーバー側設定アプリケーション「認証 Web サービス」の環境設定処理の Web サービスの確認パスワードと同じものを設定します。

確認パスワードは必須項目です、省略はできません。8 文字以上で半角の次の文字が使用できません。

- ・大文字の英字(A～Z)
- ・小文字の英字(a～z)
- ・数字(0～9)
- ・記号(+-*!/#\$%&()=¥@<>?)

APIWebServiceTimeout プロパティ**【機能】**

Web サービスのタイムアウト(デフォルト: 60 秒)を設定します。

【構文】

<VB2010>

Public Property **APIWebServiceTimeout** As **Integer**

<VB6.0>

Public Property **APIWebServiceTimeout** As **Long**

<C#>

```
public int APIWebServiceTimeout { set; get; }
```

<VC++>

```
public : long NewtonenRvcpp::IActivation:: APIWebServiceTimeout
```

【解説】

Web サービスに接続して応答を待つ最大時間を秒単位で指定します。初期値は 60 秒です。

APIWebServiceURL プロパティ**【機能】**

Web サービスの URL を設定します。

【構文】

<VB2010/VB6.0>

Public Property **APIWebServiceURL** As **String**

<C#>

```
public string APIWebServiceURL { set; get; }
```

<VC++>

```
public : _bstr_t NewtoneNRvcpp::IActivation:: APIWebServiceURL
```

【解説】

認証に関するシステムを Web サービスとして提供する Web サーバーの URL を指定します。
以下に例を示します。

自 PC のローカルホストの Web サーバー(IIS)にアクセスする例:

```
"http://localhost/Nr2WebService/Service.asmx"
```

(注)この例は実際の運用ではありえません。貴社のアプリケーションで認証 UI ライブラリ(DLL)を使用した開発時に、テスト用に使用される URL です。

自社 Web サーバー(IIS)にアクセスさせる例:

```
"http://www.newtone.co.jp/Nr2WebService/Service.asmx"
```

クラウドサービス Microsoft Azure の Web アプリ (旧 Web サイト)に配置した Web サービスを利用する例:

```
"http://newtonecojp.azurewebsites.net/Service.asmx"
```

| |
|-------------------------------------------|
| APIWebServiceUseBasicAuthentication プロパティ |
|-------------------------------------------|

【機能】

Web サービス時の基本認証の使用(デフォルト:False)を設定します。

【構文】

<VB2010/VB6.0>

Public Property **APIWebServiceUseBasicAuthentication** As **Boolean**

<C#>

```
public bool APIWebServiceUseBasicAuthentication { set; get; }
```

<VC++>

```
public : VARIANT_BOOL NewtonenRvcpp::IActivation::
```

APIWebServiceUseBasicAuthentication

【解説】

Web サーバーで基本認証を使用する場合は、当プロパティを True にします。
初期値は、False(基本認証を使用しない)です。

当プロパティは、Web サーバー(IIS)側で特定のアカウント(ユーザ名とパスワード)でアクセスできるフォルダにこの Web サービスが配置してある場合に使用できるセキュリティ設定です。

Web サーバーで基本認証を使用する一般的な手順は次の通りです。

1.サーバーPC上でユーザを作成。

この際のユーザ名とパスワードがそのまま基本認証に使われます。

2.基本認証フォルダのセキュリティ設定

フォルダのプロパティを開き、セキュリティタブで上記 1 のユーザ名を 追加し、「読み取り」権限を付与します。

3.IISでのセキュリティ設定

IIS で該当フォルダに対し「認証」の設定で「匿名認証」を無効にして 「基本認証」を有効にします。

なお、Web サーバーでの基本認証の詳細につきましてはマイクロソフト社の関連ドキュメントをご覧ください。

<API 系メソッド>

メソッド一覧

| メソッド | 機能 |
|---------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| APIActivateRegisterInternet | インターネットによる認証登録の実行 |
| APIActivateRegisterTelephone | 電話による認証登録の実行 |
| APIActivateRemoveInternet | インターネットによる認証解除の実行 |
| APIActivateRemoveTelephone | 電話による認証解除の実行 |
| APIActivateStatusCheck | エンドユーザ PC 内での認証状態の確認 |
| APIActivateStatusCheck2 | エンドユーザ PC 内での認証状態の確認 (Windows の管理者でなくても実行可能) |
| APIActivateStatusCheckOnline | オンラインで認証状態の確認 |
| APIActivateStatusCheckOnline2 | オンラインで認証状態の確認 (Windows の管理者でなくても実行可能) |
| APIDeterminationOfProxyUpdateOfExpirationDate | 「代理有効期限更新/確定」処理の実行 |
| APIGenerationOfNewCertificationID | 新規認証 ID の生成 |
| APIGetFreeItem | プロダクト ID とシリアル No.を指定して、ActivationKey テーブルより自由入力項目を取得 |
| APIGetProductIdSerialNoList | 外部データベースとのリンク用キー項目を指定し ActivationKey テーブルよりプロダクト ID とシリアル No. のペア文字列の列挙を取得 |
| APIGetProxyDataForExpirationDate | 代理有効期限取得データの取得 |
| APIGetProxyDataForRegister | 代理認証登録データの取得 |
| APIGetProxyDataForRemove | 代理認証解除データの取得 |
| APIGetProxyUpdateOfExpirationDate | 「代理有効期限/取得」処理の実行 (代理 PC で利用) |
| APIGetRegisteredInfoFromRegistry | (レジストリからの) 認証登録済み情報の取得 |
| APIGetRegisteredInfoFromRegistry2 | (レジストリからの) 認証登録済み情報の取得 (Windows の管理者でなくても実行可能) |
| APIGetRegisteredInfoFromWeb | インターネットを使用して ActivationKeyTable から「ライセンス数」「有効期限の利用」「有効期限」を、ActivationDataTable から「認証 ID」「認証日時(作成日)」を取得します。 |
| APIPreparationOfProxyUpdateOfExpirationDate | 「代理有効期限取得/準備」処理の実行 |
| APIProxyActivateRegisterExecute | 「代理認証登録/実行」処理の実行 (代理 PC で利用) |
| APIProxyActivateRegisterFix | 「代理認証登録/確定」処理の実行 |
| APIProxyActivateRegisterPrepare | 「代理認証登録/準備」処理の実行 |
| APIProxyActivateRemoveExecute | 「代理認証解除/実行」処理の実行 (代理 PC で利用) |
| APIProxyActivateRemovePrepare | 「代理認証解除/準備」処理の実行 |
| APIReadProxyServerInfoFromRegistry | レジストリからプロキシサーバーの情報を取得 |
| APIRestoreCancelStatus | 「認証解除状態回復」処理の実行 |
| APIRestoreRegisterStatus | 「認証登録状態回復」処理の実行 |
| APIRunNR2AppDateRemove | 「アプリ起動日」を削除 |
| APIRunNR2AppDateRemove2 | 「アプリ起動日」を削除 |
| APITrialStartDateRemove | 「猶予日数」の「開始日」を削除 |
| APIUpdateOfExpirationDate | 有効期限更新の実行 |
| APIWriteProxyServerInfoToRegistry | レジストリへプロキシサーバーの情報を書き込み |

APIActivateRegisterInternet メソッド

【機能】

インターネットによる認証登録を実行します。

【構文】

<VB2010/VB6.0>

Public Function **APIActivateRegisterInternet**() As **Boolean**

<C#>

public **bool** **APIActivateRegisterInternet**()

<VC++>

public : **VARIANT_BOOL** **NewtonenRvcpp::IActivation:: APIActivateRegisterInternet**()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分※ | プロパティ | 設定する内容 | |
|-------|-----------------------------------------|------------------------------------------------------------------------------------------------------------|-----------------|
| UI | APICertificationID | (認証登録する)認証 ID(取得専用)。新規認証 ID 生成メソッド (APIGenerationOfNewCertificationID メソッド)の実行により、新しい認証 ID が当プロパティに設定されます。 | |
| UI | APIProductID | (認証登録する)プロダクト ID | |
| UI | APISerialNo | (認証登録する)シリアル No. | |
| UI | APIUseProxyServer | プロキシサーバーの使用区分 | |
| UI | APIProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー 使用時 |
| UI | APIProxyServerPort | プロキシサーバーのポート | |
| UI | APIProxyServerUserName | プロキシサーバーのユーザー名 | |
| UI | APIProxyServerPassword | プロキシサーバーのパスワード | |
| Code | APIEncryptionPassword | 暗号化時のパスワード | |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 | |
| Code | APIRentalPeriod | レンタル日数(0:レンタル期間なし) | |
| Code | APITrialPeriod | 猶予(試用)日数(0:猶予期間なし) | |
| Code | APIUseCpuInfo | CPU 情報の使用区分 | |

| | | |
|------|----------------------------------------------------------|-------------------------------|
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード (基本認証使用時) |
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名 (基本認証使用時) |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIWebServiceTimeout | Web サービスのタイムアウト |
| Code | APIWebServiceURL | Web サービスの URL |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 |

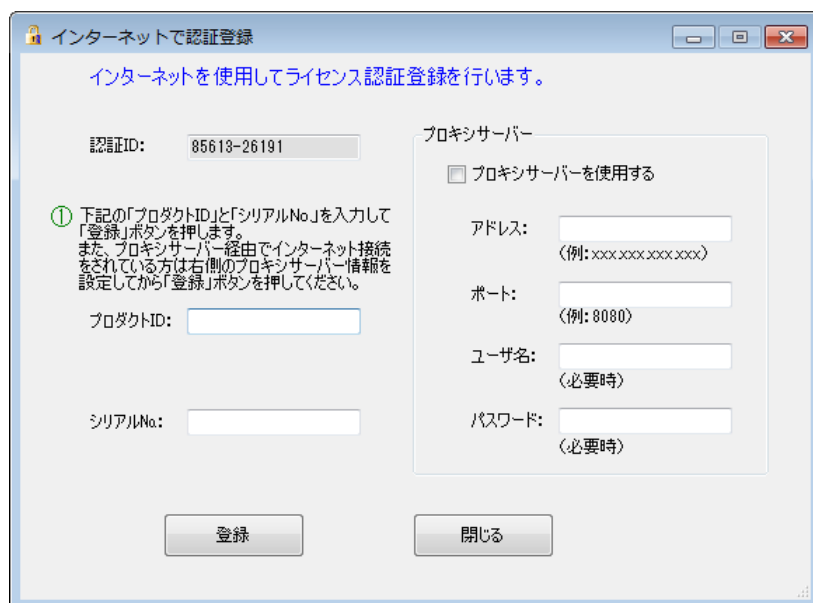
※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

インターネットによる認証登録を実行します。

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。



この画面上の「認証 ID」は、新規認証 ID 生成メソッド (APIGenerationOfNewCertificationID メソッド) の実行で設定される認証 ID プロパティ (APICertificationID プロパティ) で取得できます。

◆処理手順

一連の処理の手順は次の通りです。

[1] 新規認証 ID 生成メソッド (APIGenerationOfNewCertificationID メソッド) を実行します。
 これにより、認証 ID プロパティ (APICertificationID プロパティ) に新しい認証 ID が設定されます。
 その内容を画面に表示します。この内容表示そのものは必須ではありません。

[2] プロダクト ID とシリアル No. をエンドユーザに画面より入力させ、プロダクト ID プロパティ (APIProductID プロパティ) とシリアル No. プロパティ (APISerialNo プロパティ) にそれぞれ設定します。あるいは、別の方法で結果としてプロダクト ID プロパティ (APIProductID プロパティ) とシリアル No. プロパティ (APISerialNo プロパティ) にそれぞれ認証登録するプロダクト ID とシリアル No. を設定します。

- [3]上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。
★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。
- [4]上記の「想定 UI 画面」の「登録」ボタンに連動させるなどして、当メソッド (APIActivateRegisterInternet メソッド)を実行します。

APIActivateRegisterTelephone メソッド

【機能】

電話による認証登録を実行します。

【構文】

<VB2010/VB6.0>

Public Function **APIActivateRegisterTelephone()** As **Boolean**

<C#>

public **bool** **APIActivateRegisterTelephone()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::**

APIActivateRegisterTelephone()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|-----------|-------------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| UI | APICertificationID | (認証登録する)認証 ID(取得専用)。新規認証 ID 生成メソッド (APIGenerationOfNewCertificationID メソッド)の実行により、新しい認証 ID が当プロパティに設定されます。 |
| UI | APIProductID | (認証登録する)プロダクト ID |
| UI | APISerialNo | (認証登録する)シリアル No. |
| UI | APILicenseKey | ライセンスキー (貴社電話担当者より告げられたライセンスキーを設定) |
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数(0:レンタル期間なし) |
| Code | APITrialPeriod | 猶予(試用)日数(0:猶予期間なし) |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

| | | |
|------|----------------------------------------------------------|-------------------------------|
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード (基本認証使用時) |
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名 (基本認証使用時) |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIWebServiceTimeout | Web サービスのタイムアウト |
| Code | APIWebServiceURL | Web サービスの URL |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 |

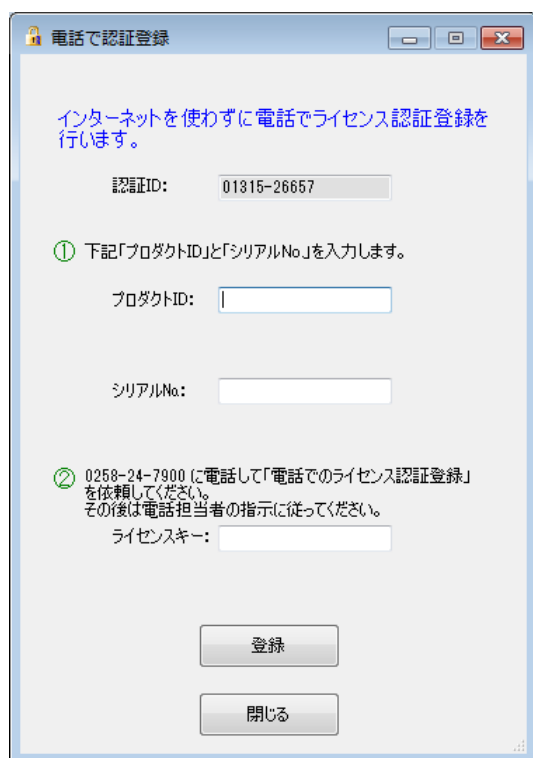
※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

電話による認証登録を実行します。

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。



この画面上の「認証 ID」は、新規認証 ID 生成メソッド (APIGenerationOfNewCertificationID メソッド) の実行で設定される認証 ID プロパティ (APICertificationID プロパティ) で取得できます。

◆処理手順

一連の処理の手順は次の通りです。

[1]新規認証 ID 生成メソッド (APIGenerationOfNewCertificationID メソッド) を実行します。
 これにより、認証 ID プロパティ (APICertificationID プロパティ) に新しい認証 ID が設定されます。
 その内容を画面に表示します。

[2]プロダクト ID とシリアル No.をエンドユーザに画面より入力させ、プロダクト ID プロパティ (APIProductID プロパティ) とシリアル No. プロパティ (APISerialNo プロパティ) にそれぞれ設定します。あるいは、別の方法で結果としてプロダクト ID プロパティ (APIProductID プロパティ) とシリアル No. プロパティ (APISerialNo プロパティ) にそれぞれ認証登録するプロダクト ID とシリアル No.

を設定します。

[3]エンドユーザに貴社に電話をしてもらいます。貴社のオペレータより聞いたライセンスキーを画面より入力させ、その値をライセンスキープロパティ(APILicenseKey プロパティ)に設定します。

[4]上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。

★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。

[5]上記の「想定 UI 画面」の「登録」ボタンに連動させるなどして、当メソッド (APIActivateRegisterTelephone メソッド)を実行します。

APIActivateRemoveInternet メソッド

【機能】

インターネットによる認証解除を実行します。

【構文】

<VB2010/VB6.0>

Public Function **APIActivateRemoveInternet()** As **Boolean**

<C#>

public **bool** **APIActivateRemoveInternet()**

<VC++>

public : **VARIANT_BOOL** **NewtonenNRvcpp::IActivation:: APIActivateRemoveInternet()**

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分※ | プロパティ | 機能 |
|-------|------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| UI | APICertificationID | (登録解除する)認証 ID(取得専用)。 (レジストリからの)認証登録済み情報の取得メソッド (APIGetRegisteredInfoFromRegistry メソッド)の実行により、登録済みの認証 ID が当プロパティに設定されます。 |
| UI | APIProductID | (認証解除する)プロダクト ID (レジストリからの)認証登録済み情報の取得メソッド (APIGetRegisteredInfoFromRegistry メソッド)の実行により、登録済みのプロダクト ID が当プロパティに設定されます。 |
| UI | APISerialNo | (認証解除する)シリアル No. (レジストリからの)認証登録済み情報の取得メソッド (APIGetRegisteredInfoFromRegistry メソッド)の実行により、登録済みのシリアル No. が当プロパティに設定されます。 |
| UI | APIUseProxyServer | プロキシサーバーの使用区分 |

| | | | |
|------|----------------------------------------------------------|------------------------|---------------|
| UI | APIProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー使用時有効 |
| UI | APIProxyServerPort | プロキシサーバーのポート | |
| UI | APIProxyServerUserName | プロキシサーバーのユーザ名 | |
| UI | APIProxyServerPassword | プロキシサーバーのパスワード | |
| Code | APIEncryptionPassword | 暗号化時のパスワード | |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 | |
| Code | APIRentalPeriod | レンタル日数 | |
| Code | APITrialPeriod | 猶予(試用)日数 | |
| Code | APIUseCpuInfo | CPU 情報の使用区分 | |
| Code | APIUseMacAddress | MAC アドレスの使用区分 | |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス | |
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード | |
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名 | |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード | |
| Code | APIWebServiceTimeout | Web サービスのタイムアウト | |
| Code | APIWebServiceURL | Web サービスの URL | |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 | |

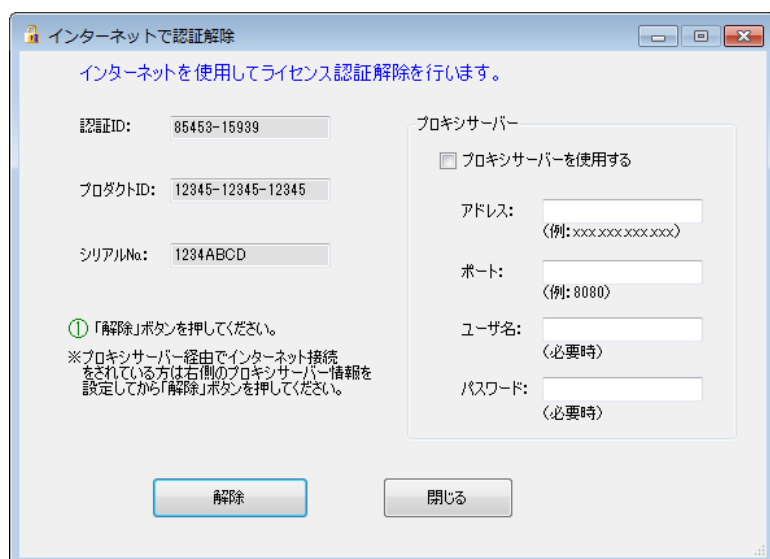
※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

インターネットによる認証解除を実行します。

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。



この画面上の「認証 ID」、「プロダクト ID」、「シリアル No.」は、(レジストリからの)認証登録済み情報の取得メソッド(APIGetRegisteredInfoFromRegistry メソッド)の実行で設定される認証 ID プロパティ(APICertificationID プロパティ)、プロダクト ID プロパティ(APIProductID プロパティ)、シリアル No. プロパティ(APISerialNo プロパティ)でそれぞれ取得できます。

◆処理手順

一連の処理の手順は次の通りです。

[1](レジストリからの)認証登録済み情報の取得メソッド(APIGetRegisteredInfoFromRegistry メソッド)を実行します。これにより、認証 ID プロパティ、プロダクト ID プロパティ、シリアル No.プロパティに各値が設定されます。

[2]認証 ID プロパティ、プロダクト ID プロパティ、シリアル No.プロパティの各値を画面に表示します。

[3]上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。

★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。

[4]上記の「想定 UI 画面」の「解除」ボタンに連動させるなどして、当メソッド(APIActivateRemoveInternet メソッド)を実行します。

APIActivateRemoveTelephone メソッド

【機能】

電話による認証解除を実行します。

【構文】

<VB2010/VB6.0>

Public Function **APIActivateRemoveTelephone()** As **Boolean**

<C#>

public **bool** **APIActivateRemoveTelephone()**

<VC++>

public : **VARIANT_BOOL** **NewtonenRvcpp::IActivation::**

APIActivateRemoveTelephone()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、正常終了時には、解除ステータスプロパティ([APIReleaseStatus](#) プロパティ)に解除ステータスが設定されます。

また、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 機能 |
|-----------|------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| UI | APICertificationID | (登録解除する)認証 ID(取得専用)。 (レジストリからの)認証登録済み情報の取得メソッド (APIGetRegisteredInfoFromRegistry メソッド)の実行により、登録済みの認証 ID が当プロパティに設定されます。 |
| UI | APIProductID | (認証解除する)プロダクト ID (レジストリからの)認証登録済み情報の取得メソッド (APIGetRegisteredInfoFromRegistry メソッド)の実行により、登録済みのプロダクト ID が当プロパティに設定されます。 |
| UI | APISerialNo | (認証解除する)シリアル No. (レジストリからの)認証登録済み情報の取得メソッド (APIGetRegisteredInfoFromRegistry メソッド) |

| | | |
|------|----------------------------------------------------------|----------------------------------------|
| | | ッド)の実行により、登録済みのシリアル No.が当プロパティに設定されます。 |
| UI | APIReleaseKey | 解除キー |
| UI | APIReleaseStatus | 解除ステータス(取得専用) |
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数 |
| Code | APITrialPeriod | 猶予(試用)日数 |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード |
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名 |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIWebServiceTimeout | Web サービスのタイムアウト |
| Code | APIWebServiceURL | Web サービスの URL |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

電話による認証解除を実行します。

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。

電話で認証解除

インターネットを使わずに電話でライセンス認証解除を行います。

認証ID: 83157-21617

プロダクトID: 12345-12345-12345

シリアルNo.: 1234ABCD

① 0258-24-7900 に電話して「電話でのライセンス認証解除」を依頼してください。
その後、電話担当者から聞いた「解除キー」を次のボックスに入力します。

解除キー: 956-11749-60711

② 次の「解除」ボタンを押してください。

解除

解除ステータス: 162-80797-49245

③ 上で表示された「解除ステータス」を電話担当者に伝えてください。

電話担当者に「解除ステータス」を伝えました。

閉じる

この画面上の「認証 ID」、「プロダクト ID」、「シリアル No.」は、(レジストリからの)認証登録済み情報の取得メソッド(APIGetRegisteredInfoFromRegistry メソッド)の実行で設定される認証 ID プロパティ(APICertificationID プロパティ)、プロダクト ID プロパティ(APIProductID プロパティ)、シリアル No. プロパティ(APISerialNo プロパティ)でそれぞれ取得できます。

この画面上の「解除キー」は、貴社オペレータより電話経由でエンドユーザが聞いて画面より入力した解除キーの値を解除キープロパティに設定します。

この画面上の「解除ステータス」は、当メソッド(APIActivateRemoveTelephone メソッド)の実行後の正常終了時に解除ステータスプロパティ(APIReleaseStatus プロパティ)で取得できます。

◆処理手順

一連の処理の手順は次の通りです。

[1](レジストリからの)認証登録済み情報の取得メソッド(APIGetRegisteredInfoFromRegistry メソッド)を実行します。これにより、認証 ID プロパティ、プロダクト ID プロパティ、シリアル No.プロパティに各値が設定されます。

[2]認証 ID プロパティ、プロダクト ID プロパティ、シリアル No.プロパティの各値を画面に表示します。

[3]解除キープロパティ(APIReleaseKey プロパティ)に、貴社オペレータより電話経由でエンドユーザが聞いて画面より入力した解除キーの値を設定します。

[4]上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。

★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。

[5]上記の「想定 UI 画面」の「解除」ボタンに連動させるなどして、当メソッド(APIActivateRemoveTelephone メソッド)を実行します。

[6]当メソッド(APIActivateRemoveTelephone メソッド)の実行後、正常終了時には、解除ステータスプロパティ(APIReleaseStatus プロパティ)に解除ステータス(文字列)が設定されますので、エンドユーザにその解除ステータスを電話で貴社のオペレータに伝えてもらいます。

APIActivateStatusCheck メソッド

【機能】

エンドユーザ PC 内での認証状態の確認を行います。

【構文】

<VB2010>

Public Function **APIActivateStatusCheck()** As **Integer**

<VB6.0>

Public Function **APIActivateStatusCheck()** As **Long**

<C#>

public **int** **APIActivateStatusCheck()**

<VC++>

public : **long** **NewtonenRvcpp::IActivation:: APIActivateStatusCheck()**

【引数】

なし

【戻り値】

0: 猶予期限切れ(猶予有効時)
 1~365: 猶予日数有
 366: 日付データの取得失敗(猶予)
 400: 未認証(猶予無効時)
 500: 認証済み
 1000: レンタル期限切れ(レンタル有効時)
 1001~2100: レンタル残日数 1~1100(戻り値から 1000 を引いて使用する)
 2101: 日付データの取得失敗(レンタル)
 -999: 認証済ハードウェア情報不一致
 -1: エラー(未設定や範囲を超えているプロパティがある)
 -21001231~-20000101: 終了した有効期限(2000/01/01~2100/12/31)
 -21001232: 日付データの取得失敗(有効期限)
 -3: PC の日付が変更されました。
 20000101~21001231: まだ有効な有効期限(2000/01/01~2100/12/31)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|-----------|-----------------------------------------|--------------------|
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数(0:レンタル期間なし) |
| Code | APITrialPeriod | 猶予(試用)日数(0:猶予期間なし) |

| | | |
|------|-------------------------------------------------------|------------------------|
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

この APIActivateStatusCheck メソッドと APIActivateStatusCheckOnline メソッドの相違点は APIActivateStatusCheck メソッドがエンドユーザ PC 内での認証状況を返すのに対し、APIActivateStatusCheckOnline メソッドはインターネットを介し貴社のデータベースにアクセスして取得した情報とエンドユーザ PC 内の情報とを照合して認証状況を返す点です。

APIActivateStatusCheck2 メソッド

【機能】

エンドユーザ PC 内での認証状態の確認を行います。
Windows の管理者でなくても実行可能です。

【構文】

<VB2010>

Public Function **APIActivateStatusCheck2**() As **Integer**

<VB6.0>

Public Function **APIActivateStatusCheck2**() As **Long**

<C#>

public **int** **APIActivateStatusCheck2**()

<VC++>

public : **long** **NewtonenNRvcpp::IActivation::APIActivateStatusCheck2**()

【引数】

なし

【戻り値】

0: 猶予期限切れ(猶予有効時)
 1~365: 猶予日数有
 366: 日付データの取得失敗(猶予)
 400: 未認証(猶予無効時)
 500: 認証済み
 1000: レンタル期限切れ(レンタル有効時)
 1001~2100: レンタル残日数 1~1100(戻り値から 1000 を引いて使用する)
 2101: 日付データの取得失敗(レンタル)
 -999: 認証済ハードウェア情報不一致
 -1: エラー(未設定や範囲を超えているプロパティがある)
 -21001231~-20000101: 終了した有効期限(2000/01/01~2100/12/31)
 -21001232: 日付データの取得失敗(有効期限)
 -3: PC の日付が変更されました。
 20000101~21001231: まだ有効な有効期限(2000/01/01~2100/12/31)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分※ | プロパティ | 設定する内容 |
|-------|-----------------------------------------|--------------------|
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数(0:レンタル期間なし) |

| | | |
|------|-------------------------------------------------------|------------------------------|
| Code | APISelectRunAppDatePathFlag | 「アプリ起動日」の読み込み/書き込み場所の切り替えフラグ |
| Code | APITrialPeriod | 猶予(試用)日数(0: 猶予期間なし) |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

以下の機能以外は従来の `APIActivateStatusCheck` メソッドと同等です。

[APISelectRunAppDatePathFlag](#) プロパティ値による場所+従来の

[APIVendorsProductStartRegistryKeyPath](#) プロパティ値の場所に「アプリ起動日」の読み込み/書き込みを行います。

このメソッドは、ユーザー様からのご要望で追加されました。

認証レスキュー！2 は、認証情報などをレジストリの

`HKEY_LOCAL_MACHINE + APIVendorsProductStartRegistryKeyPath` プロパティ値の場所へ書き込み/読み込みを行っています。

`APIRentalPeriod` プロパティ(レンタル日数)あるいは、`APITrialPeriod` プロパティ(猶予(試用)日数)が 0 以上に設定されていて、このメソッドが初めて実行された場合、残日数を確認するための開始日をレジストリに書き込みます。

その後、残日数が切れるまではレジストリへの書き込み処理はありませんでした。

しかし、バージョン 2.6.2 以降は、故意に PC の日付が変更されることを防ぐために、このメソッドを起動する度にレジストリに情報を書き込むよう変更されました。

前出のユーザー様は、Windows の管理者でなくても認証状況を確認したいとのご要望で、この故意に PC の日付が変更されることを防ぐための書き込み部分のみ「`HKEY_CURRENT_USER`(レジストリ)」あるいは「`ProgramData`(フォルダ)」へ書き込みを変更するため、このメソッドが追加されました。

APIActivateStatusCheckOnline メソッド

【機能】

オンラインで認証状態の確認を行います。

【構文】

<VB2010>

Public Function **APIActivateStatusCheckOnline()** As **Integer**

<VB6.0>

Public Function **APIActivateStatusCheckOnline()** As **Long**

<C#>

public **int** **APIActivateStatusCheckOnline()**

<VC++>

public : **long** **NewtonenRvcpp::IActivation:: APIActivateStatusCheckOnline()**

【引数】

なし

【戻り値】

- 0: PC レベルで認証されていない(PC のレジストリには認証登録情報がない)
- 1: OK(PC レベルと DB で認証登録情報が一致した)
- 2: NG(PC レベルと一致する認証登録情報が DB にない)
- 3: NG(認証済ハードウェア情報不一致)
- 4: NG(日付データの取得失敗(猶予))
- 5: NG(日付データの取得失敗(レンタル))
- 6: NG(日付データの取得失敗(有効期限))
- 11: 接続できない(認証時は電話)
- 12: 接続できない(認証時は代理)
- 999: その他エラー(接続できないなど)
- 1: エラー(未設定や範囲を超えているプロパティがある)
- 3: PC の日付が変更されました。

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 機能 | |
|--------|----------------------------------------|----------------|-----------------|
| UI | APIUseProxyServer | プロキシサーバーの使用区分 | |
| UI | APIProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー 使用時 |
| UI | APIProxyServerPort | プロキシサーバーのポート | |
| UI | APIProxyServerUserName | プロキシサーバーのユーザ名 | |
| UI | APIProxyServerPassword | プロキシサーバーのパスワード | |

| | ード |
|------|------------------------------------------------------------------------------------------|
| Code | APIEncryptionPassword 暗号化時のパスワード |
| Code | APIEncryptionSaltString 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod レンタル日数(0:レンタル期間なし) |
| Code | APITrialPeriod 猶予(試用)日数(0:猶予期間なし) |
| Code | APIUseCpuInfo CPU 情報の使用区分 |
| Code | APIUseMacAddress MAC アドレスの使用区分 |
| Code | APIVendorsProductStartRegistryKeyPath ベンダアプリケーション開始レジストリキーパス |
| Code | APIWebServiceBasicAuthenticationPassword Web サービス時の基本認証パスワード(基本認証使用時) |
| Code | APIWebServiceBasicAuthenticationUserName Web サービス時の基本認証ユーザ名(基本認証使用時) |
| Code | APIWebServiceCheckPassword Web サービス確認パスワード |
| Code | APIWebServiceTimeout Web サービスのタイムアウト |
| Code | APIWebServiceURL Web サービスの URL |
| Code | APIWebServiceUseBasicAuthentication Web サービス時の基本認証の使用区分 |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

この APIActivateStatusCheckOnline メソッドと APIActivateStatusCheck メソッドとの相違点は APIActivateStatusCheck メソッドがエンドユーザ PC 内での認証状況を返すのに対し、APIActivateStatusCheckOnline メソッドはインターネットを介し貴社のデータベースにアクセスして取得した情報とエンドユーザ PC 内の情報とを照合して認証状況を返す点です。

この APIActivateStatusCheckOnline メソッドは、なんらかの理由で現在エンドユーザが使用している貴社のアプリケーションを使用不可としたい場合などに利用できます。

通常の(レンタル期間がない)認証登録の場合、レジストリとハード情報だけの確認で OK となっても貴社がデータベース(DB)上の当該情報を削除してもエンドユーザの PC ではアプリケーションが継続して使用できてしまいます。そこで、任意のタイミングでインターネットを介し貴社のデータベースにアクセスしてそれらの情報を確認できる機能がこのメソッドです。貴社はたとえば、アプリケーションの起動時にそのメソッドを利用するコードを記述し、その戻り値によってアプリケーションを(強制的に)終了する、といった挙動を制御できます。

APIActivateStatusCheckOnline2 メソッド

【機能】

オンラインで認証状態の確認を行います。
Windows の管理者でなくても実行可能です。

【構文】

<VB2010>

Public Function **APIActivateStatusCheckOnline2()** As **Integer**

<VB6.0>

Public Function **APIActivateStatusCheckOnline2()** As **Long**

<C#>

public **int** **APIActivateStatusCheckOnline2()**

<VC++>

public : **long** **NewtonenRvcpp::IActivation:: APIActivateStatusCheckOnline2()**

【引数】

なし

【戻り値】

- 0: PC レベルで認証されていない(PC のレジストリには認証登録情報がない)
- 1: OK(PC レベルと DB で認証登録情報が一致した)
- 2: NG(PC レベルと一致する認証登録情報が DB にない)
- 3: NG(認証済ハードウェア情報不一致)
- 4: NG(日付データの取得失敗(猶予))
- 5: NG(日付データの取得失敗(レンタル))
- 6: NG(日付データの取得失敗(有効期限))
- 11: 接続できない(認証時は電話)
- 12: 接続できない(認証時は代理)
- 999: その他エラー(接続できないなど)
- 1: エラー(未設定や範囲を超えているプロパティがある)
- 3: PC の日付が変更されました。

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 機能 | |
|--------|----------------------------------------|---------------|-----------------|
| UI | APIUseProxyServer | プロキシサーバーの使用区分 | |
| UI | APIProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー 使用時 |
| UI | APIProxyServerPort | プロキシサーバーのポート | |
| UI | APIProxyServerUserName | プロキシサーバーのユーザ名 | |

| | | |
|------|----------------------------------------------------------|------------------------------|
| UI | APIProxyServerPassword | プロキシサーバーのパスワード |
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数(0:レンタル期間なし) |
| Code | APITrialPeriod | 猶予(試用)日数(0:猶予期間なし) |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード(基本認証使用時) |
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名(基本認証使用時) |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIWebServiceTimeout | Web サービスのタイムアウト |
| Code | APIWebServiceURL | Web サービスの URL |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

以下の機能以外は従来の `APIActivateStatusCheckOnline` メソッドと同等です。

[APISelectRunAppDatePathFlag](#) プロパティ値による場所+従来の

[APIVendorsProductStartRegistryKeyPath](#) プロパティ値の場所に「アプリ起動日」の読み込み/書き込みを行います。

このメソッドは、ユーザ様からのご要望で追加されました。

認証レスキュー！2 は、認証情報などをレジストリの

`HKEY_LOCAL_MACHINE + APIVendorsProductStartRegistryKeyPath` プロパティ値の場所へ書き込み/読み込みを行っています。

`APIRentalPeriod` プロパティ(レンタル日数)あるいは、`APITrialPeriod` プロパティ(猶予(試用)日数)が 0 以上に設定されていて、このメソッドが初めて実行された場合、残日数を確認するための開始日をレジストリに書き込みます。

その後、残日数が切れるまではレジストリへの書き込み処理はありませんでした。

しかし、バージョン 2.6.2 以降は、故意に PC の日付が変更されることを防ぐために、このメソッドを起動する度にレジストリに情報を書き込むよう変更されました。

前出のユーザ様は、Windows の管理者でなくても認証状況を確認したいとのご要望で、この故意に PC の日付が変更されることを防ぐための書き込み部分のみ「`HKEY_CURRENT_USER`(レジストリ)」あるいは「`ProgramData`(フォルダ)」へ書き込みを変更するため、このメソッドが追加されました。

APIDeterminationOfProxyUpdateOfExpirationDate メソッド

【機能】

「代理有効期限更新/確定」を実行します。(有効期限を更新したい PC で利用)

【構文】

<VB2010/VB6.0>

Public Function **APIDeterminationOfProxyUpdateOfExpirationDate** () As **Boolean**

<C#>

public **bool** **APIDeterminationOfProxyUpdateOfExpirationDate**()

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::**

APIDeterminationOfProxyUpdateOfExpirationDate ()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|-----------|------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| UI | APIProxyDataPath | (有効期限を更新する)代理有効期限取得データのパス ファイルの拡張子は「.NRS」です。 |
| UI | APIProductID | (有効期限を更新する)プロダクト ID 代理有効期限取得データの読み込みメソッド (APIGetProxyDataForExpirationDate メソッド) の実行により、代理有効期限取得データから取得したプロダクト ID が当プロパティに設定されます。 |
| UI | APISerialNo | (有効期限を更新する)シリアル No. 代理有効期限取得データの読み込みメソッド (APIGetProxyDataForExpirationDate メソッド) の実行により、代理有効期限取得データから取得したシリアル No.が当プロパティに設定されます。 |
| UI | APICurrentExpirationDate | (有効期限を更新する)現在の有効期限 代理有効期限取得データの読み込みメソッド (APIGetProxyDataForExpirationDate メソッド) |

| | | |
|------|-------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| | | の実行により、代理有効期限取得データから取得した現在の有効期限が当プロパティに設定されます。 |
| UI | APINewExpirationDate | (有効期限を更新する)新しい有効期限。代理有効期限取得データの読み込みメソッド (APIGetProxyDataForExpirationDate メソッド) の実行により、代理有効期限取得データから取得した新しい有効期限が当プロパティに設定されます。 |
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数(0:レンタル期間なし) |
| Code | APITrialPeriod | 猶予(試用)日数(0:猶予期間なし) |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

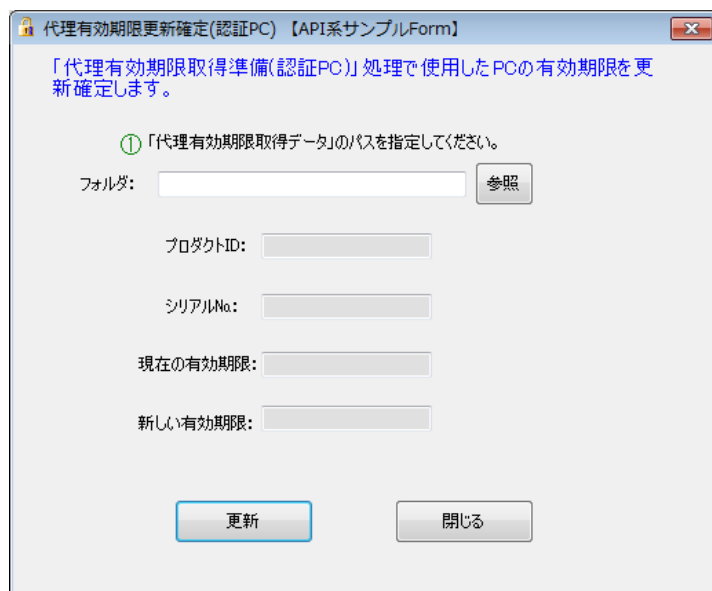
※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

「代理有効期限更新/確定」を実行します。(有効期限を更新したい PC で利用)

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。



この画面上の「プロダクト ID」、「シリアル No.」、「現在の有効期限」、「新しい有効期限」は、代理有効期限取得データの取得メソッド (APIGetProxyDataForExpirationDate メソッド) の実行で設定されるプロダクトID プロパティ (APIProductID プロパティ)、シリアル No.プロパティ (APISerialNo プロパティ)、現在の有効期限プロパティ (APICurrentExpirationDate プロパティ)、新しい有効期限プロパティ (APINewExpirationDate プロパティ) でそれぞれ取得できます。

◆処理手順

一連の処理の手順は次の通りです。

[1] 代理有効期限取得データパスをエンドユーザに画面より入力させ、代理有効期限取得データパスプロパティ(APIProxyDataPath プロパティ)に設定します。この際、ファイルの拡張子は「.NRS」とします。

[2] 代理有効期限取得データの取得メソッド(APIGetProxyDataForExpirationDate メソッド)を実行します。

これにより、プロダクト ID プロパティ(APIProductID プロパティ)、シリアル No.プロパティ(APISerialNo プロパティ)、現在の有効期限プロパティ(APICurrentExpirationDate プロパティ)、新しい有効期限プロパティ(APINewExpirationDate プロパティ)に代理有効期限取得データから取得したプロダクトID、シリアル No.、現在の有効期限、新しい有効期限それぞれが設定されます。それらの内容を画面に表示します。この内容表示そのものは必須ではありません。

[3]上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。

★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。

[5] 上記の「想定 UI 画面」の「更新」ボタンに連動させるなどして、当メソッド(APIDeterminationOfProxyUpdateOfExpirationDate メソッド)を実行します。

APIGenerationOfNewCertificationID メソッド

【機能】

認証登録時に必要な、新しい認証 ID を生成します。

【構文】

<VB2010/VB6.0>

Public Function **APIGenerationOfNewCertificationID()** As **Boolean**

<C#>

public **bool** **APIGenerationOfNewCertificationID()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::**

APIGenerationOfNewCertificationID()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|-----------|-------------------------------------------------------|------------------------|
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数(0:レンタル期間なし) |
| Code | APITrialPeriod | 猶予(試用)日数(0:猶予期間なし) |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

認証登録時に必要な、新しい認証 ID を生成します。

[1]このメソッドの戻り値は成功か失敗(True/False)です。

[2]認証登録済みの場合は、当メソッドは失敗します。

[3]このメソッドの成功時のみ、認証 ID プロパティ(APICertificationID プロパティ)に新しく生成された認証 ID が設定されます。

APIGetFreeItem メソッド

【機能】

プロダクト ID とシリアル No.を指定して、ActivationKey テーブルより自由入力項目を取得します。

【構文】

<VB2010/VB6.0>

Public Function **APIGetFreeItem** () As **Boolean**

<C#>

public **bool** **APIGetFreeItem** ()

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation:: APIGetFreeItem** ()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分※ | プロパティ | 設定する内容 | |
|-------|----------------------------------------------------------|------------------------------|-------------|
| UI | APIProductID | (認証登録する)プロダクト ID | |
| UI | APISerialNo | (認証登録する)シリアル No. | |
| UI | APIUseProxyServer | プロキシサーバーの使用区分 | |
| UI | APIProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー使用時 |
| UI | APIProxyServerPort | プロキシサーバーのポート | |
| UI | APIProxyServerUserName | プロキシサーバーのユーザ名 | |
| UI | APIProxyServerPassword | プロキシサーバーのパスワード | |
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード(基本認証使用時) | |
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名(基本認証使用時) | |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード | |
| Code | APIWebServiceTimeout | Web サービスのタイムアウト | |
| Code | APIWebServiceURL | Web サービスの URL | |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 | |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

プロダクト ID とシリアル No.を指定して、ActivationKeyTable より自由入力項目を取得します。当メソッド(APIGetFreeItem メソッド)実行後、APIFreeItem1～APIFreeItem5 プロパティに値が設定されます。

APIGetProductIdSerialNoList メソッド

【機能】

外部データベースとのリンク用キー項目を指定し ActivationKey テーブルよりプロダクト ID とシリアル No.のペア文字列の列挙を取得します。

【構文】

<VB2010>

Public Function **APIGetProductIdSerialNoList** () As **Integer**

<VB6.0>

Public Function **APIGetProductIdSerialNoList** () As **Long**

<C#>

public **int** **APIGetProductIdSerialNoList** ()

<VC++>

public : **long** **NewtoneNRvcpp::IActivation:: APIGetProductIdSerialNoList** ()

【引数】

なし

【戻り値】

取得したプロダクト ID とシリアル No.のペアリングの数。

エラー時は 0 が設定され、戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

また、APIProductIdSerialNoList プロパティに取得したプロダクト ID とシリアル No.をデリミタのカンマ (,) で区切った文字列が設定されます。該当するデータが存在しない時は、"" (空文字列) が設定されます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 | |
|-----------|----------------------------------------------------------|----------------------------------------------------------------|-----------------|
| UI | APIExternalLinkKey | APIGetProductIdSerialNoList メソッド 実行時の外部データベースとのリンク用キー項目を設定します。 | |
| UI | APIUseProxyServer | プロキシサーバーの使用区分 | |
| UI | APIProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー 使用時 |
| UI | APIProxyServerPort | プロキシサーバーのポート | |
| UI | APIProxyServerUserName | プロキシサーバーのユーザー名 | |
| UI | APIProxyServerPassword | プロキシサーバーのパスワード | |
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード (基本認証使用時) | |

| | | |
|------|----------------------------------------------------------|-----------------------------|
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名(基本認証使用時) |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIWebServiceTimeout | Web サービスのタイムアウト |
| Code | APIWebServiceURL | Web サービスの URL |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

APIExternalLinkKey プロパティに外部データベースとのリンク用キーを指定し ActivationKey テーブルよりプロダクト ID とシリアル No.のペア文字列の列挙を取得します。

認証レスキュー！とは別の外部データベースとのリンク用キーを APIExternalLinkKey プロパティに設定し、当メソッド(APIGetProductIdSerialNoList メソッド)を実行すると、認証レスキュー！のデータベースの ActivationKey テーブルより該当するプロダクト ID とシリアル No.のペア文字列の列挙をデリミタ付きで APIProductIdSerialNoList プロパティに設定して返します。

プロダクト ID とシリアル No.のペア文字列の列挙は、たとえば次のような文字列です。
デリミタはカンマ(,)です。

1111-1111-1111,aaaa1111,22222-22222-22222,bbbb2222,33333-33333-33333,cccc3333

この例では次の 3 組のプロダクト ID とシリアル No.が返されました。

プロダクト ID ="11111-11111-11111" シリアル No.="aaaa1111"

プロダクト ID ="22222-22222-22222" シリアル No.="bbbb2222"

プロダクト ID ="33333-33333-33333" シリアル No.="cccc3333"

当メソッド(APIGetProductIdSerialNoList メソッド)の具体的な利用方法の例は次のようなものがあります。

貴社のアプリケーション内でエンドユーザに対し、複数のプロダクト ID とシリアル No.のペアを一括で認証登録させたい場合を考えます。

手順 1

貴社のアプリケーション内の一括認証処理の UI 上でエンドユーザにユニークな「リンク用キー」(例: ユーザ ID)を入力させる。

手順 2

エンドユーザに入力させた「リンク用キー」を APIExternalLinkKey プロパティに設定し、他の必須設定プロパティの設定後、当メソッド(APIGetProductIdSerialNoList メソッド)を実行する。

手順 3

認証レスキュー！の認証用 DLL が、「Activationkey テーブル」上のその「リンク用キー」に該当する全レコード分のプロダクト ID とシリアル No.の列挙をデリミタ付きの文字列として APIProductIdSerialNoList プロパティに設定して返す。

手順 4

一括認証処理の UI 上にそれらの返されたプロダクト ID とシリアル No.を一覧表示し、エンドユーザの確認を求めた後、インターネットによる認証登録(APIActivateRegisterInternet メソッド)で認証登録する。

APIGetProxyDataForExpirationDate メソッド

【機能】

代理有効期限取得データを取得します。

【構文】

<VB2010/VB6.0>

Public Function **APIGetProxyDataForExpirationDate**() As **Boolean**

<C#>

public **bool** **APIGetProxyDataForExpirationDate** ()

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::**

APIGetProxyDataForExpirationDate()

【引数】

なし

【戻り値】

True: 正常

False: エラー (未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 機能 |
|-----------|-----------------------------------------|------------------------------------------------|
| UI | APIProxyDataPath | (有効期限を更新する)代理有効期限取得データのパスワードファイルの拡張子は「.NRS」です。 |
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

代理有効期限取得データを取得します。

[1]このメソッドの戻り値は成功か失敗(True/False)です。

[2]代理認証データのパスや内容が正しくない場合は、当メソッドは失敗します。

[3]このメソッドの成功時のみ、次表の各プロパティに値が設定されます。

代理有効期限更新関連のメソッドによって、事前に当メソッド(APIGetProxyDataForExpirationDateメソッド)の実行が必要かどうかや設定されるプロパティが異なります。

◆当メソッド(APIGetProxyDataForExpirationDateメソッド)の利用形態

| 代理有効期限更新関連のメソッド | 事前の当該メソッドの実行 | プロパティ | | | |
|-----------------------------------------------------------------------------------|--------------|-------------------------------------------------------|------------------------------------------------------|---------------------------------------------------------------------|-----------------------------------------------------------------|
| | | プロダクト ID プロパティ (APIProductID プロパティ) | シリアル No. プロパティ (APISerialNo プロパティ) | 現在の有効期限 (取得専用) (APICurrentExpirationDate プロパティ) | 新しい有効期限 (取得専用) (APINewExpirationDate プロパティ) |
| 代理有効期限更新準備 (APIPreparationOfProxyUpdateOfExpirationDate メソッド) | 不要 | 設定不要 | 設定不要 | 設定不要 | 設定不要 |
| 代理有効期限取得 (APIGetProxyUpdateOfExpirationDate メソッド) | 必要 | 当メソッドで設定 | 当メソッドで設定 | 当メソッドで設定 | APIGetProxyUpdateOfExpirationDate メソッドより取得 |
| 代理有効期限更新確定 (APIDeterminationOfProxyUpdateOfExpirationDate メソッド) | 必要 | 当メソッドで設定 | 当メソッドで設定 | 当メソッドで設定 | 当メソッドで設定 |

[4] 当メソッドの実行には上記の【必須設定プロパティ】にある、代理有効期限取得データのパス (APIProxyDataPath プロパティ) に適切な設定が必要です。

APIGetProxyDataForRegister メソッド

【機能】

代理認証登録データを取得します。

(代理認証機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function **APIGetProxyDataForRegister** () As **Boolean**

<C#>

public **bool** **APIGetProxyDataForRegister** ()

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::APIGetProxyDataForRegister**
()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 機能 |
|-----------|-----------------------------------------|-------------------------------------------|
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| UI | APIProxyDataPath | (認証登録する)代理認証データのパス。 ファイルの拡張子は「.NRS」です。 |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

代理認証登録データを取得します。

[1]このメソッドの戻り値は成功か失敗(True/False)です。

[2]代理認証データのパスや内容が正しくない場合は、当メソッドは失敗します。

[3]このメソッドの成功時のみ、次表の各プロパティに値が設定されます。

代理認証関連のメソッドによって、事前に当メソッド(APIGetProxyDataForRegister メソッド)の実行が必要かどうかや設定されるプロパティが異なります。

◆当メソッド(APIGetProxyDataForRegister メソッド)の利用形態

| 代理認証 関連のメソッド | 事前の当 メソッドの 実行 | プロパティ | | |
|--------------------------------------------------------------------------|---------------------|---------------------------------------------------------------|-------------------------------------------------------------------|------------------------------------------------------------------|
| | | 認証 ID プロパティ (APICertificationID プロパティ) | プロダクト ID プロ パ テ ィ (APIProductID プ ロパティ) | シリアル No.プ ロ パ テ ィ (APISerialNo プ ロパティ) |
| 代理認証登録準備 (APIProxyActivateRegi sterPrepare メソッド) | 不要 | 設定不要 | 設定不要 | 設定不要 |
| 代理認証登録実行 (APIProxyActivateRegi sterExecute メソッド) | 必要 | 当メソッドで設定 | UI 画面上などか ら設定 | UI 画面上など から設定 |
| 代理認証登録確定 (APIProxyActivateRegi sterFix メソッド) | 必要 | 当メソッドで設定 | 当メソッドで設定 | 当メソッドで設 定 |

[4]当メソッドの実行には上記の【必須設定プロパティ】にある、代理認証データパスプロパティ (APIProxyDataPath プロパティ)に適切な設定が必要です。

APIGetProxyDataForRemove メソッド

【機能】

代理認証解除データを取得します。
 (代理認証機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function **APIGetProxyDataForRemove** () As **Boolean**

<C#>

public **bool** **APIGetProxyDataForRemove** ()

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation:: APIGetProxyDataForRemove**
 ()

【引数】

なし

【戻り値】

True: 正常
 False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 機能 |
|-----------|-----------------------------------------|-------------------------------------------|
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| UI | APIProxyDataPath | (認証登録する)代理認証データのパス。 ファイルの拡張子は「.NRS」です。 |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

代理認証解除データを取得します。

[1]このメソッドの戻り値は成功か失敗(True/False)です。

[2]代理認証データのパスや内容が正しくない場合は、当メソッドは失敗します。

[3]このメソッドの成功時のみ、次表の各プロパティに値が設定されます。

代理認証関連のメソッドによって、事前に当メソッド(APIGetProxyDataForRemove メソッド)の実行が必要かどうかや設定されるプロパティが異なります。

◆当メソッド(APIGetProxyDataForRemove メソッド)の利用形態

| 代理認証 関連のメソッド | 事前の当 メソッドの 実行 | プロパティ | | |
|------------------------------------------------------------------------|---------------------|---------------------------------------------------------------|-------------------------------------------------------------------|------------------------------------------------------------------|
| | | 認証 ID プロパティ (APICertificationID プロパティ) | プロダクト ID プロ パ テ ィ (APIProductID プ ロパティ) | シリアル No.プ ロ パ テ ィ (APISerialNo プ ロパティ) |
| 代理認証解除準備 (APIProxyActivateRem ovePrepare メソッド) | 不要 | 設定不要 | 設定不要 | 設定不要 |
| 代理認証解除実行 (APIProxyActivateRem oveExecute メソッド) | 必要 | 当メソッドで設定 | 当メソッドで設定 | 当メソッドで設 定 |

[4]当メソッドの実行には上記の【必須設定プロパティ】にある、代理認証データパスプロパティ (APIProxyDataPath プロパティ)に適切な設定が必要です。

APIGetProxyUpdateOfExpirationDate メソッド

【機能】

「代理有効期限/取得」を実行します。(代理 PC で利用)

【構文】

<VB2010/VB6.0>

Public Function **APIGetProxyUpdateOfExpirationDate()** As **Boolean**

<C#>

public **bool** **APIGetProxyUpdateOfExpirationDate()**

<VC++>

public : **VARIANT_BOOL** **NewtonenRvcpp::IActivation::**

APIGetProxyUpdateOfExpirationDate()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 | |
|-----------|---------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------|----------|
| UI | APIProxyDataPath | (有効期限を更新する)代理有効期限取得データのパス ファイルの拡張子は「.NRS」です。 | |
| UI | APIProductID | (代理有効期限を取得する)プロダクト ID 代理有効期限取得データの取得メソッド (APIGetProxyDataForExpirationDate メソッド)の実行により、代理有効期限取得データから取得したプロダクト ID が当プロパティに設定されます。 | |
| UI | APISerialNo | (代理有効期限を取得する)シリアル No. 代理有効期限取得データの取得メソッド (APIGetProxyDataForExpirationDate メソッド)の実行により、代理有効期限取得データから取得したシリアル No.が当プロパティに設定されます。 | |
| UI | APIUseProxyServer | プロキシサーバーの使用区分 | |
| UI | APIProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー |
| UI | APIProxyServerPort | プロキシサーバーのポート | |

| | | | |
|------|----------------------------------------------------------|-------------------------------|-----|
| UI | APIProxyServerUserName | プロキシサーバーのユーザ名 | 使用時 |
| UI | APIProxyServerPassword | プロキシサーバーのパスワード | |
| Code | APIEncryptionPassword | 暗号化時のパスワード | |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 | |
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード (基本認証使用時) | |
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名 (基本認証使用時) | |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード | |
| Code | APIWebServiceTimeout | Web サービスのタイムアウト | |
| Code | APIWebServiceURL | Web サービスの URL | |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 | |

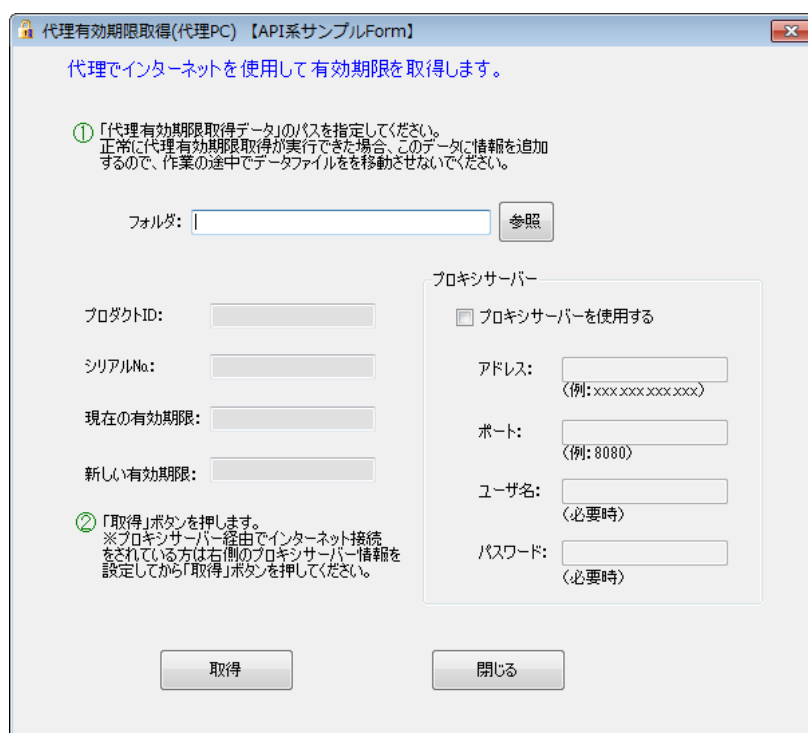
※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

「代理有効期限/取得」を実行します。(代理 PC で利用)

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。



この画面上の「プロダクト ID」、「シリアル No.」、「現在の有効期限」は、代理有効期限取得データの取得メソッド(APIGetProxyDataForExpirationDate メソッド)の実行で設定されるプロダクト ID プロパティ(APIProductID プロパティ)、シリアル No.プロパティ(APISerialNo プロパティ)、現在の有効期限プロパティ(APICurrentExpirationDate プロパティ)で取得できます。

◆処理手順

一連の処理の手順は次の通りです。

- [1] 代理有効期限取得データパスをエンドユーザに画面より入力させ、代理有効期限取得データパスプロパティ(APIProxyDataPath プロパティ)に設定します。この際、ファイルの拡張子は「.NRS」とします。
- [2] 代理有効期限取得データの取得メソッド(APIGetProxyDataForExpirationDate メソッド)を実行します。
これにより、プロダクト ID プロパティ(APIProductID プロパティ)、シリアル No.プロパティ(APISerialNo プロパティ)、現在の有効期限プロパティ(APICurrentExpirationDate プロパティ)に代理有効期限取得データから取得したプロダクト ID、プロパティ、現在の有効期限が設定されます。その内容を画面に表示します。この内容表示そのものは必須ではありません。
- [3] 代理有効期限取得データから取得したプロダクト ID とシリアル No.をプロダクト ID プロパティ(APIProductID プロパティ)とシリアル No. プロパティ(APISerialNo プロパティ)にそれぞれ設定します。あるいは、別の方法で結果としてプロダクト ID プロパティ(APIProductID プロパティ)とシリアル No. プロパティ(APISerialNo プロパティ)にそれぞれ認証登録するプロダクト ID とシリアル No.を設定します。
- [4] 上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。
★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。
- [5] 上記の「想定 UI 画面」の「取得」ボタンに連動させるなどして、当メソッド(APIGetProxyUpdateOfExpirationDate メソッド)を実行します。

APIGetRegisteredInfoFromRegistry メソッド

【機能】

レジストリから認証登録済みの情報を取得します。

【構文】

<VB2010/VB6.0>

Public Function **APIGetRegisteredInfoFromRegistry()** As **Boolean**

<C#>

public **bool** **APIGetRegisteredInfoFromRegistry()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::**

APIGetRegisteredInfoFromRegistry()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|-----------|-------------------------------------------------------|------------------------|
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数(0:レンタル期間なし) |
| Code | APITrialPeriod | 猶予(試用)日数(0:猶予期間なし) |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

レジストリから認証登録済みの情報を取得します。

[1]このメソッドの戻り値は成功か失敗(True/False)。

[2]認証されていない場合は、当メソッドは失敗します。

[3]このメソッドの成功時のみ、次のプロパティに認証登録済みの各情報が設定されます。

- ・プロダクト ID (APIProductID プロパティ)
- ・シリアル No. (APISerialNo プロパティ)
- ・認証 ID (APICertificationID プロパティ)
- ・ライセンスキー (APILicenseKey プロパティ)
- ・現在の有効期限 (APICurrentExpirationDate)

[4]これらのプロパティは、正常に読み込みが成功しない限り、初期値(空文字列)が設定されます。

APIGetRegisteredInfoFromRegistry2 メソッド

【機能】

レジストリから認証登録済みの情報を取得します。
Windows の管理者でなくても実行可能です。

【構文】

<VB2010/VB6.0>

Public Function **APIGetRegisteredInfoFromRegistry2()** As **Boolean**

<C#>

public **bool** **APIGetRegisteredInfoFromRegistry2()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::**

APIGetRegisteredInfoFromRegistry2()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|-----------|-------------------------------------------------------|------------------------------|
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数(0:レンタル期間なし) |
| Code | APISelectRunAppDatePathFlag | 「アプリ起動日」の読み込み/書き込み場所の切り替えフラグ |
| Code | APITrialPeriod | 猶予(試用)日数(0:猶予期間なし) |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

以下の機能以外は従来の [APIGetRegisteredInfoFromRegistry](#) メソッドと同等です。

[APISelectRunAppDatePathFlag](#) プロパティ値による場所+従来の

[APIVendorsProductStartRegistryKeyPath](#) プロパティ値の場所に「アプリ起動日」の読み込み/書き

込みを行います。

このメソッドは、ユーザ様からのご要望で追加いたしました。

Windows の管理者でなくても [APIActivateStatusCheck2](#) メソッドを使用して認証状態を確認後、認証情報を取得するために、このメソッドが必要となり追加されました。

APIGetRegisteredInfoFromWeb メソッド

【機能】

インターネットを使用して ActivationKeyTable から「ライセンス数」「有効期限の利用」「有効期限」を、ActivationDataTable から「認証 ID」「認証日時(作成日)」を取得します。

【構文】

<VB2010/VB6.0>

Public Function **APIGetRegisteredInfoFromWeb**() As **String**

<C#>

public **string** **APIGetRegisteredInfoFromWeb**()

<VC++>

public : **_bstr_t** **NewtoneNRvcpp::IActivation:: APIGetRegisteredInfoFromWeb**()

【引数】

なし

【戻り値】

- ①設定ライセンス数
- ②認証済ライセンス数
- ③有効期限有無
- ④有効期限
- ②分の「認証 ID」と「認証日時(作成日)」

として文字列で返します。

該当する認証キーが存在しない場合は、空を返します。

また、各項目の値を デリミタの カンマ(,)で区切った文字列が設定されます。

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|--------|----------------------------------------------------------|-------------------------------|
| UI | APIProductID | (認証登録する)プロダクト ID |
| UI | APISerialNo | (認証登録する)シリアル No. |
| UI | APIUseProxyServer | プロキシサーバーの使用区分 |
| UI | APIProxyServerAddress | プロキシサーバーのアドレス |
| UI | APIProxyServerPort | プロキシサーバーのポート |
| UI | APIProxyServerUserName | プロキシサーバーのユーザ名 |
| UI | APIProxyServerPassword | プロキシサーバーのパスワード |
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード (基本認証使用時) |
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名 (基 |

| | | |
|------|-----------------------------------------------------|---------------------|
| | | 本認証使用时) |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIWebServiceTimeout | Web サービスのタイムアウト |
| Code | APIWebServiceURL | Web サービスの URL |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 |

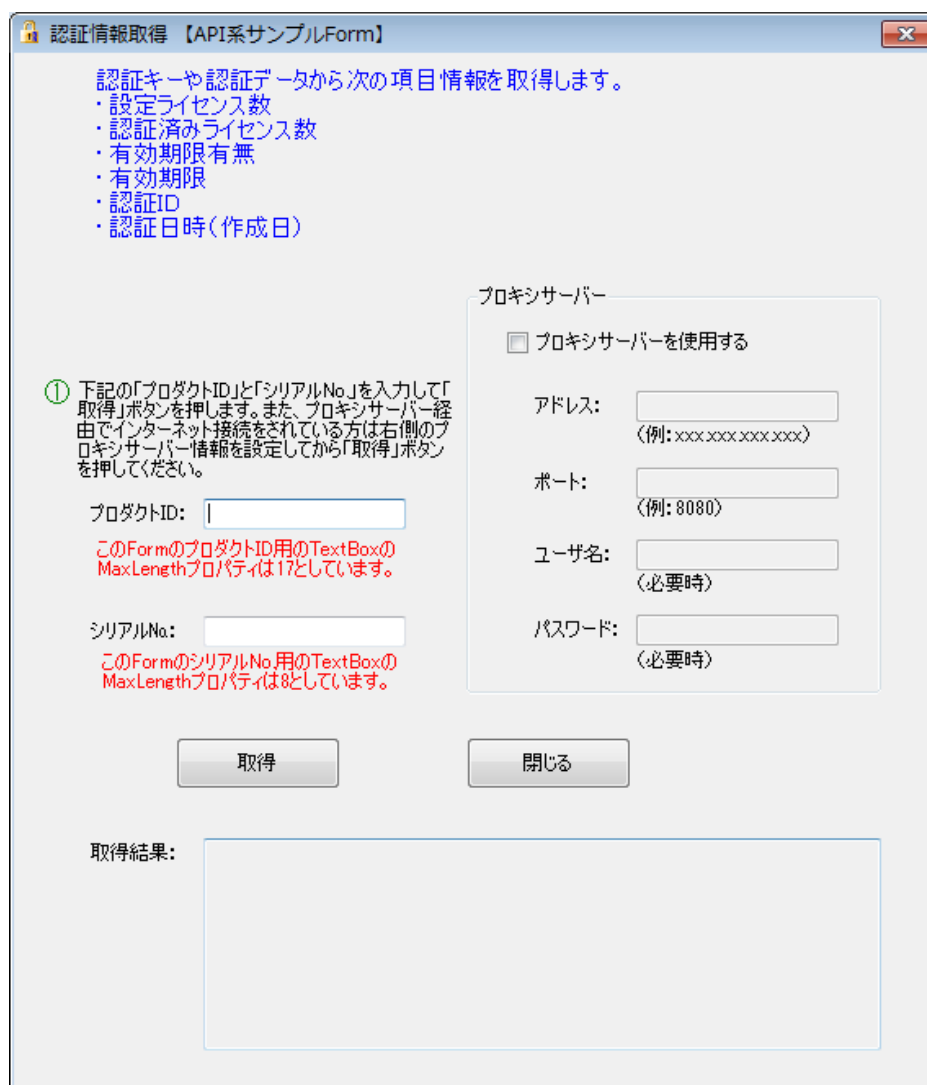
※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

インターネットを使用して ActivationKeyTable から「ライセンス数」「有効期限の利用」「有効期限」を、ActivationDataTable から「認証 ID」「認証日時(作成日)」を取得します。

◆想定UI画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。



◆処理手順

一連の処理の手順は次の通りです。

- [1]プロダクト ID とシリアル No. をエンドユーザに画面より入力させ、プロダクト ID プロパティ (APIProductID プロパティ)とシリアル No. プロパティ(APISerialNo プロパティ)にそれぞれ設定します。あるいは、別の方法で結果としてプロダクト ID プロパティ(APIProductID プロパティ)とシリアル No プロパティ(APISerialNo プロパティ)にそれぞれ認証登録するプロダクト ID と

シリアル No.を設定します。

[2]上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。

★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。

[3]上記の「想定 UI 画面」の「取得」ボタンに連動させるなどして、当メソッド (APIGetRegisteredInfoFromWeb メソッド)を実行します。

取得した文字列の列挙は、たとえば次のような文字列です。

デリミタはカンマ(,)です。

1,1,0,,50533-21818,2014/03/24 15:42:00

この例では、設定ライセンス 1 の認証済みの結果が返されました。

設定ライセンス数="1"

認証済ライセンス数="1"

有効期限有無="0"

有効期限=""

認証 ID="50533-21818"

認証日時(作成日)="2014/03/24 15:42:00"

APIPreparationOfProxyUpdateOfExpirationDate メソッド

【機能】

「代理有効期限取得/準備」を実行します。(有効期限を更新したい PC で利用)

【構文】

<VB2010/VB6.0>

Public Function APIPreparationOfProxyUpdateOfExpirationDate() As Boolean

<C#>

public bool APIPreparationOfProxyUpdateOfExpirationDate()

<VC++>

public : VARIANT_BOOL NewtonenNRvcpp::IActivation::

APIPreparationOfProxyUpdateOfExpirationDate()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|--------|-------------------------------------------------------|-------------------------------------------------|
| UI | APIProxyDataPath | (有効期限を更新する)代理有効期限取得データのパス ファイルの拡張子は「.NRS」です。 |
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数 |
| Code | APITrialPeriod | 猶予(試用)日数 |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

「代理有効期限取得/準備」を実行します。(有効期限を更新したい PC で利用)

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。

代理有効期限取得準備(認証PC)【API系サンプルForm】

認証情報を外部データ(代理有効期限取得データ)に出力します。

① 代理で有効期限を取得するために認証済PCの情報を外部データ(代理有効期限取得データ)に保存します。保存先のフォルダを指定して「保存」ボタンを押してください。次のステップは、代理PCで「代理有効期限-取得」処理を行います。

フォルダ: 参照

保存 閉じる

◆処理手順

一連の処理の手順は次の通りです。

- [1] 代理有効期限取得データパスをエンドユーザに画面より入力させ、代理有効期限取得データパスプロパティ(APIProxyDataPath プロパティ)に設定します。この際、ファイルの拡張子は「.NRS」とします。
- [2] 上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。
★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。
- [3] 上記の「想定 UI 画面」の「保存」ボタンに連動させるなどして、当メソッド(APIPreparationOfProxyUpdateOfExpirationDate メソッド)を実行します。

APIProxyActivateRegisterExecute メソッド

【機能】

「代理認証登録/実行」を実行します。(代理 PC で利用)
 (代理認証機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function **APIProxyActivateRegisterExecute()** As **Boolean**

<C#>

public **bool** **APIProxyActivateRegisterExecute()**

<VC++>

public : **VARIANT_BOOL** **NewtonenRvcpp::IActivation::**

APIProxyActivateRegisterExecute()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 | |
|-----------|----------------------------------------|----------------------------------------------------------------------------------------------------------------------------|---------------|
| UI | APIProxyDataPath | (認証登録する)代理認証データのパス。 ファイルの拡張子は「.NRS」です。 | |
| UI | APICertificationID | (認証登録する)認証 ID(取得専用)。 代理認証登録データの取得メソッド (APIGetProxyDataForRegister メソッド) の実行により、代理認証データから取得した認証 ID が当プロパティに設定されます。 | |
| UI | APIProductID | (認証登録する)プロダクト ID | |
| UI | APISerialNo | (認証登録する)シリアル No. | |
| UI | APIUseProxyServer | プロキシサーバーの使用区分 | |
| UI | APIProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー使用時有効 |
| UI | APIProxyServerPort | プロキシサーバーのポート | |
| UI | APIProxyServerUserName | プロキシサーバーのユーザ名 | |
| UI | APIProxyServerPassword | プロキシサーバーのパスワード | |

| | | |
|------|----------------------------------------------------------|------------------------|
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数(0:レンタル期間なし) |
| Code | APITrialPeriod | 猶予(試用)日数(0:猶予期間なし) |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード |
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名 |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIWebServiceTimeout | Web サービスのタイムアウト |
| Code | APIWebServiceURL | Web サービスの URL |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 |

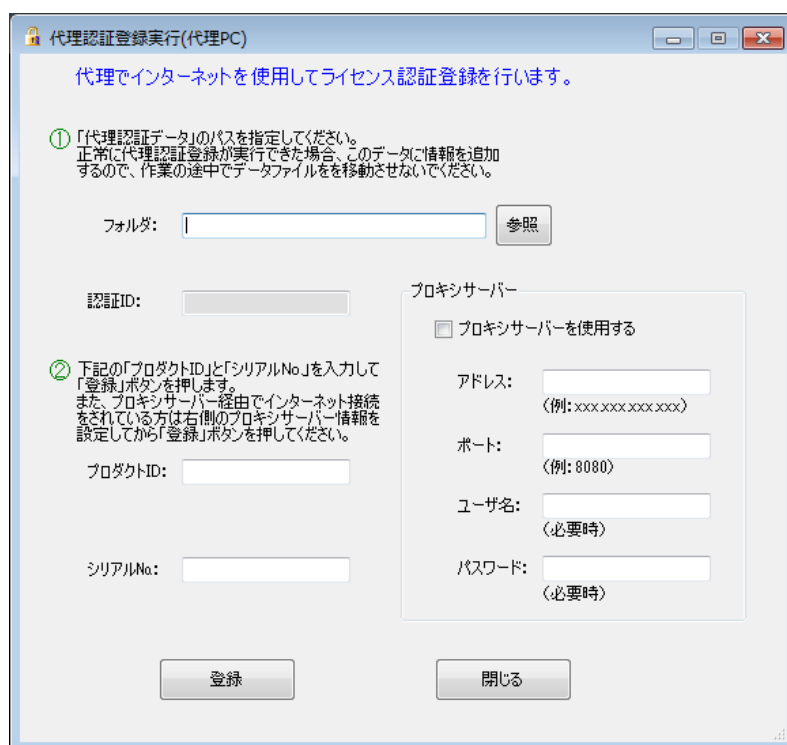
※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

「代理認証登録/実行」を実行します。(代理 PC で利用)

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。



この画面上的「認証 ID」は、代理認証登録データの取得メソッド(APIGetProxyDataForRegister メソッド)の実行で設定される認証 ID プロパティ(APICertificationID プロパティ)で取得できます。

◆処理手順

一連の処理の手順は次の通りです。

[1]代理認証データパスをエンドユーザに画面より入力させ、代理認証データパスプロパティ(APIProxyDataPath プロパティ)に設定します。この際、ファイルの拡張子は「.NRS」とします。

- [2]代理認証登録データの取得メソッド(APIGetProxyDataForRegister メソッド)を実行します。
これにより、認証IDプロパティ(APICertificationID プロパティ)に代理認証登録データから取得した認証IDが設定されます。
その内容を画面に表示します。この内容表示そのものは必須ではありません。
- [3]プロダクトID とシリアル No.をエンドユーザに画面より入力させ、プロダクトID プロパティ(APIProductID プロパティ)とシリアルNo. プロパティ(APISerialNo プロパティ)にそれぞれ設定します。あるいは、別の方法で結果としてプロダクトID プロパティ(APIProductID プロパティ)とシリアルNo. プロパティ(APISerialNo プロパティ)にそれぞれ認証登録するプロダクトID とシリアル No.を設定します。
- [4]上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。
★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。
- [5]上記の「想定 UI 画面」の「登録」ボタンに連動させるなどして、当メソッド(APIProxyActivateRegisterExecute メソッド)を実行します。

APIProxyActivateRegisterFix メソッド

【機能】

「代理認証登録/確定」を実行します。(認証登録したい PC で利用)
 (代理認証機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function **APIProxyActivateRegisterFix()** As **Boolean**

<C#>

public **bool** **APIProxyActivateRegisterFix()**

<VC++>

public : **VARIANT_BOOL** **NewtonenNRvcpp::IActivation:: APIProxyActivateRegisterFix()**

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|-----------|---------------------------------------|------------------------------------------------------------------------------------------------------------------------|
| UI | APIProxyDataPath | (認証登録する)代理認証データのパス ファイルの拡張子は「.NRS」です。 |
| UI | APICertificationID | (認証登録する)認証 ID(取得専用)。 代理認証登録データの取得メソッド (APIGetProxyDataForRegister メソッド)の実行により、代理認証データから取得した認証 ID が当プロパティに設定されます。 |
| UI | APIProductID | (認証登録する)プロダクト ID 代理認証登録データの取得メソッド (APIGetProxyDataForRegister メソッド)の実行により、代理認証データから取得したプロダクト ID が当プロパティに設定されます。 |
| UI | APISerialNo | (認証登録する)シリアル No. 代理認証登録データの取得メソッド (APIGetProxyDataForRegister メソッド)の実行により、代理認証データから取得したシリアル No.が当プロパティに設定されます。 |
| Code | APIEncryptionPassword | 暗号化時のパスワード |

| | | |
|------|-------------------------------------------------------|------------------------|
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数 (0:レンタル期間なし) |
| Code | APITrialPeriod | 猶予(試用)日数 (0:猶予期間なし) |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

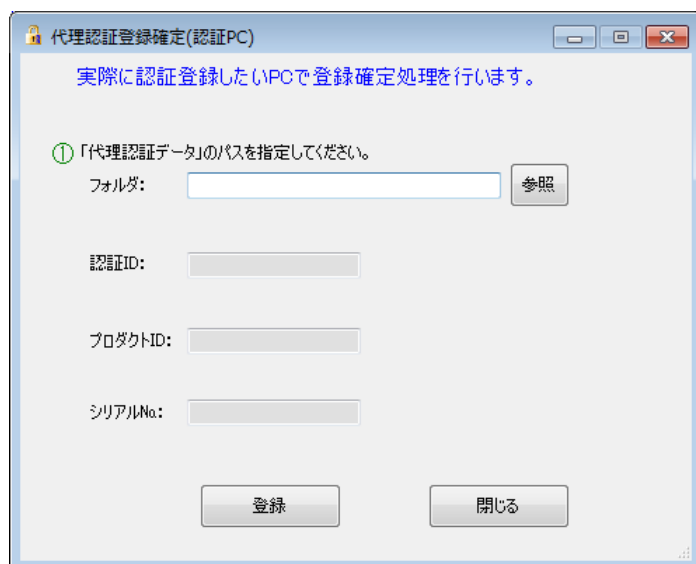
※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

「代理認証登録/確定」を実行します。(認証登録したい PC で利用)

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。



この画面上の「認証 ID」、「プロダクト ID」、「シリアル No.」は、代理認証登録データの取得メソッド (APIGetProxyDataForRegister メソッド) の実行で設定される認証 ID プロパティ (APICertificationID プロパティ)、プロダクト ID プロパティ (APIProductID プロパティ)、シリアル No. プロパティ (APISerialNo プロパティ) でそれぞれ取得できます。

◆処理手順

一連の処理の手順は次の通りです。

[1]代理認証データパスをエンドユーザに画面より入力させ、代理認証データパスプロパティ (APIProxyDataPath プロパティ) に設定します。この際、ファイルの拡張子は「.NRS」とします。

[2]代理認証登録データの取得メソッド (APIGetProxyDataForRegister メソッド) を実行します。これにより、認証 ID プロパティ (APICertificationID プロパティ)、プロダクト ID プロパティ (APIProductID プロパティ)、シリアル No. プロパティ (APISerialNo プロパティ) に代理認証データから取得した認証 ID、プロダクト ID、シリアル No.それぞれが設定されます。それらの内容を画面に表示します。この内容表示そのものは必須ではありません。

[3]上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。

★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。

[5] 上記の「想定 UI 画面」の「登録」ボタンに連動させるなどして、当メソッド (APIProxyActivateRegisterFix メソッド) を実行します。

APIProxyActivateRegisterPrepare メソッド

【機能】

「代理認証登録/準備」を実行します。(認証登録したい PC で利用)
 (代理認証機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function **APIProxyActivateRegisterPrepare()** As **Boolean**

<C#>

public **bool** **APIProxyActivateRegisterPrepare()**

<VC++>

public : **VARIANT_BOOL** **NewtonenRvcpp::IActivation::**

APIProxyActivateRegisterPrepare()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|-----------|-------------------------------------------------------|------------------------------------------|
| UI | APIProxyDataPath | (認証登録する)代理認証データのパス ファイルの拡張子は「.NRS」です。 |
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数 |
| Code | APITrialPeriod | 猶予(試用)日数 |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

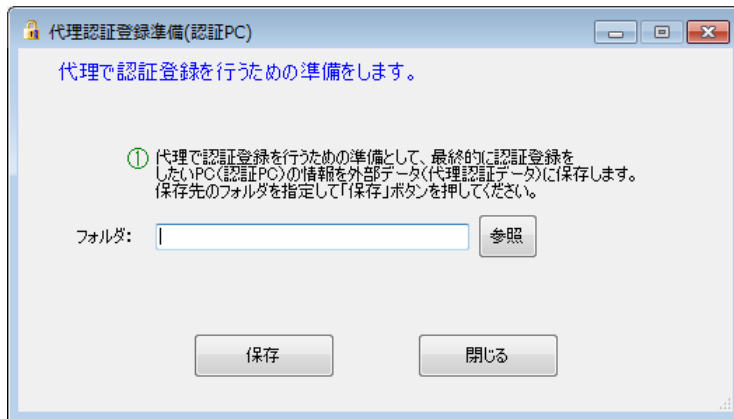
※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

「代理認証登録/準備」を実行します。(認証登録したい PC で利用)

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。



◆処理手順

一連の処理の手順は次の通りです。

[1] 代理認証データパスをエンドユーザに画面より入力させ、代理認証データパスプロパティ (APIProxyDataPath プロパティ)に設定します。この際、ファイルの拡張子は「.NRS」とします。

[2] 上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。

★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。

[3] 上記の「想定 UI 画面」の「保存」ボタンに連動させるなどして、当メソッド (APIProxyActivateRegisterPrepare メソッド)を実行します。

APIProxyActivateRemoveExecute メソッド

【機能】

「代理認証解除/実行」を実行します。(代理 PC で利用)
 (代理認証機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function **APIProxyActivateRemoveExecute()** As **Boolean**

<C#>

public **bool** **APIProxyActivateRemoveExecute()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::**

APIProxyActivateRemoveExecute()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|-----------|------------------------------------|---------------------------------------------------------------------------------------------------------------------------------|
| UI | APIProxyDataPath | (認証登録する)代理認証データのパス ファイルの拡張子は「.NRS」です。 |
| UI | APICertificationID | (認証登録する)認証 ID(取得専用)。 代理認証解除データの取得メソッド (APIGetProxyDataForRemove メソッド)の 実行により、代理認証データから取得した 認証 ID が当プロパティに設定されます。 |
| UI | APIProductID | (認証登録する)プロダクト ID 代理認証解除データの取得メソッド (APIGetProxyDataForRemove メソッド)の 実行により、代理認証データから取得した プロダクト ID が当プロパティに設定されま す。 |
| UI | APISerialNo | (認証登録する)シリアル No. 代理認証解除データの取得メソッド (APIGetProxyDataForRemove メソッド)の 実行により、代理認証データから取得した |

| | | |
|------|----------------------------------------------------------|-------------------------|
| | | シリアル No.が当プロパティに設定されます。 |
| UI | APIUseProxyServer | プロキシサーバーの使用区分 |
| UI | APIProxyServerAddress | プロキシサーバーのアドレス |
| UI | APIProxyServerPort | プロキシサーバーのポート |
| UI | APIProxyServerUserName | プロキシサーバーのユーザ名 |
| UI | APIProxyServerPassword | プロキシサーバーのパスワード |
| | | プロキシサーバー使用時有効 |
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数 |
| Code | APITrialPeriod | 猶予(試用)日数 |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード |
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名 |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIWebServiceTimeout | Web サービスのタイムアウト |
| Code | APIWebServiceURL | Web サービスの URL |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 |

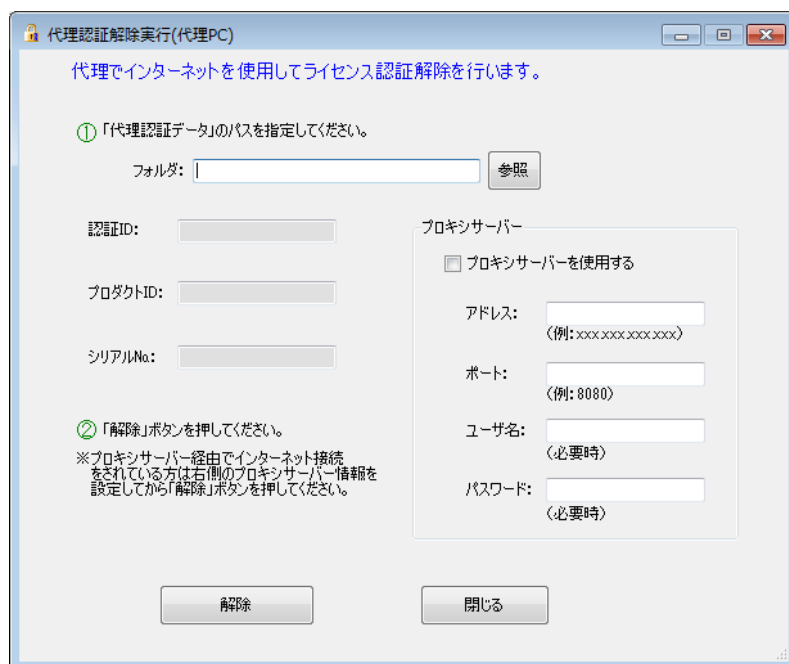
※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

「代理認証解除/実行」を実行します。(代理 PC で利用)

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。



この画面上の「認証 ID」、「プロダクト ID」、「シリアル No.」は、代理認証解除データの取得メソッド (APIGetProxyDataForRemove メソッド) の実行で設定される認証 ID プロパティ (APICertificationID プロパティ)、プロダクト ID プロパティ (APIProductID プロパティ)、シリアル No. プロパティ (APISerialNo プロパティ) でそれぞれ取得できます。

◆処理手順

一連の処理の手順は次の通りです。

[1]代理認証データパスをエンドユーザに画面より入力させ、代理認証データパスプロパティ (APIProxyDataPath プロパティ) に設定します。この際、ファイルの拡張子は「.NRS」とします。

[2]代理認証解除データの取得メソッド (APIGetProxyDataForRemove メソッド) を実行します。
これにより、認証 ID プロパティ (APICertificationID プロパティ)、プロダクト ID プロパティ (APIProductID プロパティ)、シリアル No. プロパティ (APISerialNo プロパティ) に代理認証データから取得した認証 ID、プロダクト ID、シリアル No.それぞれが設定されます。
それらの内容を画面に表示します。この内容表示そのものは必須ではありません。

[3]上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。

★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。

[4]上記の「想定 UI 画面」の「解除」ボタンに連動させるなどして、当メソッド (APIProxyActivateRemoveExecute メソッド) を実行します。

APIProxyActivateRemovePrepare メソッド

【機能】

「代理認証解除/準備」を実行します。(認証登録したい PC で利用)
(代理認証機能については、「[代理認証機能について](#)」を参照)

【構文】

<VB2010/VB6.0>

Public Function **APIProxyActivateRemovePrepare()** As **Boolean**

<C#>

public **bool** **APIProxyActivateRemovePrepare()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::**

APIProxyActivateRemovePrepare()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|-----------|-------------------------------------------------------|------------------------------------------|
| UI | APIProxyDataPath | (認証登録する)代理認証データのパス ファイルの拡張子は「.NRS」です。 |
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIRentalPeriod | レンタル日数 |
| Code | APITrialPeriod | 猶予(試用)日数 |
| Code | APIUseCpuInfo | CPU 情報の使用区分 |
| Code | APIUseMacAddress | MAC アドレスの使用区分 |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

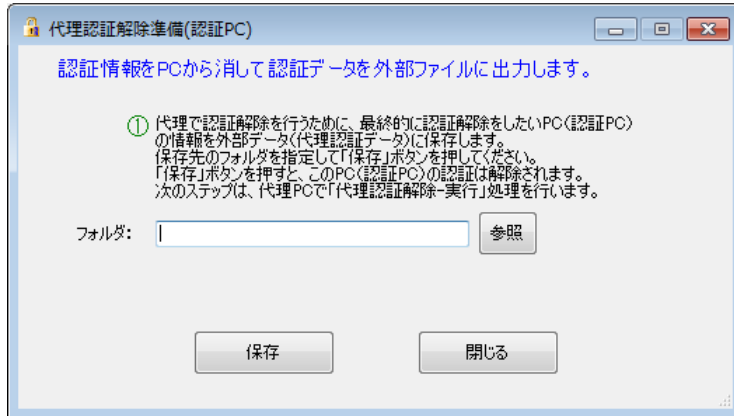
※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

「代理認証解除/準備」を実行します。(認証登録したい PC で利用)

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。



◆処理手順

一連の処理の手順は次の通りです。

[1] 代理認証データパスをエンドユーザに画面より入力させ、代理認証データパスプロパティ (APIProxyDataPath プロパティ)に設定します。この際、ファイルの拡張子は「.NRS」とします。

[2] 上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。

★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。

[3] 上記の「想定 UI 画面」の「保存」ボタンに連動させるなどして、当メソッド (APIProxyActivateRemovePrepare メソッド)を実行します。

APIReadProxyServerInfoFromRegistry メソッド

【機能】

レジストリからプロキシサーバーの情報を取得します。

【構文】

<VB2010/VB6.0>

Public Function **APIReadProxyServerInfoFromRegistry** () As **Boolean**

<C#>

public **bool** **APIReadProxyServerInfoFromRegistry** ()

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::**

APIReadProxyServerInfoFromRegistry ()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|-----------|-------------------------------------------------------|------------------------|
| Code | APIEncryptionPassword | 暗号化時のパスワード |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

レジストリからプロキシサーバーの情報を取得します。

[1]このメソッドの戻り値は成功か失敗(True/False)。

[2] プロキシサーバーのユーザ名とパスワードを暗号化されおり復号化が失敗すると、当メソッドは失敗します。また、レジストリの読み込みで何らかの原因で失敗すると、当メソッドは失敗します。

[3]このメソッドの成功時のみ、次のプロパティにプロキシサーバーの各情報が設定されます。

- ・プロキシサーバーのアドレス(APIProxyServerAddress)
- ・プロキシサーバーのパスワード(APIProxyServerPassword)

- ・プロキシサーバーのポート (APIProxyServerPort)
- ・プロキシサーバーのユーザ名 (APIProxyServerUserName)
- ・プロキシサーバーの使用区分 (APIUseProxyServer)

[4]これらのプロパティは、正常に読み込みが成功しない限り、初期値(空文字列)が設定されます。

APIRestoreCancelStatus メソッド

【機能】

「認証解除状態回復」を実行します。

【構文】

<VB2010/VB6.0>

Public Function APIRestoreCancelStatus() As Boolean

<C#>

public bool APIRestoreCancelStatus()

<VC++>

public : VARIANT_BOOL NewtonenNRvcpp::IActivation:: APIRestoreCancelStatus()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分※ | プロパティ | 機能 |
|-------|------------------------------------|------------------------------------------------------------------------------------------------------------------------------|
| UI | APICertificationID | (登録解除する)認証 ID (取得専用)。 (レジストリからの)認証登録済み情報の取得メソッド (APIGetRegisteredInfoFromRegistry メソッド) の実行により、登録済みの認証 ID が当プロパティに設定されます。 |
| UI | APIProductID | (認証解除する)プロダクト ID (レジストリからの)認証登録済み情報の取得メソッド (APIGetRegisteredInfoFromRegistry メソッド) の実行により、登録済みのプロダクト ID が当プロパティに設定されます。 |
| UI | APISerialNo | (認証解除する)シリアル No。 (レジストリからの)認証登録済み情報の取得メソッド (APIGetRegisteredInfoFromRegistry メソッド) の実行により、登録済みのシリアル No. が当プロパティに設定されます。 |
| UI | APIUseProxyServer | プロキシサーバーの使用区分 |

| | | | |
|------|----------------------------------------------------------|------------------------|---------------|
| UI | APIProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー使用時有効 |
| UI | APIProxyServerPort | プロキシサーバーのポート | |
| UI | APIProxyServerUserName | プロキシサーバーのユーザ名 | |
| UI | APIProxyServerPassword | プロキシサーバーのパスワード | |
| Code | APIEncryptionPassword | 暗号化時のパスワード | |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 | |
| Code | APIRentalPeriod | レンタル日数 | |
| Code | APITrialPeriod | 猶予(試用)日数 | |
| Code | APIUseCpuInfo | CPU 情報の使用区分 | |
| Code | APIUseMacAddress | MAC アドレスの使用区分 | |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス | |
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード | |
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名 | |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード | |
| Code | APIWebServiceTimeout | Web サービスのタイムアウト | |
| Code | APIWebServiceURL | Web サービスの URL | |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 | |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

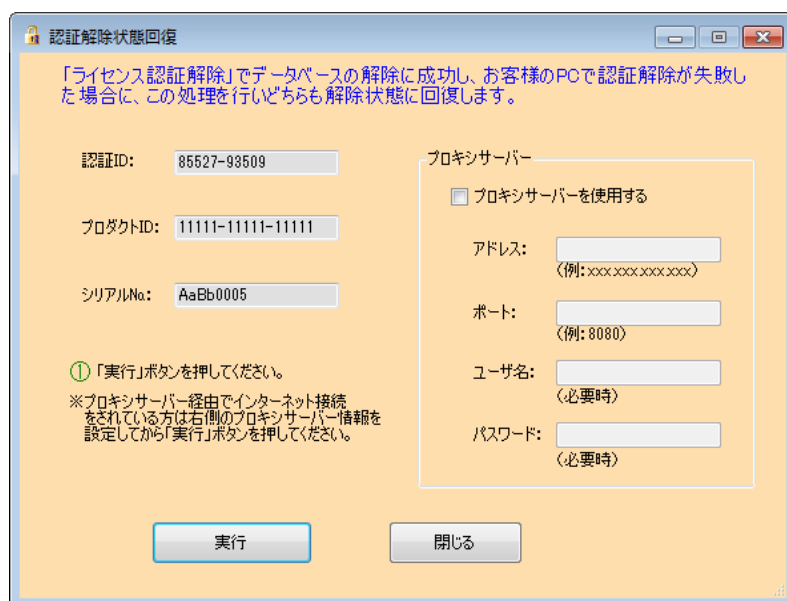
【解説】

「認証解除状態回復」を実行します。

「ライセンス認証解除」でデータベースの解除に成功したが、エンドユーザ PC での認証解除が失敗した状態になった場合に、このメソッドによる処理でエンドユーザの操作でその状態を回復しエンドユーザ PC を認証解除状態とします。

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。



この画面上の「認証 ID」、「プロダクト ID」、「シリアル No.」は、(レジストリからの)認証登録済み情報の取得メソッド(APIGetRegisteredInfoFromRegistry メソッド)の実行で設定される認証 ID プロパティ

(APICertificationID プロパティ)、プロダクト ID プロパティ(APIProductID プロパティ)、シリアル No. プロパティ(APISerialNo プロパティ)でそれぞれ取得できます。

◆処理手順

一連の処理の手順は次の通りです。

[1](レジストリからの)認証登録済み情報の取得メソッド(APIGetRegisteredInfoFromRegistry メソッド)を実行します。これにより、認証 ID プロパティ、プロダクト ID プロパティ、シリアル No.プロパティに各値が設定されます。

[2]認証 ID プロパティ、プロダクト ID プロパティ、シリアル No.プロパティの各値を画面に表示します。

[3]上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。

★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。

[4] 上記の「想定 UI 画面」の「実行」ボタンに連動させるなどして、当メソッド(APIRestoreCancelStatus メソッド)を実行します。

APIRestoreRegisterStatus メソッド

【機能】

「認証登録状態回復」を実行します。

【構文】

<VB2010/VB6.0>

Public Function **APIRestoreRegisterStatus**() As **Boolean**

<C#>

public **bool** **APIRestoreRegisterStatus**()

<VC++>

public : **VARIANT_BOOL** **NewtonenRvcpp::IActivation:: APIRestoreRegisterStatus**()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分※ | プロパティ | 設定する内容 | |
|-------|----------------------------------------------------------|------------------------|---------------|
| UI | APIProductID | (認証登録する)プロダクト ID | |
| UI | APISerialNo | (認証登録する)シリアル No. | |
| UI | APIUseProxyServer | プロキシサーバーの使用区分 | |
| UI | APIProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー使用時有効 |
| UI | APIProxyServerPort | プロキシサーバーのポート | |
| UI | APIProxyServerUserName | プロキシサーバーのユーザ名 | |
| UI | APIProxyServerPassword | プロキシサーバーのパスワード | |
| Code | APIEncryptionPassword | 暗号化時のパスワード | |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 | |
| Code | APIRentalPeriod | レンタル日数 | |
| Code | APITrialPeriod | 猶予(試用)日数 | |
| Code | APIUseCpuInfo | CPU 情報の使用区分 | |
| Code | APIUseMacAddress | MAC アドレスの使用区分 | |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス | |
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード | |
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名 | |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード | |

| | | |
|------|-----------------------------------------------------|---------------------|
| Code | APIWebServiceTimeout | Web サービスのタイムアウト |
| Code | APIWebServiceURL | Web サービスの URL |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

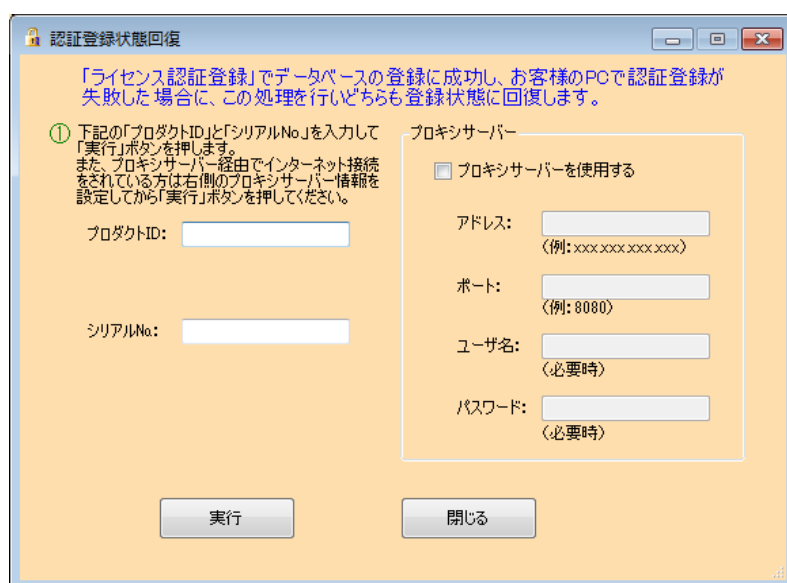
【解説】

「認証登録状態回復」を実行します。

「ライセンス認証登録」でデータベースの登録に成功したが、エンドユーザ PC での認証登録が失敗した状態になった場合に、このメソッドによる処理でエンドユーザの操作でその状態を回復しエンドユーザ PC を認証登録状態とします。

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。



◆処理手順

一連の処理の手順は次の通りです。

[1] プロダクト ID とシリアル No. をエンドユーザに画面より入力させ、プロダクト ID プロパティ (APIProductID プロパティ) とシリアル No. プロパティ (APISerialNo プロパティ) にそれぞれ設定します。あるいは、別の方法で結果としてプロダクト ID プロパティ (APIProductID プロパティ) とシリアル No. プロパティ (APISerialNo プロパティ) にそれぞれ認証登録するプロダクト ID とシリアル No. を設定します。

[2] 上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。

★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。

[3] 上記の「想定 UI 画面」の「登録」ボタンに連動させるなどして、当メソッド (APIRestoreRegisterStatus メソッド) を実行します。

APIRunNR2AppDateRemove メソッド

【機能】

「アプリ起動日」を削除します。

【構文】

<VB2010/VB6.0>

Public Function **APIRunNR2AppDateRemove()** As **Boolean**

<C#>

public **bool** **APIRunNR2AppDateRemove()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation:: APIRunNR2AppDateRemove()**

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|-----------|-------------------------------------------------------|------------------------|
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

「アプリ起動日」を削除します。

認証レスキュー！2 を含む貴社のアプリを起動した際に、レジストリに記録されている前回の「アプリ起動日」を確認します。

(もし、「アプリ起動日」が存在しなかった場合は、起動した際にレジストリに記録します。)

その「アプリ起動日」より PC の日付が古い場合、故意に PC の日付が変更されたということで「PC の日付が変更されました。(APIErrorStatus=174)」というメッセージが表示されます。

例:

たとえば、本日を 6 月 1 日とします。PC の日付を 6 月 2 日に設定しアプリを起動します。

次に、PC の日付を本日(つまり 6 月 1 日)に戻しても、「アプリ起動日」は 6 月 2 日で記録されており、PC の日付が古いので「PC の日付が変更されました。」というメッセージが表示されます。

この状態になった場合、PC の日付を 6 月 2 日に設定しない限り、認証登録等が実行できません。

このメソッドを実行後は、再度 PC の日付を本日(つまり 6 月 1 日)に設定して、アプリをご利用いただける状態にします。

APIRunNR2AppDateRemove2 メソッド

【機能】

「アプリ起動日」を削除します。

【構文】

<VB2010/VB6.0>

Public Function **APIRunNR2AppDateRemove2()** As **Boolean**

<C#>

public **bool** **APIRunNR2AppDateRemove2()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::**

APIRunNR2AppDateRemove2()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 |
|-----------|-------------------------------------------------------|------------------------------|
| Code | APISelectRunAppDatePathFlag | 「アプリ起動日」の読み込み/書き込み場所の切り替えフラグ |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

以下の機能以外は従来の **APIRunNR2AppDateRemove** メソッドと同等です。

APISelectRunAppDatePathFlag プロパティ値による場所+従来の

APIVendorsProductStartRegistryKeyPath プロパティ値の場所の「アプリ起動日」を削除します。

このメソッドは、ユーザー様からのご要望で追加いたしました。

[APIActivateStatusCheck2](#) メソッドで書き込まれた「アプリ起動日」を削除します。

| |
|------------------------------|
| APITrialStartDateRemove メソッド |
|------------------------------|

【機能】

「猶予日数」の「開始日」を削除します。

【構文】

<VB2010/VB6.0>

Public Function **APITrialStartDateRemove()** As **Boolean**

<C#>

public **bool** **APITrialStartDateRemove()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation:: APITrialStartDateRemove()**

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分※ | プロパティ | 設定する内容 |
|-------|-------------------------------------------------------|------------------------|
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

「猶予日数」の「開始日」を削除します。

このメソッドを実行後は、エンドユーザは再度「猶予日数」分の利用が可能になります。

APIUpdateOfExpirationDate メソッド

【機能】

「有効期限の更新」を実行します。

【構文】

<VB2010/VB6.0>

Public Function **APIUpdateOfExpirationDate**() As **Boolean**

<C#>

public **bool** **APIUpdateOfExpirationDate**()

<VC++>

public : **VARIANT_BOOL** **NewtonenRvcpp::IActivation:: APIUpdateOfExpirationDate**()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分※ | プロパティ | 機能 |
|-------|------------------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| UI | APIProductID | (有効期限を更新する)プロダクト ID (レジストリからの)認証登録済み情報の取得メソッド (APIGetRegisteredInfoFromRegistryメソッド)の実行により、登録済みのプロダクト ID が当プロパティに設定されます。 |
| UI | APISerialNo | (有効期限を更新する)シリアル No. (レジストリからの)認証登録済み情報の取得メソッド (APIGetRegisteredInfoFromRegistryメソッド)の実行により、登録済みのシリアル No.が当プロパティに設定されます。 |
| UI | APICurrentExpirationDate | (有効期限を更新する)現在の有効期限 (レジストリからの)認証登録済み情報の取得メソッド (APIGetRegisteredInfoFromRegistry |

| | | | |
|------|----------------------------------------------------------|-------------------------------------------------------------------------------------------------------|---------------|
| | | メソッド)の実行により、登録済みの現在の有効期限が当プロパティに設定されます。 | |
| UI | APIUseProxyServer | プロキシサーバーの使用区分 | |
| UI | APIProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー使用時有効 |
| UI | APIProxyServerPort | プロキシサーバーのポート | |
| UI | APIProxyServerUserName | プロキシサーバーのユーザ名 | |
| UI | APIProxyServerPassword | プロキシサーバーのパスワード | |
| Code | APIEncryptionPassword | 暗号化時のパスワード | |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 | |
| Code | APIOverwriteModeOfExpirationDateUpdate | 新しい有効期限が現在と同じか古い場合の対応を設定します。 True: 新しい有効期限が現在と同じか古い場合でも更新します。 False: 新しい有効期限が現在と同じか古い場合は更新しません。 | |
| Code | APIRentalPeriod | レンタル日数 | |
| Code | APITrialPeriod | 猶予(試用)日数 | |
| Code | APIUseCpuInfo | CPU 情報の使用区分 | |
| Code | APIUseMacAddress | MAC アドレスの使用区分 | |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス | |
| Code | APIWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード | |
| Code | APIWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名 | |
| Code | APIWebServiceCheckPassword | Web サービス確認パスワード | |
| Code | APIWebServiceTimeout | Web サービスのタイムアウト | |
| Code | APIWebServiceURL | Web サービスの URL | |
| Code | APIWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 | |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

「有効期限の更新」を実行します。

プロダクト ID とシリアル No.が有効期限によるライセンスの場合、当メソッドによる処理を利用してエンドユーザに有効期限を更新させることができます。

この処理をエンドユーザに実行してもらう前に、貴社で新しい有効期限の設定をする必要があります。

有効期限の設定は認証管理システムの「認証キー編集(表形式)」処理を利用します。

◆想定 UI 画面

当メソッドの利用時に、想定する貴社作成の UI(ユーザインターフェース)画面は次の通りです。

この画面の「認証 ID」、「プロダクト ID」、「シリアル No.」は、(レジストリからの)認証登録済み情報の取得メソッド(APIGetRegisteredInfoFromRegistry メソッド)の実行で設定される認証 ID プロパティ(APICertificationID プロパティ)、プロダクト ID プロパティ(APIProductID プロパティ)、シリアル No. プロパティ(APISerialNo プロパティ)でそれぞれ取得できます。

この画面の「現在の有効期限」は、(レジストリからの)認証登録済み情報の取得メソッド(APIGetRegisteredInfoFromRegistry メソッド)の実行で設定される現在の有効期限プロパティ(APICurrentExpirationDate プロパティ)で取得できます。

◆処理手順

一連の処理の手順は次の通りです。

[1](レジストリからの)認証登録済み情報の取得メソッド(APIGetRegisteredInfoFromRegistry メソッド)を実行します。

これにより、プロダクト ID プロパティ、シリアル No.プロパティ、現在の有効期限プロパティに各値が設定されます。

[2]プロダクト ID プロパティ、シリアル No.プロパティ、現在の有効期限プロパティの各値を画面に表示します。

[3]上記の【必須設定プロパティ】一覧にある「設定区分」が「Code」のプロパティに適切な値を設定します。

★これらのプロパティは、一度設定した内容で変更がなければ再設定する必要はありません。

[4] 上記の「想定 UI 画面」の「更新」ボタンに連動させるなどして、当メソッド(APIUpdateOfExpirationDate メソッド)を実行します。

[5]上記の「想定 UI 画面」の「新しい有効期限」に更新後の有効期限を表示します。更新後の有効期限は、処理手順の[1]を再度行うことで取得できます。

なお、エンドユーザに有効期限を更新させる方法として当処理を実行させる他に、エンドユーザに一度認証の解除後、再度認証の登録をしてもらうことでも更新が完了します。

エンドユーザが代理認証を利用している場合で、有効期限によるライセンスを更新する場合はその方法を使います。

APIWriteProxyServerInfoToRegistry メソッド

【機能】

レジストリへプロキシサーバーの情報を書き込みます。

【構文】

<VB2010/VB6.0>

Public Function **APIWriteProxyServerInfoToRegistry()** As **Boolean**

<C#>

public **bool** **APIWriteProxyServerInfoToRegistry()**

<VC++>

public : **VARIANT_BOOL** **NewtoneNRvcpp::IActivation::**

APIWriteProxyServerInfoToRegistry()

【引数】

なし

【戻り値】

True: 正常

False: エラー(未設定や範囲を超えているプロパティがある)

戻り値とは別に、[APIErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| 設定区分 ※ | プロパティ | 設定する内容 | |
|-----------|-------------------------------------------------------|------------------------|-----------------|
| UI | APIUseProxyServer | プロキシサーバーの使用区分 | |
| UI | APIProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー 使用時 |
| UI | APIProxyServerPort | プロキシサーバーのポート | |
| UI | APIProxyServerUserName | プロキシサーバーのユーザー名 | |
| UI | APIProxyServerPassword | プロキシサーバーのパスワード | |
| Code | APIEncryptionPassword | 暗号化時のパスワード | |
| Code | APIEncryptionSaltString | 暗号化時の Salt 文字列 | |
| Code | APIVendorsProductStartRegistryKeyPath | ベンダアプリケーション開始レジストリキーパス | |

※設定区分: UI⇒主に「想定 UI 画面」から設定されるプロパティ/Code⇒主にプログラム(コード)で設定されるプロパティ

【解説】

レジストリへプロキシサーバーの情報を書き込みます。

[1]このメソッドの戻り値は成功か失敗(True/False)。

[2]プロキシサーバーのユーザ名とパスワードを暗号化しており暗号化が失敗すると、当メソッドは失敗します。また、レジストリの書き込みで何らかの原因で失敗すると、当メソッドは失敗します。

<ASP.NET 系プロパティ>

プロパティ一覧

| プロパティ | 機能 |
|-----------------------------------------------------------|----------------------------------------------|
| APIxError 列挙体 | エラー内容を表示します。 |
| APIxErrorStatus | ASP.NET 系のメソッドを使用した際のエラーの内容を返す。(取得専用) |
| APIxExpirationDate | 有効期限を設定します。 |
| APIxExternalLinkKey | 外部データベースとの「リンク用キー」を設定します。 |
| APIxFreeItem1～5 | 自由入力項目 1～5 を設定します。 |
| APIxKindOfRandom | シリアル No.をランダムに自動作成する場合の文字種別を設定します。 |
| APIxLicenseCount | ライセンス数を設定します。 |
| APIxNumberingCount | シリアル No.を作成する数を設定します。 |
| APIxProductID | プロダクト ID を設定します。 |
| APIxProxyServerAddress | プロキシサーバーのアドレスを設定します。 |
| APIxProxyServerPassword | プロキシサーバーのパスワードを設定します。 |
| APIxProxyServerPort | プロキシサーバーのポートを設定します。 |
| APIxProxyServerUserName | プロキシサーバーのユーザ名を設定します。 |
| APIxSerialNo | シリアル No.を設定・取得します。 |
| APIxSerialNoString | 認証キー情報作成直後のシリアル No.を取得します。(取得専用) |
| APIxStartNo | シリアル No.を自動的にナンバリングして作成する場合の開始番号を設定します。 |
| APIxStartFixedString | 作成するシリアル No.の上位固定文字列を設定します。 |
| APIxStepNo | シリアル No.を自動的にナンバリングして作成する場合の間隔(ステップ)数を設定します。 |
| APIxUseProxyServer | プロキシサーバーの使用区分(デフォルト:False)を設定します。 |
| APIxUseRental | レンタル機能の使用(False:利用しない、True:利用する)を設定します。 |
| APIxUseExpirationDate | 有効期限利用の有無(False:利用しない、True:利用する)を設定します。 |
| APIxWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワードを設定します。 |
| APIxWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザ名を設定します。 |
| APIxWebServiceCheckPassword | Web サービス確認パスワード(8 文字以上)を設定します。 |
| APIxWebServiceTimeout | Web サービスのタイムアウト(デフォルト:60 秒)を設定します。 |
| APIxWebServiceURL | Web サービスの URL を設定します。 |
| APIxWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用(デフォルト:False)を設定します。 |

APIxError 列挙体

エラー内容を表します。

```
public enum APIxError
パブリックメンバ
```

| メンバ名 | 値 | 内容 | 関連するメソッド |
|----------------------------------|-----|-------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| None | 0 | エラーなし | |
| OutOfMemoryException | 11 | プログラムの実行を継続するためのメモリが不足している場合にスローされる例外。 | |
| StackOverflowException | 12 | 入れ子になったメソッド呼び出しが多くなりすぎ、実行スタックがオーバーフローした場合にスローされる例外。このクラスは継承できません。 | |
| UnauthorizedAccessException | 13 | オペレーティング システムが I/O エラーまたは特定の種類のセキュリティエラーのためにアクセスを拒否する場合、スローされる例外。 | |
| IoDirectoryNotFoundExce ption | 14 | ファイルまたはディレクトリの一部が見つからない場合にスローされる例外。 | |
| IoDriveNotFoundExce ption | 15 | 使用できないドライブまたは共有にアクセスしようとするときにスローされる例外。 | |
| IoEndOfStreamException | 16 | ストリームの末尾を越えて読み取ろうとしたときにスローされる例外。 | |
| IoFileLoadException | 17 | マネージ アセンブリが見つかったが、読み込むことができない場合にスローされる例外。 | |
| IoFileNotFoundExce ption | 18 | ディスク上に存在しないファイルにアクセスしようとして失敗したときにスローされる例外。 | |
| IoIOException | 19 | I/O エラーが発生したときにスローされる例外。 | |
| IoPathTooLongException | 20 | パス名またはファイル名がシステム定義の最大長を超えている場合にスローされる例外。 | |
| BadStrInProductID | 101 | プロダクト ID に不正な文字が含まれています。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxDeleteActivationKey APIxEditOfExpirationDate |
| BadStrInSerialNo | 102 | シリアル No.に不正な文字が含まれています。 | APIxCreateOneActivationKey APIxDeleteActiva |

| | | | |
|-------------------|-----|------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | tionKey |
| FailedEnableNIC | 104 | NIC の設定に失敗しました。(有効化) | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxDeleteActivationKey APIxEditOfExpirationDate |
| FailedDisableNIC | 105 | NIC の設定に失敗しました。(無効化) | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxDeleteActivationKey APIxEditOfExpirationDate |
| NoMACAddress | 106 | MAC アドレスが 1 つも取得できませんでした。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxDeleteActivationKey APIxEditOfExpirationDate |
| AlreadyRegistered | 111 | 既に登録済みのプロダクト ID とシリアルNo.の組み合わせがあったため登録を中止しました。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| BadCheckPassword | 112 | 確認パスワードが不正です。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxDeleteActivationKey |

| | | | |
|----------------------------|-----|--------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | APIxDeleteActivationKey APIxEditOfExpirationDate |
| BadWaiFile | 113 | WebServEnv.wai ファイルに問題があります。 認証 Web サービス実行 PC 上で「認証レスキュー！2 Web 環境設定」(WebAdmin.exe)が行われていない可能性があります。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxDeleteActivationKey APIxEditOfExpirationDate |
| ExpirationDateIsOutOfRange | 116 | 有効期限が不正です。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxCreateRandomActivationKey APIxEditOfExpirationDate |
| ExpirationDateIsOldDate | 117 | 入力された「有効期限」が無効(前日以前)です。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxCreateRandomActivationKey APIxEditOfExpirationDate |
| ProductIDIsEmpty | 156 | プロダクトIDが入力されていません。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxEditOfExpirationDate |
| SerialNoIsEmpty | 157 | シリアルNo.が入力されていません。 | APIxCreateOneActivationKey APIxEditOfExpirationDate |

| | | | |
|--------------------------|-----|------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| CanNotUseTheString | 164 | 使用できない文字列が含まれています。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxDeleteActivationKey APIxEditOfExpirationDate |
| PropertyValueNotFound | 165 | 「必須設定プロパティ」に値が設定されていません。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxDeleteActivationKey APIxEditOfExpirationDate |
| DataBaseNotFound | 170 | データベースが存在しません。データベースは「Webインストーラ」を起動し「データベースのインストール」を実行すると作成されます。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxDeleteActivationKey APIxEditOfExpirationDate |
| CanNotConnectToTheServer | 171 | 何らかの理由でサーバーに接続できませんでした。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxDeleteActivationKey APIxEditOfExpirationDate |
| DataTableNotFound | 172 | データテーブルが存在しません。Web 環境設定で「データテーブル新規作成」の処理を行ってから、再度、この処理を行ってください。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandom |

| | | | |
|---------------------------------|-----|---------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| | | | mActivationKey APIxDeleteActivationKey APIxEditOfExpirationDate |
| BadStrInFixedSerialNo | 173 | シリアルNo.の上位固定文字列に不正な文字が含まれています。 | APIxCreateNumberingActivationKey APIxCreateRandomActivationKey |
| InvalidStartNo | 174 | 開始番号は0～99999の範囲で設定してください。 | APIxCreateNumberingActivationKey |
| InvalidStepNo | 175 | ステップ数は1～99999の範囲で設定してください。 | APIxCreateNumberingActivationKey |
| InvalidNumberingNo | 176 | ナンバリング数は1～1000の範囲で設定してください。 | APIxCreateNumberingActivationKey APIxCreateRandomActivationKey |
| InvalidLicenseNo | 177 | ライセンス数は1～100の範囲で設定してください。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| CanNotMultiLicense | 178 | ライセンス数に2以上のマルチライセンスが入力されましたが、「レンタル機能を使用する」が有効になっているためシングルライセンス以外は入力できません。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| SerialNoDigitsTooMany | 179 | 上位固定文字列と番号の桁数を合わせると指定したシリアル No.の桁数より多すぎます。 | APIxCreateNumberingActivationKey APIxCreateRandomActivationKey |
| CanNotEnterBlanksInTheProductID | 182 | プロダクトIDには空白は入力できません。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxEditOfExpirationDate |
| ProductIDDigitsNotEnough | 183 | 入力されたプロダクト ID の桁数が足り | APIxCreateNumb |

| | | | |
|--------------------------------|-----|------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| h | | ません。 | eringActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxEditOfExpirationDate |
| AnErrorOccurred | 184 | 何らかのエラーが発生しました。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxDeleteActivationKey APIxDeleteActivationKey APIxEditOfExpirationDate |
| ProductIDAndSerialNoNotFound | 185 | そのプロダクトIDとシリアルNo.の組み合わせは登録されていません。 | APIxDeleteActivationKey APIxEditOfExpirationDate |
| SerialNoDigitsNotEnough | 186 | 入力されたシリアル No.の桁数が足りません。 | APIxCreateOneActivationKey APIxEditOfExpirationDate |
| CanNotEnterBlanksInTheSerialNo | 187 | シリアル No.には空白は入力できません。 | APIxCreateOneActivationKey APIxEditOfExpirationDate |
| BadStrInExternalLinkKey | 188 | リンク用キーに不正な文字が含まれています。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| BadStrInFreeItem1 | 189 | 自由入力項目 1 に不正な文字が含まれています。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| BadStrInFreeItem2 | 190 | 自由入力項目 2 に不正な文字が含まれています。 | APIxCreateNumberingActivationKey |

| | | | |
|---------------------------|-----|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| | | | APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| BadStrInFreeItem3 | 191 | 自由入力項目 3 に不正な文字が含まれています。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| BadStrInFreeItem4 | 192 | 自由入力項目 4 に不正な文字が含まれています。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| BadStrInFreeItem5 | 193 | 自由入力項目 5 に不正な文字が含まれています。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| BadKindOfRandomOfSerialNo | 194 | シリアルNo.をランダムに自動作成するための文字種別が不正です。 | APIxCreateRandomActivationKey |
| BadExpirationDate | 195 | 有効期限が不正です。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxEditOfExpirationDate |
| ProductIDsTooLong | 196 | 入力されたプロダクト ID の桁数が多いです。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxEditOfExpirationDate |
| SerialNoIsTooLong | 197 | 入力されたシリアル No.の桁数が多いです。 | APIxCreateOneActivationKey APIxEditOfExpirationDate |
| BadFormatExpirationDate | 198 | 有効期限の書式が不正です。 | APIxCreateNumber |

| | | | |
|--------------------------|-----|-------------------------|---------------------------------------------------------------------------------------------------------------|
| | | | eringActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey APIxEditOfExpirationDate |
| ExternalLinkKeyIsTooLong | 199 | 入力されたリンク用キーの桁数が多いです。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| APIxFreeItem1IsTooLong | 200 | 入力された自由入力項目 1 の桁数が多いです。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| APIxFreeItem2IsTooLong | 201 | 入力された自由入力項目 2 の桁数が多いです。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| APIxFreeItem3IsTooLong | 202 | 入力された自由入力項目 3 の桁数が多いです。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| APIxFreeItem4IsTooLong | 203 | 入力された自由入力項目 4 の桁数が多いです。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |
| APIxFreeItem5IsTooLong | 204 | 入力された自由入力項目 5 の桁数が多いです。 | APIxCreateNumberingActivationKey APIxCreateOneActivationKey APIxCreateRandomActivationKey |

APIxErrorStatus プロパティ**【機能】**

ASP.NET 系のメソッドを使用した際のエラーの内容を返します。

【構文】

<VB2010>

Public ReadOnly Property **APIxErrorStatus** As **Integer**

<C#>

```
public int APIxErrorStatus { get; }
```

【解説】

ASP.NET 系のメソッドは戻り値として正常 (True) かエラー (False) を返しますが、この APIxErrorStatus プロパティはそのエラーの内容を [APIxError 列挙体](#) の参照で返します。この APIxErrorStatus プロパティは取得専用です。

APIExpirationDate プロパティ**【機能】**

有効期限を設定します。

【構文】

<VB2010>

Public Property **APIExpirationDate** As **String**

<C#>

```
public string APIExpirationDate { set; get; }
```

【解説】

有効期限を設定します。

有効期限は、“yyyy/mm/dd”形式で設定してください。

(例)西暦 2020 年 10 月 1 日を設定する場合

```
<VB> Class1.myAPIActivate.APIExpirationDate = "2020/10/01"
```

```
<C#> Class1.myAPIActivate.APIExpirationDate = "2020/10/01";
```

```
<VC++> theApp.m_pAPIActivation->APIExpirationDate = "2020/10/01";
```

次のメソッドの実行時に当プロパティ(APIExpirationDate プロパティ)に有効期限を設定します。

- [APICreateOneActivationKey](#) メソッド
- [APICreateRandomActivationKey](#) メソッド
- [APICreateNumberingActivationKey](#) メソッド
- [APIEditOfExpirationDate](#) メソッド

APIExternalLinkKey プロパティ**【機能】**

外部データベースとの「リンク用キー」を設定します。

【構文】

<VB2010>

Public Property **APIExternalLinkKey** As **String**

<C#>

```
public string APIExternalLinkKey { set; get; }
```

【解説】

外部データベースとの「リンク用キー」を設定します。

「リンク用キー」の設定は任意で、認証レスキュー！2 の運用時の必須設定項目ではありません。
この項目は最大 64 バイトです。

次の認証キー情報を作成するメソッドの実行時に当プロパティ(APIExternalLinkKey プロパティ)に設定された値が、認証レスキュー！2 のデータベースの認証キーテーブルの「リンク用キー」に格納されます。

- ・[APICreateOneActivationKey](#) メソッド
- ・[APICreateRandomActivationKey](#) メソッド
- ・[APICreateNumberingActivationKey](#) メソッド

この内、APICreateRandomActivationKey メソッドとAPICreateNumberingActivationKey メソッドは複数の認証キー(レコード)を作成できますが、すべての認証キー(レコード)に当プロパティ(APIExternalLinkKey プロパティ)に設定された値が、作成するすべての認証キー(レコード)の「リンク用キー」に格納されます。

APIxFreeItem1～5 プロパティ

【機能】

自由入力項目 1～5 を設定します。

【構文】

<VB2010>

Public Property **APIxFreeItem1** As **String**

Public Property **APIxFreeItem2** As **String**

Public Property **APIxFreeItem3** As **String**

Public Property **APIxFreeItem4** As **String**

Public Property **APIxFreeItem5** As **String**

<C#>

```
public string APIxFreeItem1 { set; get; }
```

```
public string APIxFreeItem2 { set; get; }
```

```
public string APIxFreeItem3 { set; get; }
```

```
public string APIxFreeItem4 { set; get; }
```

```
public string APIxFreeItem5 { set; get; }
```

【解説】

自由入力項目 1～5 を設定します。

「自由入力項目」の設定は任意で、認証レスキュー！2 の運用時の必須設定項目ではありません。この項目は最大 128 バイトです。

次の認証キー情報を作成するメソッドの実行時に当プロパティ(APIxFreeItem1～5 プロパティ)に設定された値が、認証レスキュー！2 のデータベースの認証キーテーブルの「自由入力項目 1～5」に格納されます。

- ・[APIxCreateOneActivationKey](#) メソッド
- ・[APIxCreateRandomActivationKey](#) メソッド
- ・[APIxCreateNumberingActivationKey](#) メソッド

この内、APIxCreateRandomActivationKey メソッドとAPIxCreateNumberingActivationKey メソッドは認証キーテーブルの認証キー(レコード)を同時に複数作成できますが、当プロパティ(APIxFreeItem1～5 プロパティ)に設定された値が、作成するすべての認証キー(レコード)の「自由入力項目 1～5」に格納されます。

APIxKindOfRandom プロパティ**【機能】**

シリアル No.をランダムに自動作成する場合の文字種別を設定します。

【構文】

<VB2010>

Public Property **APIxKindOfRandom** As **Integer**

<C#>

```
public int APIxKindOfRandom { set; get; }
```

【解説】

シリアル No.をランダムに自動作成する場合の文字種別を設定します。

当プロパティ(APIxKindOfRandom プロパティ)に設定する値は次の通りです。

0: 数字のみ

1: 数字と英字(大文字)

2: 数字と英字(小文字)

3: 数字と英字(大小文字)

次のメソッドの実行時に当プロパティ(APIxKindOfRandom プロパティ)を設定します。

・[APIxCreateRandomActivationKey](#) メソッド

APIxLicenseCount プロパティ**【機能】**

ライセンス数を設定します。

【構文】

<VB2010>

Public Property **APIxLicenseCount** As **Integer**

<C#>

```
public int APIxLicenseCount { set; get; }
```

【解説】

ライセンス数を設定します。

次のメソッドの実行時に当プロパティ(APIxLicenseCount プロパティ)に()内のライセンス数の各値を設定します。

- ・[APIxCreateOneActivationKey](#) メソッド(作成するライセンス数)
- ・[APIxCreateRandomActivationKey](#) メソッド(作成するライセンス数)
- ・[APIxCreateNumberingActivationKey](#) メソッド(作成するライセンス数)

APIxNumberingCount プロパティ**【機能】**

シリアル No.を作成する数を設定します。

【構文】

<VB2010>

Public Property **APIxNumberingCount** As **Integer**

<C#>

```
public int APIxNumberingCount { set; get; }
```

【解説】

シリアル No.を作成する数を設定します。

次のメソッドの実行時に当プロパティ(APIxNumberingCount プロパティ)を設定します。

- ・[APIxCreateRandomActivationKey](#) メソッド
- ・[APIxCreateNumberingActivationKey](#) メソッド

APIxProductID プロパティ**【機能】**

プロダクト ID を設定します。

【構文】

<VB2010>

Public Property **APIxProductID** As **String**

<C#>

```
public string APIxProductID { set; get; }
```

【解説】

プロダクト ID を設定します。

次のメソッドの実行時に当プロパティ(APIxProductID プロパティ)に()内のプロダクト ID の各値を設定します。

- ・[APIxCreateOneActivationKey](#) メソッド(作成するプロダクト ID)
- ・[APIxCreateRandomActivationKey](#) メソッド(作成するプロダクト ID)
- ・[APIxCreateNumberingActivationKey](#) メソッド(作成するプロダクト ID)
- ・[APIxDeleteActivationKey](#) メソッド(削除するプロダクト ID)
- ・[APIxEditOfExpirationDate](#) メソッド(有効期限を変更したいプロダクト ID)

APIxProxyServerAddress プロパティ**【機能】**

プロキシサーバーのアドレスを設定します。

【構文】

<VB2010>

Public Property **APIxProxyServerAddress** As **String**

<C#>

```
public string APIxProxyServerAddress { set; get; }
```

【解説】

プロキシサーバーのアドレスを設定します。

インターネットに接続する際にプロキシサーバーを使用する場合のみ設定が必要です。

プロキシサーバーを使用する場合は、当プロパティを含む次のプロパティに適切な設定が必要です。

- [APIxProxyServerAddress](#) プロパティ(プロキシサーバーのアドレス) → **当プロパティ**
- [APIxProxyServerPort](#) プロパティ(プロキシサーバーのポート)
- [APIxProxyServerUserName](#) プロパティ(プロキシサーバーのユーザ名)
- [APIxProxyServerPassword](#) プロパティ(プロキシサーバーのパスワード)

プロキシサーバーを使用する場合、当プロパティの値を利用するメソッドは次の通りです。

- [APIxCreateOneActivationKey](#) メソッド(製品 1 本分の認証キーの新規作成)
- [APIxCreateRandomActivationKey](#) メソッド(認証キーのシリアル No.をランダムに自動作成)
- [APIxCreateNumberingActivationKey](#) メソッド(認証キーのシリアル No.を自動ナンバリング作成)
- [APIxDeleteActivationKey](#) メソッド(作成済みの認証キーの削除)
- [APIxEditOfExpirationDate](#) メソッド(既存の有効期限を変更)

APIxProxyServerPassword プロパティ**【機能】**

プロキシサーバーのパスワードを設定します。

【構文】

<VB2010>

Public Property **APIxProxyServerPassword** As **String**

<C#>

```
public string APIxProxyServerPassword { set; get; }
```

【解説】

プロキシサーバーのパスワードを設定します。

インターネットに接続する際にプロキシサーバーを使用する場合のみ設定が必要です。

プロキシサーバーを使用する場合は、当プロパティを含む次のプロパティに適切な設定が必要です。

- ・[APIxProxyServerAddress](#) プロパティ(プロキシサーバーのアドレス)
- ・[APIxProxyServerPort](#) プロパティ(プロキシサーバーのポート)
- ・[APIxProxyServerUserName](#) プロパティ(プロキシサーバーのユーザ名)
- ・[APIxProxyServerPassword](#) プロパティ(プロキシサーバーのパスワード)→**当プロパティ**

プロキシサーバーを使用する場合、当プロパティの値を利用するメソッドは次の通りです。

- ・[APIxCreateOneActivationKey](#) メソッド(製品 1 本分の認証キーの新規作成)
- ・[APIxCreateRandomActivationKey](#) メソッド(認証キーのシリアル No.をランダムに自動作成)
- ・[APIxCreateNumberingActivationKey](#) メソッド(認証キーのシリアル No.を自動ナンバリング作成)
- ・[APIxDeleteActivationKey](#) メソッド(作成済みの認証キーの削除)
- ・[APIxEditOfExpirationDate](#) メソッド(既存の有効期限を変更)

APIxProxyServerPort プロパティ**【機能】**

プロキシサーバーのポートを設定します。

【構文】

<VB2010>

Public Property **APIxProxyServerPort** As **String**

<C#>

```
public string APIxProxyServerPort { set; get; }
```

【解説】

プロキシサーバーのポートを設定します。

インターネットに接続する際にプロキシサーバーを使用する場合のみ設定が必要です。

プロキシサーバーを使用する場合は、当プロパティを含む次のプロパティに適切な設定が必要です。

- [APIxProxyServerAddress](#) プロパティ(プロキシサーバーのアドレス)
- [APIxProxyServerPort](#) プロパティ(プロキシサーバーのポート) → **当プロパティ**
- [APIxProxyServerUserName](#) プロパティ(プロキシサーバーのユーザ名)
- [APIxProxyServerPassword](#) プロパティ(プロキシサーバーのパスワード)

プロキシサーバーを使用する場合、当プロパティの値を利用するメソッドは次の通りです。

- [APIxCreateOneActivationKey](#) メソッド(製品 1 本分の認証キーの新規作成)
- [APIxCreateRandomActivationKey](#) メソッド(認証キーのシリアル No.をランダムに自動作成)
- [APIxCreateNumberingActivationKey](#) メソッド(認証キーのシリアル No.を自動ナンバリング作成)
- [APIxDeleteActivationKey](#) メソッド(作成済みの認証キーの削除)
- [APIxEditOfExpirationDate](#) メソッド(既存の有効期限を変更)

APIxProxyServerUserName プロパティ**【機能】**

プロキシサーバーのユーザ名を設定します。

【構文】

<VB2010>

Public Property **APIxProxyServerUserName** As **String**

<C#>

```
public string APIxProxyServerUserName { set; get; }
```

【解説】

プロキシサーバーのユーザ名を設定します。

インターネットに接続する際にプロキシサーバーを使用する場合のみ設定が必要です。

プロキシサーバーを使用する場合は、当プロパティを含む次のプロパティに適切な設定が必要です。

- ・[APIxProxyServerAddress](#) プロパティ(プロキシサーバーのアドレス)
- ・[APIxProxyServerPort](#) プロパティ(プロキシサーバーのポート)
- ・[APIxProxyServerUserName](#) プロパティ(プロキシサーバーのユーザ名)→当プロパティ
- ・[APIxProxyServerPassword](#) プロパティ(プロキシサーバーのパスワード)

プロキシサーバーを使用する場合、当プロパティの値を利用するメソッドは次の通りです。

- ・[APIxCreateOneActivationKey](#) メソッド(製品 1 本分の認証キーの新規作成)
- ・[APIxCreateRandomActivationKey](#) メソッド(認証キーのシリアル No.をランダムに自動作成)
- ・[APIxCreateNumberingActivationKey](#) メソッド(認証キーのシリアル No.を自動ナンバリング作成)
- ・[APIxDeleteActivationKey](#) メソッド(作成済みの認証キーの削除)
- ・[APIxEditOfExpirationDate](#) メソッド(既存の有効期限を変更)

APIxSerialNo プロパティ**【機能】**

シリアル No.を設定します。

【構文】

<VB2010>

Public Property **APIxSerialNo** As **String**

<C#>

```
public string APIxSerialNo { set; get; }
```

【解説】

シリアル No.を設定します。

次のメソッドの実行時に当プロパティ(APIxProductID プロパティ)に()内のシリアル No.の各値を設定します。

- ・[APIxCreateOneActivationKey](#) メソッド(作成するシリアル No.)
- ・[APIxDeleteActivationKey](#) メソッド(削除するシリアル No.)
- ・[APIxEditOfExpirationDate](#) メソッド(有効期限を変更したいシリアル No.)

APIxSerialNoString プロパティ**【機能】**

認証キー情報作成直後のシリアル No.を取得します。(取得専用)

【構文】

<VB2010>

Public Property **APIxSerialNoString** As **String**

<C#>

```
public string APIxSerialNoString { get; }
```

【解説】

認証キー情報作成直後のシリアル No.を取得します。(取得専用)

認証キー情報とは、プロダクト ID、シリアル No.、ライセンス数、有効期限利用の有無、有効期限などの総称です。

次のメソッドを実行すると当プロパティ(APIxSerialNoString プロパティ)に新しく生成されたシリアル No の文字列が設定されます。

- [APIxCreateOneActivationKey](#) メソッド
- [APIxCreateNumberingActivationKey](#) メソッド
- [APIxCreateRandomActivationKey](#) メソッド

複数のシリアル No.が生成された場合は、デリミタのカンマ(,)で区切られた文字列が設定されます。

(例) "00000001,00000002,00000003"

上記メソッド実行時のエラーによるシリアル No.の未生成時は、当プロパティ(APIxSerialNo プロパティ)には、""(空文字列)が設定されます。

APIxStartNo プロパティ**【機能】**

シリアル No.を自動的にナンバリングして作成する場合の開始番号を設定します。

【構文】

<VB2010>

Public Property **APIxStartNo** As **Integer**

<C#>

```
public int APIxStartNo { set; get; }
```

【解説】

シリアル No.を自動的にナンバリングして作成する場合の開始番号を設定します。

次のメソッドの実行時に当プロパティ(APIxStartNo プロパティ)を設定します。

・[APIxCreateNumberingActivationKey](#) メソッド

APIxStartFixedString プロパティ**【機能】**

作成するシリアル No.の上位固定文字列を設定します。

【構文】

<VB2010>

Public Property **APIxStartFixedString** As **String**

<C#>

public **string** **APIxStartFixedString** { set; get; }

【解説】

作成するシリアル No.の上位固定文字列を設定します。

(例) 作成するシリアル No.の上位 4 桁を"0001"で固定する場合

<VB> Class1.myAPIActivate.APIxStartFixedString = "0001"

<C#> Class1.myAPIActivate.APIxStartFixedString = "0001";

<VC++> theApp.m_pAPIActivation-> APIxStartFixedString = "0001";

次のメソッドの実行時に当プロパティ(APIxStartFixedString プロパティ)に上位固定文字列を設定します。

- ・[APIxCreateRandomActivationKey](#) メソッド
- ・[APIxCreateNumberingActivationKey](#) メソッド

APIxStepNo プロパティ**【機能】**

シリアル No.を自動的にナンバリングして作成する場合の間隔(ステップ)数を設定します。

【構文】

<VB2010>

Public Property **APIxStepNo** As **Integer**

<C#>

public **int** **APIxStepNo** { set; get; }

【解説】

シリアル No.を自動的にナンバリングして作成する場合の間隔(ステップ)数を設定します。

次のメソッドの実行時に当プロパティ(APIxStepNo プロパティ)を設定します。

・[APIxCreateNumberingActivationKey](#) メソッド

APIUseProxyServer プロパティ

【機能】

プロキシサーバーの使用区分(デフォルト:False)を設定します。

【構文】

<VB2010>

Public Property **APIUseProxyServer** As **Boolean**

<C#>

```
public bool APIUseProxyServer { set; get; }
```

【解説】

プロキシサーバーの使用区分(デフォルト:False)を設定します。

インターネットに接続する際にプロキシサーバーを使用するかどうかを設定します。

プロキシサーバーを使用する場合は当プロパティに True を、使用しない場合は False をそれぞれ設定します。

プロキシサーバーを使用する場合は、次のプロパティに適切な設定が必要です。

- ・[APIProxyServerAddress](#) プロパティ(プロキシサーバーのアドレス)
- ・[APIProxyServerPort](#) プロパティ(プロキシサーバーのポート)
- ・[APIProxyServerUserName](#) プロパティ(プロキシサーバーのユーザ名)
- ・[APIProxyServerPassword](#) プロパティ(プロキシサーバーのパスワード)

プロキシサーバーを使用する場合、当プロパティの値を利用するメソッドは次の通りです。

- ・[APICreateOneActivationKey](#) メソッド(製品 1 本分の認証キーの新規作成)
- ・[APICreateRandomActivationKey](#) メソッド(認証キーのシリアル No.をランダムに自動作成)
- ・[APICreateNumberingActivationKey](#) メソッド(認証キーのシリアル No.を自動ナンバリング作成)
- ・[APIDeleteActivationKey](#) メソッド(作成済みの認証キーの削除)
- ・[APIEditOfExpirationDate](#) メソッド(既存の有効期限を変更)

APIxUseRental プロパティ

【機能】

レンタル機能の使用 (False: 利用しない、True: 利用する) を設定します。

【構文】

<VB2010>

Public Property **APIxUseRental** As **Boolean**

<C#>

```
public bool APIxUseRental { set; get; }
```

【解説】

貴社がエンドユーザに配布するアプリケーションでレンタル機能を使用するかどうかを設定します。レンタル機能を使用しない場合は False、使用する場合は True を設定します。

認証レスキュー！2 の認証 UI ライブラリ (DLL) の次表のプロパティに何を設定しているかにより、当プロパティ (APIxUseRental プロパティ) の値が決まります。

| 認証 UI ライブラリ (DLL) の利用形態 | 認証 UI ライブラリ (DLL) で使用しているプロパティ | 認証 UI ライブラリ (DLL) でのプロパティの値 | 当プロパティ (APIxUseRental プロパティ) に設定する値 |
|-------------------------|--------------------------------|-----------------------------|-------------------------------------|
| UI 系で使用している場合 | RentalPeriod プロパティ | 0 | False |
| | | 1~1100 | True |
| API 系で使用している場合 | APIRentalPeriod プロパティ | 0 | False |
| | | 1~1100 | True |

※お客様のパッケージ製品にマルチライセンス (例: 10 ライセンス) が含まれる場合はレンタル機能は使用できません。レンタル機能を使用するにはシングルライセンス (1 ライセンス) の製品である必要があります。また、レンタル機能を使用する場合は「電話で認証登録」機能は利用できません。

APIxUseExpirationDate プロパティ**【機能】**

有効期限利用の有無 (False: 利用しない、True: 利用する) を設定します。

【構文】

<VB2010>

Public Property **APIxUseExpirationDate** As **Boolean**

<C#>

```
public bool APIxUseExpirationDate { set; get; }
```

【解説】

有効期限利用の有無 (False: 利用しない、True: 利用する) を設定します。

次のメソッドの実行時に当プロパティ (APIxUseExpirationDate プロパティ) に有効期限利用の有無を設定します。

- [APIxCreateOneActivationKey](#) メソッド
- [APIxCreateRandomActivationKey](#) メソッド
- [APIxCreateNumberingActivationKey](#) メソッド
- [APIxEditOfExpirationDate](#) メソッド

APIxWebServiceBasicAuthenticationPassword プロパティ**【機能】**

Web サービス時の基本認証パスワードを設定します。

【構文】

<VB2010>

Public Property **APIxWebServiceBasicAuthenticationPassword** As **String**

<C#>

```
public string APIxWebServiceBasicAuthenticationPassword { set; get; }
```

【解説】

Web サーバーで基本認証を使用する場合に、そのパスワードを指定します。

基本認証に関しては、[APIxWebServiceUseBasicAuthentication](#) プロパティを参照してください。

APIxWebServiceBasicAuthenticationUserName プロパティ**【機能】**

Web サービス時の基本認証ユーザ名を設定します。

【構文】

<VB2010>

Public Property **APIxWebServiceBasicAuthenticationUserName** As **String**

<C#>

```
public string APIxWebServiceBasicAuthenticationUserName { set; get; }
```

【解説】

Web サーバーで基本認証を使用する場合に、そのユーザ名を指定します。

基本認証に関しては、[APIxWebServiceUseBasicAuthentication](#) プロパティを参照してください。

APIxWebServiceCheckPassword プロパティ**【機能】**

Web サービス確認パスワード(8 文字以上)を設定します。

【構文】

<VB2010>

Public Property **APIxWebServiceCheckPassword** As **String**

<C#>

```
public string APIxWebServiceCheckPassword { set; get; }
```

【解説】

Web サービスを利用する場合の確認用のパスワードを設定します。ここでは、Web サーバー側設定アプリケーション「認証 Web サービス」の環境設定処理の Web サービスの確認パスワードと同じものを設定します。

確認パスワードは必須項目です、省略はできません。8 文字以上で半角の次の文字が使用できません。

- ・大文字の英字(A～Z)
- ・小文字の英字(a～z)
- ・数字(0～9)
- ・記号(+-*!/#\$%&()=¥@<>?)

APIxWebServiceTimeout プロパティ**【機能】**

Web サービスのタイムアウト(デフォルト: 60 秒)を設定します。

【構文】

<VB2010>

Public Property **APIxWebServiceTimeout** As **Integer**

<C#>

```
public int APIxWebServiceTimeout { set; get; }
```

【解説】

Web サービスに接続して応答を待つ最大時間を秒単位で指定します。初期値は 60 秒です。

APIxWebServiceURL プロパティ**【機能】**

Web サービスの URL を設定します。

【構文】

<VB2010>

Public Property **APIxWebServiceURL** As **String**

<C#>

```
public string APIxWebServiceURL { set; get; }
```

【解説】

認証に関するシステムを Web サービスとして提供する Web サーバーの URL を指定します。
以下に例を示します。

自 PC のローカルホストの Web サーバー(IIS)にアクセスする例:

```
"http://localhost/Nr2WebService/Service.aspx"
```

(注)この例は実際の運用ではありえません。貴社のアプリケーションで認証 UI ライブラリ(DLL)を使用した開発時に、テスト用に使用される URL です。

自社 Web サーバー(IIS)にアクセスさせる例:

```
"http://www.newtone.co.jp/Nr2WebService/Service.aspx"
```

クラウドサービス Microsoft Azure の Web アプリ (旧 Web サイト)に配置した Web サービスを利用する例:

```
"http://newtonecojp.azurewebsites.net/Service.aspx"
```

APIxWebServiceUseBasicAuthentication プロパティ**【機能】**

Web サービス時の基本認証の使用(デフォルト:False)を設定します。

【構文】

<VB2010>

Public Property **APIxWebServiceUseBasicAuthentication** As **Boolean**

<C#>

```
public bool APIxWebServiceUseBasicAuthentication { set; get; }
```

【解説】

Web サーバーで基本認証を使用する場合は、当プロパティを True にします。
初期値は、False(基本認証を使用しない)です。

当プロパティは、Web サーバー(IIS)側で特定のアカウント(ユーザ名とパスワード)でアクセスできるフォルダにこの Web サービスが配置してある場合に使用できるセキュリティ設定です。

Web サーバーで基本認証を使用する一般的な手順は次の通りです。

1.サーバーPC 上でユーザを作成。

この際のユーザ名とパスワードがそのまま基本認証に使われます。

2.基本認証フォルダのセキュリティ設定

フォルダのプロパティを開き、セキュリティタブで上記 1 のユーザ名を 追加し、「読み取り」権限を付与します。

3.IIS でのセキュリティ設定

IIS で該当フォルダに対し「認証」の設定で「匿名認証」を無効にして 「基本認証」を有効にします。

なお、Web サーバーでの基本認証の詳細につきましてはマイクロソフト社の関連ドキュメントをご覧ください。

<ASP.NET 系メソッド>

メソッド一覧

| メソッド | 機能 |
|--------------------------------------------------|----------------------------------------------|
| APIxCreateNumberingActivationKey | 製品の認証キー情報を指定した数だけシリアル No.を自動的にナンバリングして作成します。 |
| APIxCreateOneActivationKey | 製品 1 本分の認証キー情報を新規に作成します。 |
| APIxCreateRandomActivationKey | 製品の認証キー情報を指定した数だけ、シリアル No.をランダムに自動作成します。 |
| APIxDeleteActivationKey | 作成済みの認証キー情報を削除します。 |
| APIxEditOfExpirationDate | 既存の認証キー情報内の有効期限を変更します。 |

APIxCreateNumberingActivationKey メソッド

【機能】

製品の認証キー情報を指定した数だけシリアル No.を自動的にナンバリングして作成します。

【構文】

<VB2010>

Public Function **APIxCreateNumberingActivationKey** () As **Integer**

<C#>

public **int** **APIxCreateNumberingActivationKey** ()

【引数】

なし

【戻り値】

生成されたシリアル No.の数。

エラー時は 0 が設定され、戻り値とは別に、[APIxErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

また、APIxSerialNoString プロパティに新しく生成されたシリアル No.をデリミタのカンマ(,)で区切った文字列が設定されます。シリアル No.の未生成時は、""(空文字列)が設定されます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| プロパティ | 機能 | |
|-----------------------------------------------------------|---------------------------------------|-------------|
| APIxUseProxyServer | プロキシサーバーの使用区分 | |
| APIxProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー使用時 |
| APIxProxyServerPort | プロキシサーバーのポート | |
| APIxProxyServerUserName | プロキシサーバーのユーザー名 | |
| APIxProxyServerPassword | プロキシサーバーのパスワード | |
| APIxWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード(基本認証使用時) | |
| APIxWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザー名(基本認証使用時) | |
| APIxWebServiceCheckPassword | Web サービス確認パスワード | |
| APIxWebServiceTimeout | Web サービスのタイムアウト | |
| APIxWebServiceURL | Web サービスの URL | |
| APIxWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 | |
| APIxProductID | プロダクト ID | |
| APIxLicenseCount | ライセンス数 | |
| APIxUseExpirationDate | 有効期限利用の有無 (False : 利用しない、True : 利用する) | |
| APIxExpirationDate | 有効期限 | |
| APIxStartFixedString | 作成するシリアル No.の上位固定文字列 | |

| | |
|-------------------------------------|------------------------------------|
| APIxStartNo | 作成するシリアル No.開始番号 |
| APIxStepNo | 作成するシリアル No.間隔(ステップ)数 |
| APIxNumberingCount | 作成するシリアル No.の数 |
| APIxUseRental | レンタル機能利用の有無(False:利用しない、True:利用する) |
| APIxExternalLinkKey | 外部データベースとのリンク用キー項目を設定 |
| APIxFreeItem1~5 | 自由入力項目 1~5 を設定 |

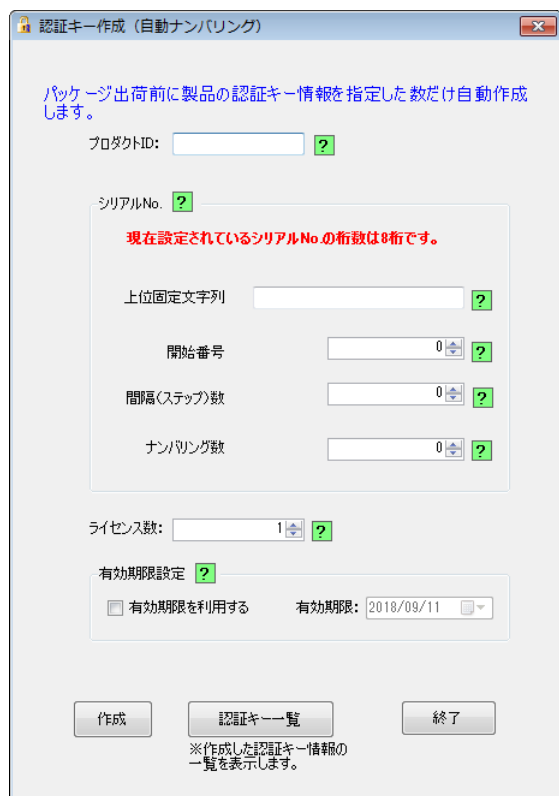
【解説】

このメソッドは、認証キー情報を指定した数だけシリアル No.を自動的にナンバリングして作成します。

認証キー情報とは、プロダクト ID、シリアル No.、ライセンス数、有効期限利用の有無、有効期限などの総称です。

認証管理システム(InsideSystem.exe)の「認証キー作成(自動ナンバリング)」処理と同等の機能を提供します。

◆「認証キー作成(自動ナンバリング)」処理の画面例(参考)



◆処理手順

一連の処理の手順は次の通りです。

[1]上記の【必須設定プロパティ】一覧にある各プロパティに適切な値を設定します。

[2]当メソッド(APIxCreateNumberingActivationKey メソッド)を実行します。

APIxCreateOneActivationKey メソッド

【機能】

製品 1 本分の認証キー情報を新規に作成します。

【構文】

<VB2010>

Public Function **APIxCreateOneActivationKey** () As **Integer**

<C#>

public **int** **APIxCreateOneActivationKey**()

【引数】

なし

【戻り値】

生成されたシリアル No.の数(当メソッドの場合は 1)。

エラー時は 0 が設定され、戻り値とは別に、[APIxErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

また、APIxSerialNoString プロパティには新しく生成されたシリアル No. (当メソッドの場合は、APIxSerialNo プロパティに指定した値)が設定されます。エラーによるシリアル No.の未生成時は、この APIxSerialNoString プロパティには、"" (空文字列)が設定されます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| プロパティ | 機能 | |
|-----------------------------------------------------------|---------------------------------------|-------------|
| APIxUseProxyServer | プロキシサーバーの使用区分 | |
| APIxProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー使用時 |
| APIxProxyServerPort | プロキシサーバーのポート | |
| APIxProxyServerUserName | プロキシサーバーのユーザー名 | |
| APIxProxyServerPassword | プロキシサーバーのパスワード | |
| APIxWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード(基本認証使用時) | |
| APIxWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザー名(基本認証使用時) | |
| APIxWebServiceCheckPassword | Web サービス確認パスワード | |
| APIxWebServiceTimeout | Web サービスのタイムアウト | |
| APIxWebServiceURL | Web サービスの URL | |
| APIxWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 | |
| APIxProductID | (作成する)プロダクト ID | |
| APIxSerialNo | (作成する)シリアル No. | |
| APIxLicenseCount | (作成する)ライセンス数 | |
| APIxUseExpirationDate | 有効期限利用の有無 (False : 利用しない、True : 利用する) | |
| APIxExpirationDate | 有効期限 | |

| | |
|-------------------------------------|---------------------------------------|
| APIxUseRental | レンタル機能利用の有無 (False: 利用しない、True: 利用する) |
| APIxExternalLinkKey | 外部データベースとのリンク用キー項目を設定 |
| APIxFreeItem1~5 | 自由入力項目 1~5 を設定 |

【解説】

このメソッドは、製品 1 本分の認証キー情報を新規に作成します。

認証キー情報とは、プロダクト ID、シリアル No.、ライセンス数、有効期限利用の有無、有効期限などの総称です。

認証管理システム (InsideSystem.exe) の「認証キー作成 (個別)」処理と同等の機能を提供します。

◆「認証キー作成 (個別)」処理の画面例 (参考)

◆処理手順

一連の処理の手順は次の通りです。

[1] 上記の【必須設定プロパティ】一覧にある各プロパティに適切な値を設定します。

[2] 当メソッド (APIxCreateOneActivationKey メソッド) を実行します。

APIxCreateRandomActivationKey メソッド

【機能】

製品の認証キー情報を指定した数だけ、シリアル No.をランダムに自動作成します。

【構文】

<VB2010>

Public Function **APIxCreateRandomActivationKey** () As **Integer**

<C#>

public **int** **APIxCreateRandomActivationKey** ()

【引数】

なし

【戻り値】

生成されたシリアル No.の数。

エラー時は 0 が設定され、戻り値とは別に、[APIxErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

また、[APIxSerialNoString](#) プロパティに新しく生成されたシリアル No.をデリミタのカンマ(,)で区切った文字列が設定されます。シリアル No.の未生成時は、"" (空文字列)が設定されます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

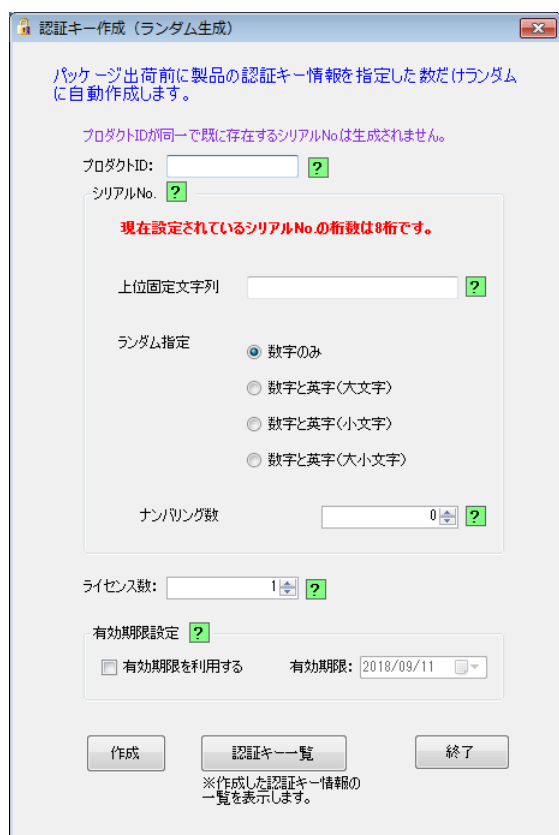
| プロパティ | 機能 | |
|-----------------------------------------------------------|---------------------------------------|-------------|
| APIxUseProxyServer | プロキシサーバーの使用区分 | |
| APIxProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー使用時 |
| APIxProxyServerPort | プロキシサーバーのポート | |
| APIxProxyServerUserName | プロキシサーバーのユーザー名 | |
| APIxProxyServerPassword | プロキシサーバーのパスワード | |
| APIxWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード(基本認証使用時) | |
| APIxWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザー名(基本認証使用時) | |
| APIxWebServiceCheckPassword | Web サービス確認パスワード | |
| APIxWebServiceTimeout | Web サービスのタイムアウト | |
| APIxWebServiceURL | Web サービスの URL | |
| APIxWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 | |
| APIxProductID | プロダクト ID | |
| APIxLicenseCount | ライセンス数 | |
| APIxUseExpirationDate | 有効期限利用の有無 (False : 利用しない、True : 利用する) | |
| APIxExpirationDate | 有効期限 | |
| APIxStartFixedString | シリアル No.の上位固定文字列 | |
| APIxKindOfRandom | シリアル No.のランダム文字種の指定 | |

| | |
|-------------------------------------|----------------------------------------------------|
| | 0: 数字のみ、1: 数字と英字(大文字)、2: 数字と英字(小文字)、3: 数字と英字(大小文字) |
| APIxNumberingCount | 作成するシリアル No.の数 |
| APIxUseRental | レンタル機能利用の有無(False: 利用しない、True: 利用する) |
| APIxExternalLinkKey | 外部データベースとのリンク用キー項目を設定 |
| APIxFreeItem1~5 | 自由入力項目 1~5 を設定 |

【解説】

このメソッドは、認証キー情報を指定した数だけ、シリアル No.をランダムに自動作成します。認証キー情報とは、プロダクト ID、シリアル No.、ライセンス数、有効期限利用の有無、有効期限などの総称です。認証管理システム(InsideSystem.exe)の「認証キー作成(ランダム生成)」処理と同等の機能を提供します。

◆「認証キー作成(ランダム生成)」処理の画面例(参考)



◆処理手順

一連の処理の手順は次の通りです。

- [1]上記の【必須設定プロパティ】一覧にある各プロパティに適切な値を設定します。
- [2]当メソッド(APIxCreateRandomActivationKey メソッド)を実行します。

APIxDeleteActivationKey メソッド

【機能】

作成済みの認証キー情報を削除します。

【構文】

<VB2010>

Public Function **APIxDeleteActivationKey** () As **Integer**

<C#>

public **int** **APIxDeleteActivationKey** ()

【引数】

なし

【戻り値】

削除に成功した場合は 1 が設定されます。

エラー時は 0 が設定され、戻り値とは別に、[APIxErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| プロパティ | 機能 | |
|-----------------------------------------------------------|------------------------------|-------------|
| APIxUseProxyServer | プロキシサーバーの使用区分 | |
| APIxProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー使用時 |
| APIxProxyServerPort | プロキシサーバーのポート | |
| APIxProxyServerUserName | プロキシサーバーのユーザー名 | |
| APIxProxyServerPassword | プロキシサーバーのパスワード | |
| APIxWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード(基本認証使用時) | |
| APIxWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザー名(基本認証使用時) | |
| APIxWebServiceCheckPassword | Web サービス確認パスワード | |
| APIxWebServiceTimeout | Web サービスのタイムアウト | |
| APIxWebServiceURL | Web サービスの URL | |
| APIxWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 | |
| APIxProductID | (削除する)プロダクト ID | |
| APIxSerialNo | (削除する)シリアル No. | |

【解説】

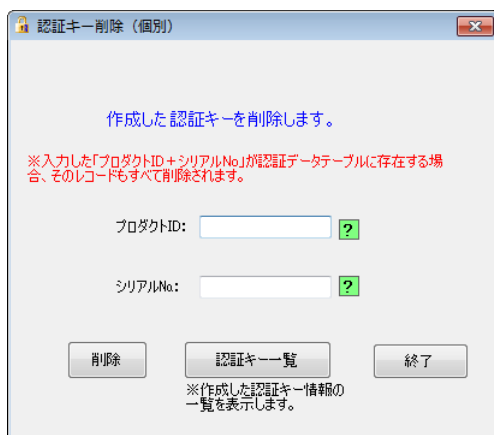
このメソッドは、作成済みの認証キー情報を削除します。

認証キー情報とは、プロダクト ID、シリアル No.、ライセンス数、有効期限利用の有無、有効期限などの総称です。

認証管理システム(InsideSystem.exe)の「認証キー削除(個別)」処理と同等の機能を提供します。

指定した認証キーテーブルの「プロダクト ID」+「シリアル No.」が認証データテーブルに存在する場合は、そのレコードもすべて削除されます。

◆「認証キー作成(個別)」処理の画面例(参考)



◆処理手順

一連の処理の手順は次の通りです。

[1]上記の【必須設定プロパティ】一覧にある各プロパティに適切な値を設定します。

[2]当メソッド(APIxDeleteActivationKey メソッド)を実行します。

APIxEditOfExpirationDate メソッド

【機能】

既存の認証キー情報内の有効期限を変更します。

【構文】

<VB2010>

Public Function **APIxEditOfExpirationDate** () As **Integer**

<C#>

public **int** **APIxEditOfExpirationDate** ()

【引数】

なし

【戻り値】

有効期限の変更成功した場合は 1 が設定されます。

エラー時は 0 が設定され、戻り値とは別に、[APIxErrorStatus](#) プロパティに当メソッド実行後のエラー内容がセットされます。

【必須設定プロパティ】

このメソッドの利用には、あらかじめ次のプロパティに適切な値が設定されている必要があります。

| プロパティ | 機能 | |
|-----------------------------------------------------------|---------------------------------------|-------------|
| APIxUseProxyServer | プロキシサーバーの使用区分 | |
| APIxProxyServerAddress | プロキシサーバーのアドレス | プロキシサーバー使用時 |
| APIxProxyServerPort | プロキシサーバーのポート | |
| APIxProxyServerUserName | プロキシサーバーのユーザー名 | |
| APIxProxyServerPassword | プロキシサーバーのパスワード | |
| APIxWebServiceBasicAuthenticationPassword | Web サービス時の基本認証パスワード(基本認証使用時) | |
| APIxWebServiceBasicAuthenticationUserName | Web サービス時の基本認証ユーザー名(基本認証使用時) | |
| APIxWebServiceCheckPassword | Web サービス確認パスワード | |
| APIxWebServiceTimeout | Web サービスのタイムアウト | |
| APIxWebServiceURL | Web サービスの URL | |
| APIxWebServiceUseBasicAuthentication | Web サービス時の基本認証の使用区分 | |
| APIxProductID | (有効期限を変更する)プロダクト ID | |
| APIxSerialNo | (有効期限を変更する)シリアル No. | |
| APIxUseExpirationDate | 有効期限利用の有無 (False : 利用しない、True : 利用する) | |
| APIxExpirationDate | (新しい)有効期限 | |

【解説】

このメソッドは、既存の認証キー情報の内の有効期限を変更します。

認証キー情報とは、プロダクト ID、シリアル No.、ライセンス数、有効期限利用の有無、有効期限な

どの総称です。

認証管理システム (InsideSystem.exe) の「認証キー編集 (表形式)」処理の一部と同等の機能を提供します。

◆「認証キー編集 (表形式)」処理の画面例 (参考)

製品の認証キー情報を検索して編集します。

検索

プロダクトID: ? (未入力: 指定なし)

シリアルNo.: 先頭指定文字列: ? (未入力: 指定なし)

有効期限利用のみ表示

特定の有効期限のみ表示: 2018/09/11 ~ 2018/09/11

検索実行

※既に認証データが存在する行や、入力不可項目はグレーで表示され編集はできません。
ただし、「有効期限」は更新のため変更可能です。

| | プロダクトID | シリアルNo. | ライセンス数 | プラス許可数 | 有効期限利用 | 有効期限 (例: 2018/09/11) | 作成日時 | 編集 |
|-----|-------------------|-----------|--------|--------|-------------------------------------|----------------------------|---------------------|----|
| ▶ 1 | 33333-33333-33333 | ABC00001 | 1 | 0 | <input checked="" type="checkbox"/> | 2017/10/31 | 2017/10/04 15:26:40 | ○ |
| 2 | 33333-33333-33333 | ABC00002 | 1 | 0 | <input checked="" type="checkbox"/> | 2017/10/31 | 2017/10/04 15:26:40 | ○ |
| 3 | 33333-33333-33333 | ABC00003 | 1 | 0 | <input checked="" type="checkbox"/> | 2017/10/31 | 2017/10/04 15:26:40 | ○ |
| 4 | 33333-33333-33333 | ABC00004 | 1 | 0 | <input checked="" type="checkbox"/> | 2017/10/31 | 2017/10/04 15:26:40 | ○ |
| 5 | 33333-33333-33333 | ABC00005 | 1 | 0 | <input checked="" type="checkbox"/> | 2017/10/31 | 2017/10/04 15:26:40 | ○ |
| 6 | 44444-44444-44444 | 555555555 | 1 | 0 | <input checked="" type="checkbox"/> | 2017/10/04 | 2017/10/04 15:27:37 | ○ |
| 7 | 66666-66666-66666 | ABC12722 | 1 | 0 | <input checked="" type="checkbox"/> | 2017/10/04 | 2017/10/04 15:28:07 | ○ |

有効期限の一括設定
上表内の認証キーすべての「有効期限」を一括設定する場合は、次の共通有効期限を指定して「一括設定」ボタンを押します。

共通有効期限: 2018/09/11

一括設定

登録

Excelファイル出力

終了

◆処理手順

一連の処理の手順は次の通りです。

[1]上記の【必須設定プロパティ】一覧にある各プロパティに適切な値を設定します。

[2]当メソッド (APIxEditOfExpirationDate メソッド) を実行します。

代理認証機能について

次の 8 つのメソッド

- 1.「代理認証登録/準備」処理の呼び出し/ProxyActivateRegisterPrepare
- 2.「代理認証登録/確定」処理の呼び出し/ProxyActivateRegisterFix
- 3.「代理認証解除/準備」処理の呼び出し/ProxyActivateRemovePrepare
- 4.「代理有効期限更新/準備」処理の呼び出し/PreparationOfProxyUpdateOfExpirationDate
- 5.「代理有効期限更新/確定」処理の呼び出し/DeterminationOfProxyUpdateOfExpirationDate
- 6.「代理認証登録/実行」処理の呼び出し(代理 PC で利用)/ProxyActivateRegisterExecute
- 7.「代理認証解除/実行」処理の呼び出し(代理 PC で利用)/ProxyActivateRemoveExecute
- 8.「代理有効期限/取得」処理の呼び出し(代理 PC で利用)/GetProxyUpdateOfExpirationDate

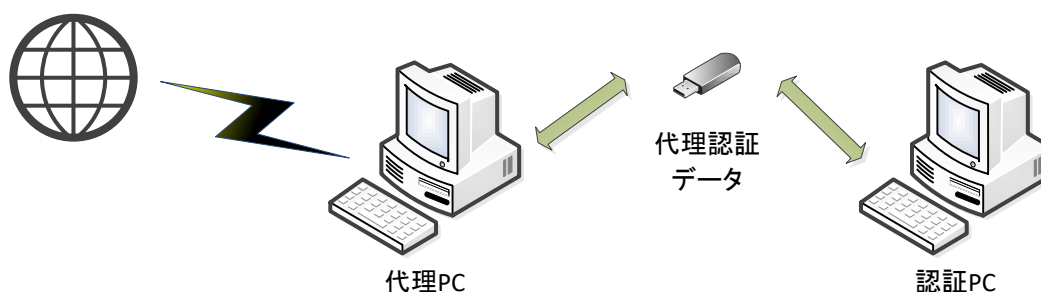
は代理で認証する機能です。

これらの機能は、実際にライセンスの認証登録をして貴社のアプリケーションを動作させたいエンドユーザの PC がインターネット経由での認証ができない状態で、かつ貴社も電話による認証を受け付ける体制をもたない場合の手段として利用できます。

これは、エンドユーザにとっては電話で認証する場合と違い貴社の休業日でも認証でき、貴社にとっては電話受付による人材や体制のためのコストが不要になる、という双方にとってのメリットがあります。

代理認証は、エンドユーザがこの機能を使ってインターネット接続できる他の PC で代わりに認証をして、その認証情報を貴社のアプリケーションを動作させたい PC に保存する、というものです。

次図のような流れになります。



この代理認証機能を使用しないのであれば以降の記載は必要ありませんので読み飛ばしてください。

◆エンドユーザが行う処理

<代理認証登録>

・最終的に認証登録したい PC で、「代理認証登録/準備」処理を行い、代理認証準備データを USB メモリなどに出力します。

・代理に使う PC で、代理認証準備データを読み込み「代理認証登録/実行」処理をインターネット経由で行い、代理認証実行データを USB メモリなどに出力します。

・最終的に認証登録したい PC で、代理認証実行データを読み込み「代理認証登録/確定」処理を行い認証登録が確定します。

<代理認証解除>

- ・最終的に認証解除したい PC で、「代理認証**解除**/準備」処理を行い、代理認証準備データを USB メモリなどに出力します。この時点で認証 PC の認証は解除されます。
- ・代理に使う PC で、代理認証準備データを読み込み「代理認証**解除**/実行」処理をインターネット経由で行い、認証**解除**が確定します。

<代理有効期限更新>

- ・最終的に認証登録済の有効期限を更新したい PC で、「代理**有効期限更新**/準備」処理を行い、代理有効期限更新準備データを USB メモリなどに出力します。
- ・代理に使う PC で、代理有効期限更新準備データを読み込み「代理**有効期限**/取得」処理をインターネット経由で行い、代理有効期限取得データを USB メモリなどに出力します。
- ・最終的に認証登録済の有効期限を更新したい PC で、代理有効期限取得データを読み込み「代理**有効期限更新**/確定」処理を行い有効期限の**更新**が確定します。

◆貴社が行う作業

上記の「エンドユーザが行う処理」をエンドユーザに操作させるために認証 UI ライブラリ(DLL)を利用してアプリケーションに代理機能を組み込む必要があります。

エンドユーザに対し、最終的に認証したい PC で作業させる処理の例は付属の開発言語別のサンプルプロジェクトの **PackageApp** 内に、代理 PC で作業させる処理の例は付属の開発言語別のサンプルプロジェクトの **ProxyApp** にそれぞれあります。

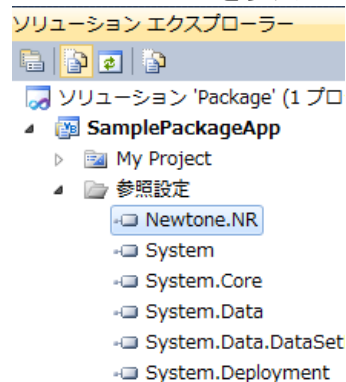
したがって、貴社は最終的に認証したい PC 用のアプリケーションの他に代理 PC で代理処理をするための (**ProxyApp** のような) アプリケーションを作成し配布する必要があります。

■認証 UI ライブラリの参照

Visual Studio 2010 (Visual Basic 2010/C# 2010) の場合

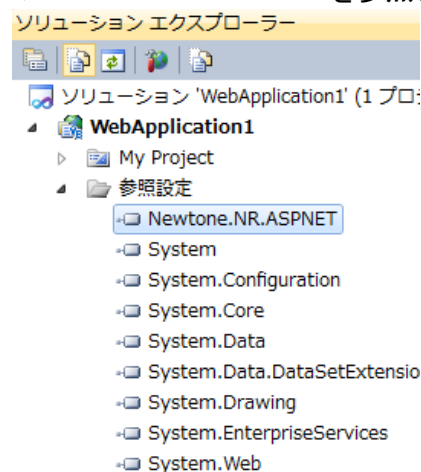
認証レスキュー！の認証 UI ライブラリ(DLL)を使用するプロジェクトの「参照の追加」で Newtone.NR.dll を参照します。

<Newtone.NR.dll を参照した後の参照設定の様子 (Visual Basic 2010)>



また、ASP.NET 系の DLL を使用する場合は、プロジェクトの「参照の追加」で Newtone.NR.ASPNET.dll を参照します。

<Newtone.NR.ASPNET.dll を参照した後の参照設定の様子 (Visual Basic 2010)>



Newtone.NR.dll および Newtone.NR.ASPNET.dll は、インストール先がデフォルトの場合、次のフォルダ内にあります。

.NET Framework4.0 用の DLL

<32bitOS の場合> C:\Program Files\Newtone\NR2\NR2DLL\NRDLL\Framework4.0

<64bitOS の場合> C:\Program Files (x86)\Newtone\NR2\NR2DLL\NRDLL\Framework4.0

.NET Framework3.5 用の DLL

<32bitOS の場合> C:\Program Files\Newtone\NR2\NR2DLL\NRDLL\Framework3.5

<64bitOS の場合> C:\Program Files (x86)\Newtone\NR2\NR2DLL\NRDLL\Framework3.5

Visual Studio 6.0 (Visual Basic 6.0) の場合

認証 UI ライブラリ(DLL)を Visual Basic(以降 VB)6 から利用する方法は次の通りです。

次の 2 つのファイルが必要です。

Newton.NR.dll

Newton.NR.tlb

これらはインストール先がデフォルトの場合、次のフォルダ内にあります。

<32bitOS の場合> C:\Program Files\Newton\NR2\NR2DLL\NRDLL\Framework4.0

<64bitOS の場合> C:\Program Files (x86)\Newton\NR2\NR2DLL\NRDLL\Framework4.0

手順 1: DLL の登録

次の通りコマンドプロンプトなどで DLL の登録(解除)を行います。

<DllPath>は、上記のフォルダパスを示します。

◆DLL の登録

```
C:\WINDOWS\Microsoft.NET\Framework\v4.0.30319\regasm.exe "<DllPath>Newton.NR.dll" /tlb:"<DllPath>Newton.NR.tlb" /codebase /nologo
```

(例:)<32bitOS の場合>

```
C:\WINDOWS\Microsoft.NET\Framework\v4.0.30319\regasm.exe "C:\Program Files\Newton\NR2\NR2DLL\NRDLL\Framework4.0\Newton.NR.dll" /tlb:"C:\Program Files\Newton\NR2\NR2DLL\NRDLL\Framework4.0\Newton.NR.tlb" /codebase /nologo
```

(例:)<64bitOS の場合>

```
C:\WINDOWS\Microsoft.NET\Framework\v4.0.30319\regasm.exe "C:\Program Files (x86)\Newton\NR2\NR2DLL\NRDLL\Framework4.0\Newton.NR.dll" /tlb:"C:\Program Files (x86)\Newton\NR2\NR2DLL\NRDLL\Framework4.0\Newton.NR.tlb" /codebase /nologo
```

また、登録した DLL を解除する場合は次の通りです。

◆DLL の解除

```
C:\WINDOWS\Microsoft.NET\Framework\v4.0.30319\regasm.exe "<DllPath>Newton.NR.dll" /tlb:"<DllPath>Newton.NR.tlb" /u /nologo
```

(例:)<32bitOS の場合>

```
C:\WINDOWS\Microsoft.NET\Framework\v4.0.30319\regasm.exe "C:\Program Files\Newton\NR2\NR2DLL\NRDLL\Framework4.0\Newton.NR.dll" /tlb:"C:\Program Files\Newton\NR2\NR2DLL\NRDLL\Newton.NR.tlb" /u /nologo
```

(例:)<64bitOS の場合>

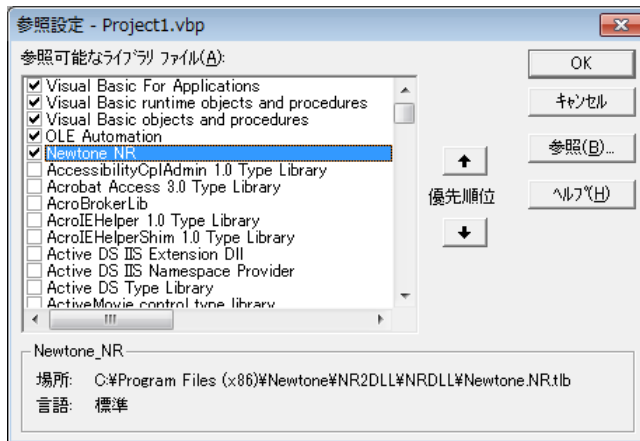
```
C:\WINDOWS\Microsoft.NET\Framework\v4.0.30319\regasm.exe "C:\Program Files (x86)\Newton\NR2\NR2DLL\NRDLL\Framework4.0\Newton.NR.dll" /tlb:"C:\Program Files (x86)\Newton\NR2\NR2DLL\NRDLL\Newton.NR.tlb" /u /nologo
```

手順 2: VB6 の IDE で DLL を参照

上記の DLL の登録後、VB6 の IDE 上で次の通り参照設定を行います。

メニューバーからプロジェクト→参照設定ダイアログで、「Newton.NR」にチェックを入れると DLL が

使用可能となります。



Visual C++の場合

認証 UI ライブラリ(DLL)を Visual C++(以降 VC++)から利用する方法は次の通りです。

次の3つのファイルが必要です。

Newton.NR.dll

NewtonNRvcpp.dll

NewtonNRvcpp.tlb

これらは、認証レスキュー！2の「認証 UI ライブラリ」をインストールしてインストール先がデフォルトの場合、次のフォルダ内にあります。

<32bitOS の場合> C:\Program Files\Newton\NR2\NR2DLL\NRDLL\Framework4.0

<64bitOS の場合> C:\Program Files (x86)\Newton\NR2\NR2DLL\NRDLL\Framework4.0

手順 1: DLL の登録

次の通りコマンドプロンプトなどで DLL の登録(解除)を行います。

<DllPath>は、上記のフォルダパスを示します。

----- DLL の登録 -----

```
C:\WINDOWS\Microsoft.NET\Framework(または Framework64)\v4.0.30319\regasm.exe "<DllPath>NewtonNRvcpp.dll" /tlb:"<DllPath>NewtonNRvcpp.tlb" /codebase /nologo
```

(例:)<32bitOS の場合>

```
C:\WINDOWS\Microsoft.NET\Framework\v4.0.30319\regasm.exe "C:\Program Files\Newton\NR2\NR2DLL\NRDLL\Framework4.0\NewtonNRvcpp.dll" /tlb:"C:\Program Files\Newton\NR2\NR2DLL\NRDLL\Framework4.0\NewtonNRvcpp.tlb" /codebase /nologo
```

(例:)<64bitOS 上で 32bit(Win32,x86)EXE を実行する場合>

```
C:\WINDOWS\Microsoft.NET\Framework\v4.0.30319\regasm.exe "C:\Program Files (x86)\Newton\NR2\NR2DLL\NRDLL\Framework4.0\NewtonNRvcpp.dll" /tlb:"C:\Program Files (x86)\Newton\NR2\NR2DLL\NRDLL\Framework4.0\NewtonNRvcpp.tlb" /codebase /nologo
```

(例:)<64bitOS 上で 64bit(Win64,x64)EXE を実行する場合>

```
C:\WINDOWS\Microsoft.NET\Framework64\v4.0.30319\regasm.exe "C:\Program Files (x86)\Newton\NR2\NR2DLL\NRDLL\Framework4.0\NewtonNRvcpp.dll" /tlb:"C:\Program Files (x86)\Newton\NR2\NR2DLL\NRDLL\Framework4.0\NewtonNRvcpp.tlb" /codebase /nologo
```

また、登録した DLL を解除する場合は次の通りです。

----- DLL の解除 -----

```
C:\WINDOWS\Microsoft.NET\Framework(または Framework64)\v4.0.30319\regasm.exe "<DllPath>NewtonNRvcpp.dll" /tlb:"<DllPath>NewtonNRvcpp.tlb" /u /nologo
```

(例:)<32bitOS の場合>

```
C:¥WINDOWS¥Microsoft.NET¥Framework¥v4.0.30319¥regasm.exe "C:¥Program Files¥Newtone¥NR2¥NR2DLL¥NRDLL¥Framework4.0¥NewtoneNRvcpp.dll" /tlb:"C:¥Program Files¥Newtone¥NR2¥NR2DLL¥NRDLL¥Framework4.0¥NewtoneNRvcpp.tlb" /u /nologo
```

(例:)<64bitOS 上で 32bit(Win32,x86)EXE の実行を解除する場合>

```
C:¥WINDOWS¥Microsoft.NET¥Framework¥v4.0.30319¥regasm.exe "C:¥Program Files (x86)¥Newtone¥NR2¥NR2DLL¥NRDLL¥Framework4.0¥NewtoneNRvcpp.dll" /tlb:"C:¥Program Files (x86)¥Newtone¥NR2¥NR2DLL¥NRDLL¥Framework4.0¥NewtoneNRvcpp.tlb" /u /nologo
```

(例:)<64bitOS 上で 64bit(Win64,x64)EXE の実行を解除する場合>

```
C:¥WINDOWS¥Microsoft.NET¥Framework64¥v4.0.30319¥regasm.exe "C:¥Program Files (x86)¥Newtone¥NR2¥NR2DLL¥NRDLL¥Framework4.0¥NewtoneNRvcpp.dll" /tlb:"C:¥Program Files (x86)¥Newtone¥NR2¥NR2DLL¥NRDLL¥Framework4.0¥NewtoneNRvcpp.tlb" /u /nologo
```

手順 2: VC++用のタイプライブラリの Path をアプリケーション内で指定する

VC++アプリケーションから認証レスキュー！2 の VC++用の DLL を利用するためには、そのアプリケーション内で VC++用のタイプライブラリ(NewtoneNRvcpp.tlb)の Path を指定する必要があります。認証レスキュー！2 の VC++用のサンプルプロジェクトではタイプライブラリ(NewtoneNRvcpp.tlb)へのパスは次のファイル内に記述してありますのでご確認ください。

サンプルプロジェクト PackageApp → PackageApp.h

サンプルプロジェクト ProxyApp → ProxyAppDlg.h

(コード例)

```
// 認証レスキュー！2 の VC++用の tlb をインポート
#import "..¥¥¥¥¥NRDLL¥¥Framework4.0¥¥NewtoneNRvcpp.tlb" //raw_interfaces_only
//※当サンプルでは、raw_interfaces_only 属性は付与しない
```


■認証 UI ライブラリ (DLL) を利用したコーディング <UI 系の場合>

Visual Basic 2010 の場合

付属のサンプルプロジェクト (NR2DLL¥SampleProject¥VisualBasic2010¥PackageApp) の例で説明します。

最初に、Newtone.NR.Activation クラスのインスタンスを作成します。
次の例では、2つのFormで同一のインスタンスで使用するためクラスClass1内でPublic Sharedで宣言しています。

```
Public Class Class1
```

```
    Public Shared myActivate As Newtone.NR.Activation = New Newtone.NR.Activation()
```

```
End Class
```

次に DLL のプロパティを設定します。
これらのプロパティは通常、最初に 1 回だけ固定的に設定するコードを記述します。

```
Public Class Form1
```

```
    Private Sub Form1_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load
```

```
        '-----
        ' Newtone.NR.dllのプロパティの設定
        ' 【重要】DLLの以下のプロパティは必ず適切なものを設定してください。
        '-----

        ' ベンダ製品スタート開始レジストリキーパス (※最終的には必ず貴社のものに変更してください)
        Class1.myActivate.VendorsProductStartRegistryKeyPath =
"Software¥Newtone¥NinshoRescue¥NR-200¥SampleProject"
        ' 電話で認証時の電話番号 (※最終的には必ず貴社のものに変更してください)
        Class1.myActivate.TelephoneNumber = "012-345-6789"
        ' 暗号化時のパスワード (※最終的には必ず貴社のものに変更してください)
        Class1.myActivate.EncryptionPassword = "12345678ABCDEFGH"
        ' 暗号化時のSalt文字列(8バイト以上) (※最終的には必ず貴社のものに変更してください)
        Class1.myActivate.EncryptionSaltString = "認証レスキュー!"
        ' 猶予日数 (デフォルト: 0日、設定可能範囲: 1~365日) (※最終的には必ず貴社のものに変更してください)
        Class1.myActivate.TrialPeriod = 0
        ' 猶予期間の名称 (デフォルト: "猶予 (試用) 期間")
        Class1.myActivate.TrialPeriodName = "猶予 (試用)"
        ' レンタル日数 (デフォルト: 0日、設定可能範囲: 1~1100)
        Class1.myActivate.RentalPeriod = 0
        ' レンタル期間の名称 (デフォルト: "レンタル")
        Class1.myActivate.RentalPeriodName = "レンタル"
        ' MACアドレスの使用 (デフォルト: True) 「代理認証」利用時はMACアドレス必須
        Class1.myActivate.UseMacAddress = True
        ' CPU情報の使用 (デフォルト: True)
        Class1.myActivate.UseCpuInfo = True
        ' WebサービスのURL (※最終的には必ず貴社のものに変更してください)
        Class1.myActivate.WebServiceURL = "http://localhost/NR2WebService/Service.asmx"
        ' Webサービス時の基本認証の使用 (デフォルト: False)
        Class1.myActivate.WebServiceUseBasicAuthentication = False
```

```
' Webサービス時の基本認証ユーザ名
Class1.myActivate.WebServiceBasicAuthenticationUserName = ""
' Webサービス時の基本認証パスワード
Class1.myActivate.WebServiceBasicAuthenticationPassword = ""
' Webサービス確認パスワード (※最終的には必ず貴社のものに変更してください)
Class1.myActivate.WebServiceCheckPassword = "ABcd1234"
' プロダクトIDの桁数 (デフォルト: 17) (※最終的には必ず貴社のものに変更してください)
Class1.myActivate.ProductIdNumberOfDigits = 17
' シリアルNo.の桁数 (デフォルト: 8) (※最終的には必ず貴社のものに変更してください)
Class1.myActivate.SerialNoNumberOfDigits = 8
' Webサービスのタイムアウト (デフォルト: 60秒)
Class1.myActivate.WebServiceTimeout = 60
' 認証登録時の設定プロダクトID (デフォルト: 空文字列)
Class1.myActivate.SetProductID = ""
' 認証登録時の設定シリアルNo. (デフォルト: 空文字列)
Class1.myActivate.SetSerialNo = ""

CertificationStatus() ' 認証状況の確認
```

End Sub

...

End Class

後は、必要に応じてDLLのメソッドを呼び出します。

次の例では、ActivateStatusCheck()メソッドで「認証状態確認」を行います。

Public Class Form1

Private Sub CertificationStatus()

```
' -----
' 認証状況の確認
' -----
```

```
' Button1、Button2、Button3を含むGroupBox1のEnabledプロパティをFalseにして無効とする。
GroupBox1.Enabled = False
```

```
Dim ret As Integer
Dim stat As String ' 表示メッセージ
```

```
ret = Class1.myActivate.ActivateStatusCheck()
```

```
' 認証状況確認
```

```
Select Case ret
```

```
Case 0 ' 期限切れ (猶予有効時)
```

```
stat = "[ID:" + ret.ToString + "]" + Class1.myActivate.TrialPeriodName + "期限
が切れました。" + vbCr + "メニューは実行できません。" + vbCr + "「ライセンス管理」からライセンス
の認証登録を行ってください。"
```

```
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
```

```
Case 1 To 365 ' 猶予日数有
```

```
stat = "[ID:" + ret.ToString + "]" + Class1.myActivate.TrialPeriodName + "期間
残日数は" + ret.ToString + "日です。" + vbCr + "続行します。"
```

```
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
```

MessageBoxIcon.Information)

```
' 猶予（試用）残日数があるので、Button1、Button2、Button3を含む
' groupBox1のEnabledプロパティをTrueにして有効とする。
GroupBox1.Enabled = True
```

Case 400 ' 未認証（猶予無効時）

```
stat = "[ID:" + ret.ToString + "]" + "ライセンスが認証されていません。" + vbCr +
"メニューは実行できません。" + vbCr + "「ライセンス管理」からライセンスの認証登録を行ってください。"
```

```
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
```

Case 500 ' 認証済み

```
' 認証済みなので、Button1、Button2、Button3を含む
' groupBox1のEnabledプロパティをTrueにして有効とする。
GroupBox1.Enabled = True
```

Case 1000 ' レンタル期限切れ

```
stat = "[ID:" + ret.ToString + "]" + Class1.myActivate.RentalPeriodName + "期限
が切れました。" + vbCr + "一度、認証解除を行ってから" + vbCr + "新しいシリアルNo. を使って認証登
録を行ってください。"
```

```
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
```

Case 1001 To 2100 ' レンタル日数有

```
stat = "[ID:" + ret.ToString + "]" + Class1.myActivate.RentalPeriodName + "期間
残日数は" + (ret - 1000).ToString + "日です。" + vbCr + "続行します。"
```

```
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Information)
```

```
GroupBox1.Enabled = True
```

Case -999 ' 認証済ハードウェア情報不一致

```
stat = "[ID:" + ret.ToString + "]" + "認証済ですが認証時のハードウェア情報と一
致しない情報があります。"
```

```
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
```

Case -1 ' その他エラー

```
stat = "[ID:" + ret.ToString + "]" + "認証状況確認中に何らかのエラーが発生しま
した。"
```

```
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
```

Case 20000101 To 21001231 ' 「有効期限」以内

```
stat = "有効期限は、" + Mid(ret, 1, 4) + "/" + Mid(ret, 5, 2) + "/" + Mid(ret, 7,
2) + "までです。"
```

```
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Information)
```

```
GroupBox1.Enabled = True
```

Case -21001231 To -20000101 ' 「有効期限」以外

```
ret = ret * -1
stat = "有効期限は、" + Mid(ret, 1, 4) + "/" + Mid(ret, 5, 2) + "/" + Mid(ret, 7,
2) + "までです。"
```

```
stat = stat + vbCr + "有効期限が切れました。"
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
```

Case 366 ' 日付データの取得失敗（猶予）

```
stat = "日付データの取得に失敗しました。（猶予）"
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
```

Case 2101 ' 日付データの取得失敗（レンタル）

```

        stat = "日付データの取得に失敗しました。(レンタル)"
        MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
        MessageBoxIcon.Exclamation)
    Case -21001232 '日付データの取得失敗(有効期限)
        stat = "日付データの取得に失敗しました。(有効期限)"
        MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
        MessageBoxIcon.Exclamation)

    Case Else '想定外
        stat = "[ID:" + ret.ToString + "]" + "認証状況確認中に想定外のエラーが発生しま
        した"
        MessageBox.Show(stat)

    End Select

End Sub

...

End Class

```

次の例では、認証状況表示ダイアログを呼び出しています。

```

Public Class Form2

    Private Sub Button1_Click(sender As System.Object, e As System.EventArgs) Handles Button1.Click
        ' 認証状況表示
        If Class1.myActivate.ActivateStatusDisp() = False Then
            MessageBox.Show("エラー")
        End If
    End Sub

    ...

End Class

```

以下の例では、認証の登録や解除に関する各処理を呼び出しています。

```

Public Class Form2

    Private Sub Button2_Click(sender As System.Object, e As System.EventArgs) Handles Button2.Click

        ' 認証登録/インターネット
        If Class1.myActivate.ActivateRegisterInternet() = False Then
            MessageBox.Show("エラー")
        End If

    End Sub

    Private Sub Button3_Click(sender As System.Object, e As System.EventArgs) Handles Button3.Click

        ' 認証登録/電話
        If Class1.myActivate.ActivateRegisterTelephone() = False Then
            MessageBox.Show("エラー")
        End If

    End Sub

End Class

```

```

Private Sub Button4_Click(sender As System.Object, e As System.EventArgs) Handles Button4.Click
    ' 認証解除/インターネット
    If Class1.myActivate.ActivateRemoveInternet() = False Then
        MessageBox.Show("エラー")
    End If
End Sub

Private Sub Button5_Click(sender As System.Object, e As System.EventArgs) Handles Button5.Click
    ' 認証解除/電話
    If Class1.myActivate.ActivateRemoveTelephone() = False Then
        MessageBox.Show("エラー")
    End If
End Sub

Private Sub Button6_Click(sender As System.Object, e As System.EventArgs) Handles Button6.Click
    ' 代理認証登録/準備
    If Class1.myActivate.ProxyActivateRegisterPrepare() = False Then
        MessageBox.Show("エラー")
    End If
End Sub

Private Sub Button7_Click(sender As System.Object, e As System.EventArgs) Handles Button7.Click
    ' 代理認証登録/確定
    If Class1.myActivate.ProxyActivateRegisterFix() = False Then
        MessageBox.Show("エラー")
    End If
End Sub

Private Sub Button8_Click(sender As System.Object, e As System.EventArgs) Handles Button8.Click
    ' 代理認証解除/準備
    If Class1.myActivate.ProxyActivateRemovePrepare() = False Then
        MessageBox.Show("エラー")
    End If
End Sub

...

End Class

```

C# 2010 の場合

付属のサンプルプロジェクト(NR2DLL¥SampleProject¥CSharp2010¥PackageApp)の例で説明します。

最初に、Newtone.NR.Activationクラスのインスタンスを作成します。
次の例では、2つのFormで同一のインスタンスで使用するためクラスClass1内でpublic staticで宣言しています。

```
class Class1
{
    public static Newtowne.NR.Activation myActivate = new Newtowne.NR.Activation();
}
```

次に DLL のプロパティを設定します。
これらのプロパティは通常、最初に 1 回だけ固定的に設定するコードを記述します。

```
public partial class Form1 : Form
{
    private void Form1_Load(object sender, EventArgs e)
    {
        //-----
        //Newtone.NR.dllのプロパティの設定
        //【重要】DLLの以下のプロパティは必ず適切なものを設定してください。
        //-----

        //ベンダ製品スタート開始レジストリキーパス（※最終的には必ず貴社のものに変更してください）
        Class1.myActivate.VendorsProductStartRegistryKeyPath =
        "Software¥¥Newtone¥¥NinshoRescue¥¥NR-200¥¥SampleProject";
        //電話で認証時の電話番号（※最終的には必ず貴社のものに変更してください）
        Class1.myActivate.TelephoneNumber = "012-345-6789";
        //暗号化時のパスワード（※最終的には必ず貴社のものに変更してください）
        Class1.myActivate.EncryptionPassword = "12345678ABCDEFGH";
        //暗号化時のSalt文字列(8バイト以上)（※最終的には必ず貴社のものに変更してください）
        Class1.myActivate.EncryptionSaltString = "認証レスキュー！";
        //猶予日数（デフォルト：0日、設定可能範囲：1～365日）（※最終的には必ず貴社のものに変更してください）
        Class1.myActivate.TrialPeriod = 0;
        //猶予期間の名称（デフォルト：“猶予（試用）”）
        Class1.myActivate.TrialPeriodName = "猶予（試用）";
        //レンタル日数（デフォルト：0日、設定可能範囲：1～1100）
        Class1.myActivate.RentalPeriod = 0;
        //レンタル期間の名称（デフォルト：“レンタル”）
        Class1.myActivate.RentalPeriodName = "レンタル";
        //MACアドレスの使用（デフォルト：True）「代理認証」利用時はMACアドレス必須
        Class1.myActivate.UseMacAddress = true;
        //CPU情報の使用（デフォルト：True）
        Class1.myActivate.UseCpuInfo = true;
        //WebサービスのURL（※最終的には必ず貴社のものに変更してください）
        Class1.myActivate.WebServiceURL = "http://localhost/NR2WebService/Service.asmx";
        //Webサービス時の基本認証の使用（デフォルト：False）
        Class1.myActivate.WebServiceUseBasicAuthentication = false;
        //Webサービス時の基本認証ユーザ名
        Class1.myActivate.WebServiceBasicAuthenticationUserName = "";
        //Webサービス時の基本認証パスワード
        Class1.myActivate.WebServiceBasicAuthenticationPassword = "";
        //Webサービス確認パスワード（※最終的には必ず貴社のものに変更してください）
```

```

Class1.myActivate.WebServiceCheckPassword = "ABcd1234";
//プロダクトIDの桁数 (デフォルト:17) (※最終的には必ず貴社のものに変更してください)
Class1.myActivate.ProductIdNumberOfDigits = 17;
//シリアルNo.の桁数 (デフォルト:8) (※最終的には必ず貴社のものに変更してください)
Class1.myActivate.SerialNoNumberOfDigits = 8;
//Webサービスのタイムアウト (デフォルト:60秒)
Class1.myActivate.WebServiceTimeout = 60;
//認証登録時の設定プロダクトID (デフォルト:空文字列)
Class1.myActivate.SetProductID = "";
//認証登録時の設定シリアルNo. (デフォルト:空文字列)
Class1.myActivate.SetSerialNo = "";

CertificationStatus(); //認証状況の確認
}
...
}

```

後は、必要に応じてDLLのメソッドを呼び出します。

次の例では、ActivateStatusCheck()メソッドで「認証状態確認」を行います。

```

public partial class Form1 : Form
{
    private void CertificationStatus()
    {
        //-----
        //認証状況の確認
        //-----

        //Button1、Button2、Button3を含むGroupBox1のEnabledプロパティをFalseにして無効とする。
        GroupBox1.Enabled = false;

        int ret = 0;
        string stat = null;
        //表示メッセージ

        ret = Class1.myActivate.ActivateStatusCheck();

        //認証状況確認

        if (ret >= 1 && ret <= 365)
        {
            //猶予日数有
            stat = "[ID:" + ret.ToString() + "]" + Class1.myActivate.TrialPeriodName + "期間残
日数は" + ret.ToString() + "日です。" + "¥r" + "続行します。";
            MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK, MessageBoxIcon.Information);
            //猶予 (試用) 残日数があるので、Button1、Button2、Button3を含む
            //GroupBox1のEnabledプロパティをTrueにして有効とする。
            GroupBox1.Enabled = true;
            return;
        }

        if (ret >= 1001 && ret <= 2100)
        {

```

```

//レンタル日数有
stat = "[ID:" + ret.ToString() + "]" + Class1.myActivate.RentalPeriodName + "期間残
日数は" + (ret-1000).ToString() + "日です。" + "¥r" + "続行します。";
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK, MessageBoxIcon.Information);
GroupBox1.Enabled = true;
return;
}

if (ret >= 20000101 && ret <= 21001231)
{
//「有効期限」以内

String retStr = "";
retStr = ret.ToString();

stat = "有効期限は、" + retStr.Substring(0, 4) + "/" + retStr.Substring(4, 2) + "/" +
retStr.Substring(6, 2) + "までです。";
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK, MessageBoxIcon.Information);
GroupBox1.Enabled = true;
return ;
}

if (ret >= -21001231 && ret <= -20000101)
{
//「有効期限」以外

String retStr = "";
ret = ret * -1;
retStr = ret.ToString();

stat = "有効期限は、" + retStr.Substring(0, 4) + "/" + retStr.Substring(4, 2) + "/" +
retStr.Substring(6, 2) + "までです。";
stat = stat + Environment.NewLine + "有効期限が切れました。";
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
return;
}

switch (ret)
{
case 0:
//期限切れ（猶予有効時）
stat = "[ID:" + ret.ToString() + "]" + Class1.myActivate.TrialPeriodName + "期
限が切れました。" + "¥r" + "メニューは実行できません。" + "¥r" + "「ライセンス管理」からライセン
スの認証登録を行ってください。";
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);

break;
case 400:
//未認証（猶予無効時）
stat = "[ID:" + ret.ToString() + "]" + "ライセンスが認証されていません。" + "¥r"
+ "メニューは実行できません。" + "¥r" + "「ライセンス管理」からライセンスの認証登録を行ってくだ
さい。";
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);

break;
case 500:

```



```

//認証済み
//認証済みなので、Button1、Button2、Button3を含む
//GroupBox1のEnabledプロパティをTrueにして有効とする。
GroupBox1.Enabled = true;

break;
case 1000:
//レンタル期限切れ
stat = "[ID:" + ret.ToString() + "]" + Class1.myActivate.RentalPeriodName + "
期限が切れました。" + "¥r" + "一度、認証解除を行ってから" + "¥r" + "新しいシリアルNo.を使って認
証登録を行ってください。";
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);
break;

case -999:
//認証済ハードウェア情報不一致
stat = "[ID:" + ret.ToString() + "]" + "認証済ですが認証時のハードウェア情報と
一致しない情報があります。";
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);

break;
case -1:
//その他エラー
stat = "[ID:" + ret.ToString() + "]" + "認証状況確認中に何らかのエラーが発生し
ました。";
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);

break;

case 366:
//日付データの取得失敗（猶予）
stat = "日付データの取得に失敗しました。（猶予）";
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);

break;
case 2101:
//日付データの取得失敗（レンタル）
stat = "日付データの取得に失敗しました。（レンタル）";
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);

break;
case -21001232:
//日付データの取得失敗（有効期限）
stat = "日付データの取得に失敗しました。（有効期限）";
MessageBox.Show(stat, "認証確認", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);

break;

default:
//想定外
stat = "[ID:" + ret.ToString() + "]" + "認証状況確認中に想定外のエラーが発生し
ました";
MessageBox.Show(stat);

```

```

        break;
    }
}
...
}

```

次の例では、認証状況表示ダイアログを呼び出しています。

```

public partial class Form2 : Form
{
    private void Button1_Click(object sender, EventArgs e)
    {
        // 認証状況表示
        if (Class1.myActivate.ActivateStatusDisp() == false)
        {
            MessageBox.Show("エラー");
        }
    }
}

```

以下の例では、認証の登録や解除に関する各処理を呼び出しています。

```

public partial class Form2 : Form
{
    private void Button2_Click(object sender, EventArgs e)
    {
        //認証登録/インターネット
        if (Class1.myActivate.ActivateRegisterInternet() == false)
        {
            MessageBox.Show("エラー");
        }
    }

    private void Button3_Click(object sender, EventArgs e)
    {
        //認証登録/電話
        if (Class1.myActivate.ActivateRegisterTelephone() == false)
        {
            MessageBox.Show("エラー");
        }
    }

    private void Button4_Click(object sender, EventArgs e)
    {
        //認証解除/インターネット
        if (Class1.myActivate.ActivateRemoveInternet() == false)
        {
            MessageBox.Show("エラー");
        }
    }

    private void Button5_Click(object sender, EventArgs e)
    {
        //認証解除/電話
        if (Class1.myActivate.ActivateRemoveTelephone() == false)

```

```
        {  
            MessageBox.Show("エラー");  
        }  
    }  
  
    private void Button6_Click(object sender, EventArgs e)  
    {  
        //代理認証登録/準備  
        if (Class1.myActivate.ProxyActivateRegisterPrepare() == false)  
        {  
            MessageBox.Show("エラー");  
        }  
    }  
  
    private void Button7_Click(object sender, EventArgs e)  
    {  
        //代理認証登録/確定  
        if (Class1.myActivate.ProxyActivateRegisterFix() == false)  
        {  
            MessageBox.Show("エラー");  
        }  
    }  
  
    private void Button8_Click(object sender, EventArgs e)  
    {  
        //代理認証解除/準備  
        if (Class1.myActivate.ProxyActivateRemovePrepare() == false)  
        {  
            MessageBox.Show("エラー");  
        }  
    }  
    . . .  
}
```

Visual Studio 6.0 (Visual Basic 6.0) の場合

付属のサンプルプロジェクト(NR2DLL¥SampleProject¥VisualBasic6.0¥PackageApp)の例で説明します。

最初に、Newtone.NR.Activation クラスのインスタンスを作成します。
次の例では、2つの Form で同一のインスタンスで使用するため標準モジュール Module1 内で Public で宣言しています。

```
Public myActivate As New Newtone_NR.Activation
```

次に DLL のプロパティを設定します。
これらのプロパティは通常、最初に 1 回だけ固定的に設定するコードを記述します。

<Form1 内のコード>

```
Private Sub Form_Load()
```

```

'-----
' Newtone.NR.dll のプロパティの設定
' 【重要】DLL の以下のプロパティは必ず適切なものを設定してください。
'-----

' ベンダ製品スタート開始レジストリキーパス (※最終的には必ず貴社のものに変更してください)

myActivate.VendorsProductStartRegistryKeyPath = "Software¥Newtone¥NinshoRescue¥NR-200¥SampleProject"
' 電話で認証時の電話番号 (※最終的には必ず貴社のものに変更してください)
myActivate.TelephoneNumber = "012-345-6789"
' 暗号化時のパスワード (※最終的には必ず貴社のものに変更してください)
myActivate.EncryptionPassword = "12345678ABCDEFGH"
' 暗号化時の Salt 文字列 (8 バイト以上) (※最終的には必ず貴社のものに変更してください)
myActivate.EncryptionSaltString = "認証レスキュー!"
' 猶予日数 (デフォルト: 0 日、設定可能範囲: 1~365 日) (※最終的には必ず貴社のものに変更してください)
myActivate.TrialPeriod = 0
' 猶予期間の名称 (デフォルト: "猶予 (試用)")
myActivate.TrialPeriodName = "猶予 (試用)"
' レンタル日数 (デフォルト: 0 日、設定可能範囲: 1~1100)
myActivate.RentalPeriod = 0
' レンタル期間の名称 (デフォルト: "レンタル")
myActivate.RentalPeriodName = "レンタル"
' MAC アドレスの使用 (デフォルト: True) 「代理認証」利用時は MAC アドレス必須
myActivate.UseMacAddress = True
' CPU 情報の使用 (デフォルト: True)
myActivate.UseCpuInfo = True
' Web サービスの URL (※最終的には必ず貴社のものに変更してください)
myActivate.WebServiceURL = "http://localhost/NR2WebService/Service.asmx"
' Web サービス時の基本認証の使用 (デフォルト: False)
myActivate.WebServiceUseBasicAuthentication = False
' Web サービス時の基本認証ユーザ名
myActivate.WebServiceBasicAuthenticationUserName = ""
' Web サービス時の基本認証パスワード
myActivate.WebServiceBasicAuthenticationPassword = ""
' Web サービス確認パスワード (※最終的には必ず貴社のものに変更してください)

```

```

myActivate.WebServiceCheckPassword = "ABcd1234"
' プロダクト ID の桁数 (デフォルト: 17) (※最終的には必ず貴社のものに変更してください)
myActivate.ProductIdNumberOfDigits = 17
' シリアル No. の桁数 (デフォルト: 8) (※最終的には必ず貴社のものに変更してください)
myActivate.SerialNoNumberOfDigits = 8
' Web サービスのタイムアウト (デフォルト: 60 秒)
myActivate.WebServiceTimeout = 60
' 認証登録時の設定プロダクト ID (デフォルト: 空文字列)
myActivate.SetProductID = ""
' 認証登録時の設定シリアル No. (デフォルト: 空文字列)
myActivate.SetSerialNo = ""

```

```

CertificationStatus ' 認証状況の確認
End Sub

```

後は、必要に応じて DLL のメソッドを呼び出します。

次の例では、ActivateStatusCheck()メソッドで「認証状態確認」を行います。

<Form1 内のコード>

```

Private Sub CertificationStatus ()
' -----
' 認証状況の確認
' -----

' Command1、Command2、Command3、Frame1 の Enabled プロパティを False にして無効とする。
Frame1.Enabled = False
Command1.Enabled = False
Command2.Enabled = False
Command3.Enabled = False

Dim ret As Integer
Dim stat As String ' 表示メッセージ

ret = myActivate.ActivateStatusCheck ()

' 認証状況確認
Select Case ret
    Case 0 ' 期限切れ (猶予有効時)
        stat = "[ID:" + Str (ret) + "]" + myActivate.TrialPeriodName + "期限が切れました。"
+ vbCr + "メニューは実行できません。" + vbCr + "「ライセンス管理」からライセンスの認証登録を行っ
てください。"
        MsgBox stat, vbExclamation, "認証確認"

    Case 1 To 365 ' 猶予日数有
        stat = "[ID:" + Str (ret) + "]" + myActivate.TrialPeriodName + "期間残日数は" + Str
(ret) + "日です。" + vbCr + "続行します。"
        MsgBox stat, vbInformation, "認証確認"
        ' 猶予 (試用) 残日数があるので、Button1、Button2、Button3 を含む
        ' Frame1 の Enabled プロパティを True にして有効とする。
        Frame1.Enabled = True
        ' Command1、Command2、Command3 を有効にする。
        Command1.Enabled = True
        Command2.Enabled = True
        Command3.Enabled = True

```

```

Case 400 '未認証 (猶予無効時)
    stat = "[ID:" + Str(ret) + "]" + "ライセンスが認証されていません。" + vbCr + "メニューは実行できません。" + vbCr + "「ライセンス管理」からライセンスの認証登録を行ってください。"
    MsgBox stat, vbExclamation, "認証確認"

Case 500 '認証済み
    '認証済みなので、Button1、Button2、Button3を含む
    'Frame1のEnabledプロパティをTrueにして有効とする。
    Frame1.Enabled = True
    'Command1、Command2、Command3を有効にする。
    Command1.Enabled = True
    Command2.Enabled = True
    Command3.Enabled = True

Case 1000 'レンタル期限切れ
    stat = "[ID:" + Str(ret) + "]" + myActivate.RentalPeriodName + "期限が切れました。"
    + vbCr + "一度、認証解除を行ってから" + vbCr + "新しいシリアル No. を使って認証登録を行ってください。"
    MsgBox stat, vbExclamation, "認証確認"
Case 1001 To 2100 'レンタル日数有
    stat = "[ID:" + Str(ret) + "]" + myActivate.RentalPeriodName + "期間残日数は" + Str
    (ret - 1000) + "日です。" + vbCr + "続行します。"
    MsgBox stat, vbInformation, "認証確認"

    'Frame1のEnabledプロパティをTrueにして有効とする。
    Frame1.Enabled = True
    'Command1、Command2、Command3を有効にする。
    Command1.Enabled = True
    Command2.Enabled = True
    Command3.Enabled = True

Case -999 '認証済ハードウェア情報不一致
    stat = "[ID:" + Str(ret) + "]" + "認証済ですが認証時のハードウェア情報と一致しない
    情報があります。"
    MsgBox stat, vbExclamation, "認証確認"

Case -1 'その他エラー
    stat = "[ID:" + Str(ret) + "]" + "認証状況確認中に何らかのエラーが発生しました。"
    MsgBox stat, vbExclamation, "認証確認"

Case 20000101 To 21001231 '「有効期限」以内
    stat = "有効期限は、" + Mid(ret, 1, 4) + "/" + Mid(ret, 5, 2) + "/" + Mid(ret, 7, 2)
    + "までです。"
    MsgBox stat, vbExclamation, "認証確認"
    Frame1.Enabled = True
Case -21001231 To -20000101 '「有効期限」以外
    ret = ret * -1
    stat = "有効期限は、" + Mid(ret, 1, 4) + "/" + Mid(ret, 5, 2) + "/" + Mid(ret, 7, 2)
    + "までです。"
    stat = stat + vbCr + "有効期限が切れました。"
    MsgBox stat, vbExclamation, "認証確認"

Case 366 '日付データの取得失敗 (猶予)
    stat = "日付データの取得に失敗しました。(猶予)"
    MsgBox stat, vbExclamation, "認証確認"
Case 2101 '日付データの取得失敗 (レンタル)
    stat = "日付データの取得に失敗しました。(レンタル)"
    MsgBox stat, vbExclamation, "認証確認"
Case -21001232 '日付データの取得失敗 (有効期限)

```

```
stat = "日付データの取得に失敗しました。(有効期限)"
MsgBox stat, vbExclamation, "認証確認"
```

```
Case Else ' 想定外
stat = "[ID:" + Str(ret) + "]" + "認証状況確認中に想定外のエラーが発生しました"
MsgBox stat, vbExclamation, "認証確認"
```

```
End Select
```

```
End Sub
```

次の例では、認証状況表示ダイアログを呼び出しています。

<Form2 内のコード>

```
Private Sub Command1_Click()
' 認証状況表示
If myActivate.ActivateStatusDisp() = False Then
MsgBox "エラー"
End If
End Sub
```

以下の例では、認証の登録や解除に関する各処理を呼び出しています。

<Form2 内のコード>

```
Private Sub Command2_Click()

' 認証登録/インターネット
If myActivate.ActivateRegisterInternet() = False Then
MsgBox "エラー"
End If
```

```
End Sub
```

```
Private Sub Command3_Click()
```

```
' 認証登録/電話
If myActivate.ActivateRegisterTelephone() = False Then
MsgBox "エラー"
End If
```

```
End Sub
```

```
Private Sub Command4_Click()
```

```
' 認証解除/インターネット
If myActivate.ActivateRemoveInternet() = False Then
MsgBox "エラー"
End If
```

```
End Sub
```

```
Private Sub Command5_Click()
```

```
' 認証解除/電話
If myActivate.ActivateRemoveTelephone() = False Then
```

```
        MsgBox "エラー"  
    End If  
  
End Sub  
  
Private Sub Command6_Click()  
  
    '代理認証登録/準備  
    If myActivate.ProxyActivateRegisterPrepare() = False Then  
        MsgBox "エラー"  
    End If  
  
End Sub  
  
Private Sub Command7_Click()  
  
    '代理認証登録/確定  
    If myActivate.ProxyActivateRegisterFix() = False Then  
        MsgBox "エラー"  
    End If  
  
End Sub  
  
Private Sub Command8_Click()  
  
    '代理認証解除/準備  
    If myActivate.ProxyActivateRemovePrepare() = False Then  
        MsgBox "エラー"  
    End If  
  
End Sub
```


Visual C++の場合

付属のサンプルプロジェクト(NR2DLL¥SampleProject¥VC++2010¥PackageApp)の例で説明します。

最初に、NewtoneNRvcpp.IActivationクラスを使用するためにタイプライブラリのインポートをします。次の例では、2つのダイアログで同一のインスタンスで使用するためPackageApp.h内でタイプライブラリのインポート、クラスの宣言をしています。

認証レスキュー！2のVC++用タイプライブラリをインポートします。

```
#import "..¥¥..¥¥..¥¥NRDLL¥¥Framework4.0¥¥NewtoneNRvcpp.tlb"
```

認証レスキュー！2のVC++用タイプライブラリの名前空間を宣言します。

```
using namespace NewtoneNRvcpp;
```

```
class CPackageAppApp : public CWinApp
{
public:
    CPackageAppApp();
```

```
    . . .
```

```
    //認証レスキューActivationクラスのポインタの宣言
    IActivationPtr m_pActivation;
```

```
    . . .
```

```
};
```

```
//m_pActivationを各ダイアログから呼べるようにCPackageAppAppクラスの変数を宣言
extern CPackageAppApp theApp;
```

次にPackageAppDlg.cpp内のOnInitDialogイベント内でvcppActivationクラスのinterfaceポインタを作成してNewtoneNRvcpp.IActivationクラスのインスタンスを作成します。

次に DLL のプロパティを設定します。これらのプロパティは通常、最初に 1 回だけ固定的に設定するコードを記述するので PackageAppDlg.cpp 内の OnInitDialog イベント内に記述します。

```
BOOL CPackageAppDlg::OnInitDialog()
{
```

```
    . . .
```

```
    // COM の初期化
    HRESULT hr = CoInitialize(NULL);
```

```
    // vcppActivationクラスのinterfaceポインタを作る
    IActivationPtr pIAct(__uuidof(vcppActivation));
```

```
    // IActivationクラスのインスタンスを作成
    theApp.m_pActivation = pIAct;
```

```
    //-----
    //Newtone.NR.dllのプロパティの設定
```

```

// 【重要】DLLの以下のプロパティは必ず適切なものを設定してください。
//-----

//ベンダ製品スタート開始レジストリキーパス（※最終的には必ず貴社のものに変更してください）
theApp.m_pActivation->VendorsProductStartRegistryKeyPath =
"Software¥¥Newtone¥¥NinshoRescue¥¥NR-200¥¥SampleProject";
//電話で認証時の電話番号（※最終的には必ず貴社のものに変更してください）
theApp.m_pActivation->TelephoneNumber = "012-345-6789";
//暗号化時のパスワード（※最終的には必ず貴社のものに変更してください）
theApp.m_pActivation->EncryptionPassword = "12345678ABCDEFGH";
//暗号化時のSalt文字列(8バイト以上)（※最終的には必ず貴社のものに変更してください）
theApp.m_pActivation->EncryptionSaltString = "認証レスキュー!";
//猶予日数（デフォルト：0日、設定可能範囲：1~365日）（※最終的には必ず貴社のものに変更して
ください）
theApp.m_pActivation->TrialPeriod = 0;
//猶予期間の名称（デフォルト："猶予（試用）"）
theApp.m_pActivation->TrialPeriodName = "猶予（試用）";
//レンタル日数（デフォルト：0日、設定可能範囲：1~1100）
theApp.m_pActivation->RentalPeriod = 0;
//レンタル期間の名称（デフォルト："レンタル"）
theApp.m_pActivation->RentalPeriodName = "レンタル";
//MACアドレスの使用（デフォルト：True）「代理認証」利用時はMACアドレス必須
theApp.m_pActivation->UseMacAddress = true;
//CPU情報の使用（デフォルト：True）
theApp.m_pActivation->UseCpuInfo = true;
//WebサービスのURL（※最終的には必ず貴社のものに変更してください）
theApp.m_pActivation->WebServiceURL = "http://localhost/NR2WebService/Service.asmx";
//Webサービス時の基本認証の使用（デフォルト：False）
theApp.m_pActivation->WebServiceUseBasicAuthentication = false;
//Webサービス時の基本認証ユーザ名
theApp.m_pActivation->WebServiceBasicAuthenticationUserName = "";
//Webサービス時の基本認証パスワード
theApp.m_pActivation->WebServiceBasicAuthenticationPassword = "";
//Webサービス確認パスワード（※最終的には必ず貴社のものに変更してください）
theApp.m_pActivation->WebServiceCheckPassword = "ABcd1234";
//プロダクトIDの桁数（デフォルト：17）（※最終的には必ず貴社のものに変更してください）
theApp.m_pActivation->ProductIdNumberOfDigits = 17;
//シリアルNo.の桁数（デフォルト：8）（※最終的には必ず貴社のものに変更してください）
theApp.m_pActivation->SerialNoNumberOfDigits = 8;
//Webサービスのタイムアウト（デフォルト：60秒）
theApp.m_pActivation->WebServiceTimeout = 60;
//認証登録時の設定プロダクトID（デフォルト：空文字列）
theApp.m_pActivation->SetProductID = "";
//認証登録時の設定シリアルNo.（デフォルト：空文字列）
theApp.m_pActivation->SetSerialNo = "";

CertificationStatus(); //認証状況の確認

. . .

}

```

アプリケーションの終了時にCOMの終了処理をします。

//例：ダイアログの「×」ボタンで終了した時の処理

```
void CPackageAppDlg::OnClose()
```

```
{
```

```
    // TODO: ここにメッセージ ハンドラー コードを追加するか、既定の処理を呼び出します。
```

```

CDialogEx::OnClose();

// Uninitialize COM.
CoUninitialize();

}

```

後は、必要に応じてDLLのメソッドを呼び出します。

次の例では、ActivateStatusCheck()メソッドで「認証状態確認」を行います。

```

void CPackageAppDlg::CertificationStatus()
{
    //-----
    //認証状況の確認
    //-----

    //Button1、Button2、Button3を無効とする。
    Button1.EnableWindow(false);
    Button2.EnableWindow(false);
    Button3.EnableWindow(false);

    CString stat;
    //表示メッセージ

    long ret = theApp.m_pActivation->ActivateStatusCheck();
    CString strRet;
    strRet.Format(_T("%d"), ret);
    CString tpName = theApp.m_pActivation->TrialPeriodName;
    CString rpName = theApp.m_pActivation->RentalPeriodName;

    if (ret >= 1 && ret <= 365)
    {
        //猶予日数有
        stat = _T("[ID:") + strRet + _T("] ") + tpName + _T("期間残日数は") + strRet + _T("日です。¥r続行します。");
        MessageBox(stat, _T("認証確認"), MB_OK);

        //猶予（試用）残日数があるので、Button1、Button2、Button3を有効とする。
        Button1.EnableWindow(true);
        Button2.EnableWindow(true);
        Button3.EnableWindow(true);

        return;
    }

    if (ret >= 1001 && ret <= 2100)
    {
        //レンタル日数有
        CString strRet2;
        strRet2.Format(_T("%d"), ret - 1000);
        stat = _T("[ID:") + strRet + _T("] ") + rpName + _T("期間残日数は") + strRet2 + _T("日です。¥r続行します。");
        MessageBox(stat, _T("認証確認"), MB_OK);
        Button1.EnableWindow(true);
        Button2.EnableWindow(true);
    }
}

```

```

        Button3.EnableWindow(true);
        return;
    }

    if (ret >= 20000101 && ret <= 21001231)
    {
        // 「有効期限」内
        stat = _T("有効期限は") + strRet.Mid(0, 4) + _T("/") + strRet.Mid(4, 2) + _T("/") +
strRet.Mid(6, 2) + _T("までです。");
        MessageBox(stat, _T("認証確認"), MB_OK);
        Button1.EnableWindow(true);
        Button2.EnableWindow(true);
        Button3.EnableWindow(true);
        return;
    }

    if (ret >= -21001231 && ret <= -20000101)
    {
        // 「有効期限」外
        stat = _T("有効期限は") + strRet.Mid(0, 4) + _T("/") + strRet.Mid(4, 2) + _T("/") +
strRet.Mid(6, 2) + _T("までです。")
            + _T("¥r有効期限が切れました。");
        MessageBox(stat, _T("認証確認"), MB_OK);
        Button1.EnableWindow(true);
        Button2.EnableWindow(true);
        Button3.EnableWindow(true);
        return;
    }

    switch (ret)
    {
        case 0:
            //期限切れ (猶予有効時)
            stat = _T("[ID:") + strRet + _T("] ") + tpName + _T("期限が切れました。¥rメニュー
は実行できません。¥r「ライセンス管理」からライセンスの認証登録を行ってください。");
            MessageBox(stat, _T("認証確認"), MB_OK);

            break;

        case 400:
            //未認証 (猶予無効時)
            stat = _T("[ID:") + strRet + _T("] ライセンスが認証されていません。¥rメニューは実
行できません。¥r「ライセンス管理」からライセンスの認証登録を行ってください。");
            MessageBox(stat, _T("認証確認"), MB_OK);

            break;

        case 500:
            //認証済み
            //認証済みなので、Button1、Button2、Button3を有効とする。
            Button1.EnableWindow(true);
            Button2.EnableWindow(true);
            Button3.EnableWindow(true);

            break;

        case 1000:
            //レンタル期限切れ
            stat = _T("[ID:") + strRet + _T("] ") + rpName + _T("期限が切れました。¥r一度、認

```

```

証解除を行ってから¥r新しいシリアルNo. を使って認証登録を行ってください。");
    MessageBox(stat, _T("認証確認"), MB_OK);
    break;

    case -999:
        //認証済ハードウェア情報不一致
        stat = _T("[ID:") + strRet + _T("] 認証済ですが認証時のハードウェア情報と一致しな
い情報があります。");
        MessageBox(stat, _T("認証確認"), MB_OK);

        break;

    case -1:
        //その他エラー
        stat = _T("[ID:") + strRet + _T("] 認証状況確認中に何らかのエラーが発生しました。
");
        MessageBox(stat, _T("認証確認"), MB_OK);

        break;

    case 366:
        //日付データの取得失敗（猶予）
        stat = "日付データの取得に失敗しました。（猶予）";
        MessageBox(stat, _T("認証確認"), MB_OK);

        break;

    case 2101:
        //日付データの取得失敗（レンタル）
        stat = "日付データの取得に失敗しました。（レンタル）";
        MessageBox(stat, _T("認証確認"), MB_OK);

        break;

    case -21001232:
        //日付データの取得失敗（有効期限）
        stat = "日付データの取得に失敗しました。（有効期限）";
        MessageBox(stat, _T("認証確認"), MB_OK);

        break;

    default:
        //想定外
        stat = _T("[ID:") + strRet + _T("] 認証状況確認中に想定外のエラーが発生しました");
        MessageBox(stat);

        break;
    }
}

```

以下では諸々の具体的な使用法の例を紹介します。PackageAppサンプルでは、Dlg1ダイアログの各ボタンクリックで実行します。

次の例では、認証状況表示ダイアログを呼び出しています。

```
// 認証状況表示
```

```
voidDlg1::OnBnClickedButton1()
{
    if (theApp.m_pActivation->ActivateStatusDisp() == false)
        MessageBox(_T("エラー"));
}
```

以下の例では、認証の登録や解除に関する各処理を呼び出しています。

```
//認証登録/インターネット
voidDlg1::OnBnClickedButton2()
{
    if (theApp.m_pActivation->ActivateRegisterInternet() == false)
        MessageBox(_T("エラー"));
}

//認証登録/電話
voidDlg1::OnBnClickedButton3()
{
    if (theApp.m_pActivation->ActivateRegisterTelephone() == false)
        MessageBox(_T("エラー"));
}

//認証解除/インターネット
voidDlg1::OnBnClickedButton4()
{
    if (theApp.m_pActivation->ActivateRemoveInternet() == false)
        MessageBox(_T("エラー"));
}

//認証解除/電話
voidDlg1::OnBnClickedButton5()
{
    if (theApp.m_pActivation->ActivateRemoveTelephone() == false)
        MessageBox(_T("エラー"));
}

//代理認証登録/準備
voidDlg1::OnBnClickedButton6()
{
    if (theApp.m_pActivation->ProxyActivateRegisterPrepare() == false)
        MessageBox(_T("エラー"));
}

//代理認証登録/確定
voidDlg1::OnBnClickedButton7()
{
    if (theApp.m_pActivation->ProxyActivateRegisterFix() == false)
        MessageBox(_T("エラー"));
}

//代理認証解除/準備
voidDlg1::OnBnClickedButton8()
{
    if (theApp.m_pActivation->ProxyActivateRemovePrepare() == false)
        MessageBox(_T("エラー"));
}
```

```

//認証状態オンライン確認
void Dlg1::OnBnClickedButton9()
{
    long ret = theApp.m_pActivation->ActivateStatusCheckOnline();

    CString strRet;
    strRet.Format(_T("%d"), ret);

    CString msg = _T("戻り値は「") + strRet + _T("」でした。¥r¥r")
        + _T(" (0:PCレベルで認証されていない (PCのレジストリには認証登録情報がない)
¥r")
        + _T(" 1:OK (PCレベルとデータベースで認証登録情報が一致した) ¥r")
        + _T(" 2:NG (PCレベルと一致する認証登録情報がデータベースにない) ¥r")
        + _T(" 3:NG (認証済ハードウェア情報不一致) ¥r")
        + _T(" 4:NG (日付データの取得失敗 (猶予) ) ¥r")
        + _T(" 5:NG (日付データの取得失敗 (レンタル) ) ¥r")
        + _T(" 6:NG (日付データの取得失敗 (有効期限) ) ¥r")
        + _T(" -11:接続できない (認証時は電話) ¥r")
        + _T(" -12:接続できない (認証時は代理) ¥r")
        + _T("-999:その他エラー (接続できないなど) ¥r")
        + _T(" -1 : エラー (未設定や範囲を超えているプロパティがある) ");

    MessageBox(msg, _T("【認証状態オンライン確認メソッド】"));
}

//認証登録状態復元メソッド
void Dlg1::OnBnClickedButton10()
{
    if (theApp.m_pActivation->RestoreRegisterStatus() == false)
        MessageBox(_T("エラー"));
}

//認証解除状態復元メソッド
void Dlg1::OnBnClickedButton11()
{
    if (theApp.m_pActivation->RestoreCancelStatus() == false)
        MessageBox(_T("エラー"));
}

```

■認証 UI ライブラリ (DLL) の配布

・配布が必要なファイル

お客様(エンドユーザ)向けに配布するアプリケーションとともに次のファイルを配布する必要があります。この場合のアプリケーションは、認証 UI ライブラリ(DLL)を使用したアプリケーションを指します。

Newton.NR.dll

Newton.NR.tlb (Visual Basic 6.0 で作成したアプリケーションの配布時)

NewtonNRvcpp.dll (Visual C++で作成したアプリケーションの配布時)

NewtonNRvcpp.tlb (Visual C++で作成したアプリケーションの配布時)

これらは、認証レスキュー！2の「認証 UI ライブラリ」をインストールしてインストール先がデフォルトの場合、次のフォルダ内にあります。

.NET Framework4.0 用の DLL

<32bitOS の場合> C:\Program Files\Newton\NR2\NR2DLL\NRDLL\Framework4.0

<64bitOS の場合> C:\Program Files (x86)\Newton\NR2\NR2DLL\NRDLL\Framework4.0

.NET Framework3.5 用の DLL

<32bitOS の場合> C:\Program Files\Newton\NR2\NR2DLL\NRDLL\Framework3.5

<64bitOS の場合> C:\Program Files (x86)\Newton\NR2\NR2DLL\NRDLL\Framework3.5

※認証 UI ライブラリ(DLL)の動作には、.NET Framework4.0 または、Framework4.5 が必要です。

・お客様(エンドユーザ)PC 上での配置

認証 UI ライブラリ(DLL)をお客様(エンドユーザ)向けに配布するアプリケーションとともに配布する場合、インストーラなどでお客様(エンドユーザ)の PC 上でアプリケーション実行時にパスが通るように配置してください。

配布するアプリケーションと同じフォルダに配置するのが最も簡単な方法です。

<配布するアプリケーションが Visual Basic 6.0 で作成したアプリケーションの場合>

インストーラなどでお客様(エンドユーザ)の PC 上で DLL の登録も必要です。

詳しくは、

SampleProject\VisualBasic6.0 フォルダまたは SampleProject_API\VisualBasic6.0 フォルダの

【認証 UI ライブラリ(DLL)を VB6 から利用する方法】.txt

をご覧ください。

<配布するアプリケーションが Visual C++で作成したアプリケーションの場合>

インストーラなどでお客様(エンドユーザ)の PC 上で DLL の登録も必要です。

また、Visual C++で作成するアプリケーション内で、VC++用のタイプライブラリ(NewtoneNRvcpp.tlb)の Path をコードで指定する必要があります。

詳しくは、

SampleProject\VC++2010 フォルダまたは SampleProject_API\VC++2010 フォルダの

【認証 UI ライブラリ(DLL)を VC++から利用する方法】.txt

をご覧ください。

●認証レスキュー！で使う主なテーブルの概要

ここでは、「認証レスキュー！」で扱う SQL Server のテーブルを確認します。
 テーブルは、以下の 3 種類について説明します。

- ・<認証キーテーブル>
- ・<認証データテーブル>
- ・<認証ログテーブル>

以降でそれぞれを詳しく見て行きましょう。

<認証キーテーブル>

このテーブルは、ライセンス認証のキーとなるテーブルで、出荷する(アプリケーション)製品1本毎に1レコードが追加されます。

たとえば、出荷する製品の1本が4ライセンス製品であれば、ライセンス数には4と入力します。実際のレコード追加や削除は、社内システムの「認証キー作成」および「認証キー削除」処理で行います。

<認証キーテーブルの例>

| 製品の定義例 | | プロダクト ID | シリアル No. | ライセンス数 | プラス許可数 |
|------------------|----------|-------------------|----------|--------|--------|
| 製品名 | ライセンス | | | | |
| AAA システム Ver.2.0 | 1 ライセンス | 00001-00002-00001 | A01-0001 | 1 | 0 |
| 〃 | 〃 | 〃 | A01-0002 | 1 | 0 |
| 〃 | 〃 | 〃 | A01-0003 | 1 | 0 |
| 〃 | 4 ライセンス | 00001-00002-00004 | A04-0001 | 4 | 0 |
| BBB システム Ver.1 | 1 ライセンス | 00002-00001-00001 | B01-0001 | 1 | 0 |
| 〃 | 〃 | 〃 | B01-0002 | 1 | 0 |
| 〃 | 10 ライセンス | 00002-00001-00010 | B10-0001 | 10 | 0 |

・プラス許可数

これはライセンス認証登録済みのエンドユーザの PC がクラッシュなどに見舞われ、エンドユーザの PC ではライセンス認証解除をできないため、OS を再セットアップした PC や別の PC に、再度アプリケーションのライセンス認証登録ができない場合に使う特別な項目です。

初期値は 0 です。「認証レスキュー！」では、処理の入力項目ではありません。

認証業務用社内 PC の「認証管理システム」の「電話認証解除の対応」処理で、「クラッシュ」オプションを指定して認証解除を行った場合にこの「プラス許可数」項目が+1 されます。

この項目の必要性は、後で説明します。

<認証データテーブル>

このテーブルは、出荷した製品をエンドユーザがライセンス認証登録を行って成功した場合に1レコード追加されます。このテーブルのプロダクト ID とシリアル No.が同一なレコード数がそのまま現在のライセンス数になります。

たとえば、このテーブルに同一の「プロダクト ID とシリアル No.」の組み合わせで3レコード存在する

場合は、現在 3 ライセンス分の認証が登録済みであることとなります。

「認証レスキュー！」のライセンス認証のロジックでは、そのことを使用してライセンス数の上限を、前述の<認証キーテーブル>に登録した「ライセンス数」項目(と「プラス許可数」項目を足したもの)と比較しチェックしています。

このテーブルへの実際のレコード追加と削除は、次のタイミングで行われます。

インターネットで認証の場合、エンドユーザ PC で実行されるパッケージに組み込まれたプログラムからの呼び出しによる Web サービス内で行われます。

また、電話によるライセンス認証では社内システムの「電話認証登録の対応」、および「電話認証解除の対応」処理で行われます。

<認証データテーブルの例>

| 製品の定義例 | | プロダクト ID | シリアル No. | 認証 ID | ライセンスキー |
|------------------|----------|-------------------|----------|-------------|----------------|
| 製品名 | ライセンス | | | | |
| AAA システム Ver.2.0 | 1 ライセンス | 00001-00002-00001 | A01-0001 | 11111-1111 | ***** ***** |
| " | " | " | A01-0002 | 22222-22222 | ***** ***** |
| " | " | " | A01-0003 | 33333-33333 | ***** ***** |
| " | 4 ライセンス | 00001-00002-00004 | A04-0001 | 44444-44444 | ***** ***** |
| " | " | " | " | 55555-55555 | ***** ***** |
| " | " | " | " | 66666-66666 | ***** ***** |
| BBB システム Ver.1 | 1 ライセンス | 00002-00001-00001 | B01-0001 | 77777-77777 | ***** ***** |
| " | 10 ライセンス | 00002-00001-00010 | B10-0001 | 88888-88888 | ***** ***** |
| " | " | " | " | 99999-99999 | ***** ***** |

4 ライセンス分の 3 ライセンスが既に認証登録されている

・認証 ID

5 桁 - 5 桁 (数字のみ)

エンドユーザ PC 側の認証登録処理時に自動的に発生します。

・ライセンスキー

15 桁 (数字のみ)

上記の「認証 ID」とプロダクト ID、シリアル No.をもとに生成されます。

このライセンスキーは、インターネットでの認証の場合は、Web サービスで生成後エンドユーザ PC 側の認証登録処理(のプログラム)に返され、プログラムが自動的に処理します。

また、電話での認証登録時はオペレータに電話で伝えられたライセンスキーをエンドユーザが画面で入力して登録します。

ここでは、<認証キーテーブル>と<認証データテーブル>の両方のテーブルの項目説明が終わったところで、両テーブルに密接に関連する「プラス許可数」項目について説明します。

<認証キーテーブル>の「プラス許可数」項目の必要性

通常、ライセンス認証登録済みの PC がクラッシュすると、エンドユーザは製品に添付されているプロダクト ID とシリアル No.しかわかりません。認証解除には認証 ID も必要です。

たとえば、製品が 4PC ライセンスといった複数ライセンスの場合、プロダクト ID+シリアル No.は同一ですからこの情報だけでは、4 台中でクラッシュした個体 PC1 台を特定できません。

この場合、<認証データテーブル>には、同一のプロダクト ID+シリアル No.を持つレコードは最大で 4レコード存在することになりますが、認証 ID やライセンスキーが分からないのでどのレコードがクラッシュした PC 分なのか判定できません。

<認証キーテーブル>

| 製品の定義例 | | プロダクト ID | シリアル No. | ライセンス数 | プラス許可数 |
|------------------|---------|-------------------|----------|--------|--------|
| 製品名 | ライセンス | | | | |
| AAA システム Ver.2.0 | 1 ライセンス | 00001-00002-00001 | A01-0001 | 1 | 0 |
| " | 4 ライセンス | 00001-00002-00004 | A04-0001 | 4 | 0 |
| BBB システム Ver.1 | 1 ライセンス | 00002-00001-00001 | B01-0001 | 1 | 0 |

<認証データテーブル>

| 製品の定義例 | | プロダクト ID | シリアル No. | 認証 ID | ライセンスキー |
|------------------|---------|-------------------|----------|-------------|----------------|
| 製品名 | ライセンス | | | | |
| AAA システム Ver.2.0 | 1 ライセンス | 00001-00002-00001 | A01-0001 | 11111-1111 | ***** ***** |
| " | 4 ライセンス | 00001-00002-00004 | A04-0001 | 44444-44444 | ***** ***** |
| " | " | " | " | 55555-55555 | ***** ***** |
| " | " | " | " | 66666-66666 | ***** ***** |
| " | " | " | " | 12345-12345 | ***** ***** |
| BBB システム Ver.1 | 1 ライセンス | 00002-00001-00001 | B01-0001 | 77777-77777 | ***** ***** |

認証 ID が分からないと、これら 4 レコード分のどの PC 分(レコード)がクラッシュしたのか分からない

また消去法で考えたとして、クラッシュしていない残りの 3 台すべての PC の認証情報をエンドユーザから聞き出し煩雑な対応をすることは、エンドユーザはもちろん貴社にとっても得策ではありませんし、仮に 50 ライセンスだったらどうでしょう？ PC49 台分の認証情報をエンドユーザから教えてもらわねばなりません。

そこでこれらの解決方法として、前述の通り社内システムの「電話認証解除の対応」処理で、「クラッシュ」オプションを指定して認証解除を行い「プラス許可数」項目が+1 されると、「ライセンス数」+「プラス許可数」の合計が最大ライセンス数としてみなされ、未登録のライセンス数に 1 ライセンス分の猶予ができるわけです。

<認証キーテーブル>

| 製品の定義例 | | プロダクト ID | シリアル No. | ライセンス数 | プラス許可数 |
|------------------|---------|-------------------|----------|--------|--------|
| 製品名 | ライセンス | | | | |
| AAA システム Ver.2.0 | 1 ライセンス | 00001-00002-00001 | A01-0001 | 1 | 0 |
| 〃 | 4 ライセンス | 00001-00002-00004 | A04-0001 | 4 | 1 |
| BBB システム Ver.1 | 1 ライセンス | 00002-00001-00001 | B01-0001 | 1 | 0 |

合計 5 ライセンス (<認証 データ テーブル> 上での 5 レコード) まで登録可能となる

<認証データテーブル>

| 製品の定義例 | | プロダクト ID | シリアル No. | 認証 ID | ライセンスキー |
|------------------|---------|-------------------|----------|-------------|----------------|
| 製品名 | ライセンス | | | | |
| AAA システム Ver.2.0 | 1 ライセンス | 00001-00002-00001 | A01-0001 | 11111-1111 | ***** ***** |
| 〃 | 4 ライセンス | 00001-00002-00004 | A04-0001 | 44444-44444 | ***** ***** |
| 〃 | 〃 | 〃 | 〃 | 55555-55555 | ***** ***** |
| 〃 | 〃 | 〃 | 〃 | 66666-66666 | ***** ***** |
| 〃 | 〃 | 〃 | 〃 | 12345-12345 | ***** ***** |
| 〃 | 〃 | 〃 | 〃 | XXXXX-XXXXX | - |
| BBB システム Ver.1 | 1 ライセンス | 00002-00001-00001 | B01-0001 | 77777-77777 | ***** ***** |

その際、クラッシュした PC 分の<認証データテーブル>内のレコードは、以降活用されることのないデッドレコードとなりますが、システム上の問題はありません。

<認証ログテーブル>

このテーブルには、ライセンス認証に関わる様々な処理結果がログとして 1 レコードずつ記録されます。

このテーブルの内容は、社内システムの「ログの表示」処理で確認、また必要に応じて削除することができます。

<認証ログテーブルの例>

| 自動No. | 日時 | 処理 | ステータス | 認証区分 | メモ | プロダクトID | シリアルNo. | 認証ID | MACアドレス1 | CPU情報 |
|-------|---------------------|--------|--------|---------|----|-------------------|----------|-------------|--------------|--------------------------|
| 1 | 2014/02/24 15:38:00 | 1:キー作成 | 1:正常登録 | 1:オンライン | | 00001-00001-00001 | A0000001 | | E840F260C430 | Intel(R) Core(TM) i3-212 |
| 2 | 2014/02/24 15:38:00 | 1:キー作成 | 1:正常登録 | 1:オンライン | | 00001-00001-00001 | A0000002 | | E840F260C430 | Intel(R) Core(TM) i3-212 |
| 3 | 2014/02/24 15:38:00 | 1:キー作成 | 1:正常登録 | 1:オンライン | | 00001-00001-00001 | A0000003 | | E840F260C430 | Intel(R) Core(TM) i3-212 |
| 4 | 2014/02/24 15:39:00 | 1:キー作成 | 1:正常登録 | 1:オンライン | | 00001-00001-00001 | A0000004 | | E840F260C430 | Intel(R) Core(TM) i3-212 |
| 5 | 2014/02/24 15:39:00 | 1:キー作成 | 1:正常登録 | 1:オンライン | | 00001-00001-00001 | A0000005 | | E840F260C430 | Intel(R) Core(TM) i3-212 |
| 6 | 2014/03/24 15:42:00 | 3:認証登録 | 1:正常登録 | 1:オンライン | | 00001-00001-00001 | A0000001 | 50533-21818 | E840F260C430 | Intel(R) Core(TM) i3-212 |
| 7 | 2014/03/24 15:42:00 | 3:認証登録 | 1:正常登録 | 1:オンライン | | 00001-00001-00001 | A0000003 | 17958-26503 | E840F260C430 | Intel(R) Core(TM) i3-212 |
| 8 | 2014/03/24 15:42:00 | 3:認証登録 | 1:正常登録 | 1:オンライン | | 00001-00001-00001 | A0000002 | 78359-54685 | E840F260C430 | Intel(R) Core(TM) i3-212 |
| 9 | 2014/03/25 18:17:00 | 1:キー作成 | 1:正常登録 | 1:オンライン | | 12345-12345-12345 | 1234ABCD | | E840F260C430 | Intel(R) Core(TM) i3-212 |
| 10 | 2014/03/27 10:47:13 | 3:認証登録 | 1:正常登録 | 1:オンライン | | 12345-12345-12345 | 1234ABCD | 25672-67548 | E840F260C430 | Intel(R) Core(TM) i3-212 |
| 11 | 2014/03/27 10:47:18 | 4:認証解除 | 1:正常解除 | 1:オンライン | | 12345-12345-12345 | 1234ABCD | 25672-67548 | E840F260C430 | Intel(R) Core(TM) i3-212 |

・自動 No.

レコード追加時に自動的に付番されます。
初期値(シード)、増分(インクリメント)ともに1です。

・日時

ログが記録された日時です。

・処理区分

ログが書き込まれる際に、行われていた処理を示します。

・ステータスフラグ

ログが書き込まれる際に、その処理における結果状態を示します。

「処理区分」と「ステータスフラグ」の定義は次の通りです。

| 処理区分 | ステータスフラグ |
|----------------------|-----------|
| 1:キー作成 | 1:正常登録 |
| | 2:重複データ |
| | 3:何らかの原因 |
| | 11:同時実行違反 |
| 2:キー削除 | 1:正常削除 |
| | 2:該当キーなし |
| | 3:何らかの原因 |
| 3:認証登録 | 1:正常登録 |
| | 2:ライセンス超え |
| | 3:該当キーなし |
| | 4:何らかの原因 |
| | 5:重複データ |
| 4:認証解除 | 1:正常解除 |
| | 3:該当データなし |
| | 4:何らかの原因 |
| 5:キー作成 (自動ナンバリング) | 1:正常登録 |
| | 2:重複データ |
| | 3:何らかの原因 |
| | 11:同時実行違反 |
| 6:キー作成 (表形式) | 1:正常登録 |
| | 2:重複データ |
| | 3:何らかの原因 |
| | 11:同時実行違反 |

| | |
|---------------------|-----------|
| 7:キー編集 (表形式) | 1:正常登録 |
| | 2:重複データ |
| | 3:何らかの原因 |
| | 11:同時実行違反 |
| 8:キー削除 (選択削除) | 1:正常削除 |
| | 2:該当キーなし |
| | 3:何らかの原因 |
| 9:キー削除 (全行削除) | 1:正常削除 |
| | 2:該当キーなし |
| | 3:何らかの原因 |
| 10:クラッシュ解除 | 1:正常解除 |
| | 2:該当データなし |
| | 3:何らかの原因 |
| 11:キー作成 (インポート) | 1:正常登録 |
| | 2:重複データ |
| | 3:何らかの原因 |
| | 11:同時実行違反 |
| 12:キー作成 (ランダム生成) | 1:正常登録 |
| | 2:重複データ |
| | 3:何らかの原因 |
| | 11:同時実行違反 |
| 13:有効期限の更新 | 1:正常更新 |

・メモ

認証キー編集時の検索条件や認証登録時の有効期限などさまざまな情報を記録します。

・MAC アドレス

認証 UI ライブラリ(DLL)の UseMacAddress プロパティ、または APIUseMacAddress プロパティが True に設定されている場合、ログが書き込まれる際に、エンドユーザ PC の MAC アドレスを最大で 5 個記録します。

MAC アドレスは LAN ポートなどのネットワーク機器に固有な物理アドレスです。これを利用することでエンドユーザ PC の不正利用時の認証識別情報の精度を上げます。

・CPU 情報

認証 UI ライブラリ(DLL)の UseCpuInfo プロパティ、または APIUseCpuInfo プロパティが True に設定されている場合、ログが書き込まれる際に、エンドユーザ PC の CPU 情報を記録します。

これを利用することでエンドユーザ PC の不正利用時の認証識別情報の精度を上げます。

・IP アドレス

ログが書き込まれる際にエンドユーザ PC 側の(グローバル)IP アドレスを記録します。

なお、ローカル PC で Web サービスを実行している場合などの localhost を表す IP アドレスは、IPv4 で "127.0.0.1"、IPv6 で "::1" が記録されます。

● 「認証管理システム」のその他の処理説明

認証業務用社内 PC の「認証管理システム」の処理で先の初期設定で説明していない下図中の赤枠の処理について簡単に説明します。



※検索結果の表示上限数について

以降の各処理中の認証キーテーブルや認証データテーブル、およびログテーブルの検索機能では、検索条件の結果が 1 万行を超えるとメッセージが表示され検索条件を絞り込むように案内されます。

ただし、「認証状況」における認証データテーブルの検索結果の行数に関しては、Excel ファイルへの出力機能があるため制限はありません。

それでは、各処理を簡単に説明します。

■ 認証キー作成（表形式）

パッケージ出荷前に製品の認証キー情報を表形式で作成します。

行番号を押して一行選択し、[Delete]キーで削除することができます。

| | プロダクトID | シリアル No. | ライセンス数 | 有効期限利用 | 有効期限 (例: 2016/03/26) |
|------|---------|----------|--------|--------------------------|-------------------------|
| ▶▶ 1 | | | 1 | <input type="checkbox"/> | |

作成 認証キー一覧 終了

※作成した認証キー情報の一覧を表示します。

パッケージ出荷前に製品 1 本毎の認証キー情報を表形式で一度に複数作成できます。

「作成」ボタン:

表に入力された内容で、<認証キーテーブル>に追加します。

「認証キー一覧」ボタン:

登録されている<認証キーテーブル>の一覧を検索条件を指定して表示します。

■ 認証キー作成（個別）

パッケージ出荷前に製品 1 本毎の認証キー情報を作成します。

「作成」ボタン:

入力された内容で、<認証キーテーブル>に 1 レコード追加します。

「認証キー一覧」ボタン:

登録されている<認証キーテーブル>の一覧を検索条件を指定して表示します。

■ 認証キー作成（ランダム生成）

パッケージ出荷前に製品の認証キー情報を指定した数だけランダムに自動作成します。

プロダクトIDが同一で既に存在するシリアルNoは生成されません。

プロダクトID: ?

シリアルNo. ?

現在設定されているシリアルNo.の桁数は9桁です。

上位固定文字列 ?

ランダム指定

数字のみ

数字と英字(大文字)

数字と英字(小文字)

数字と英字(大小文字)

ナンバリング数 ?

ライセンス数: ?

有効期限設定 ?

有効期限を利用する 有効期限: 2016/03/26

作成 認証キー一覧 終了

※作成した認証キー情報の一覧を表示します。

パッケージ出荷前に製品の認証キー情報を指定した数だけランダムに自動生成します。

「上位固定文字列」の指定や「ランダム指定」の選択ができます。

「作成」ボタン:

入力された内容で、<認証キーテーブル>に追加します。

「認証キー一覧」ボタン:

登録されている<認証キーテーブル>の一覧を検索条件を指定して表示します。

■ 認証キー作成（インポート）

パッケージ出荷前に製品の認証キー情報をインポートして作成します。

PCにインストールされているExcelのバージョンによってインポートできるファイルが異なります。
(Excel2007以上: .xlsx / Excel2003以下: .xls)

2行目から読み込みます。
列は5列で、1列目はプロダクトID、2列目はシリアルNo.、3列目はライセンス数、4列目は有効期限利用、5列目は有効期限です。

複数のファイルを繰り返しインポートできますが、インポートした順に最下行に追加されます。
Excelのブックの1シート目のみがインポートの対象となります。

インポートするファイルを指定してください。

ファイル:

行番号を押して一行選択し、[Delete]キーで削除することができます。

| | プロダクトID | シリアル No. | ライセンス数 | 有効期限利用 | 有効期限 (例: 2016/03/26) |
|------|---------|----------|--------|--------------------------|-------------------------|
| ▶▶ 1 | | | 1 | <input type="checkbox"/> | |

※作成した認証キー情報の一覧を表示します。

パッケージ出荷前に製品の認証キー情報をインポートして作成します。
PC にインストールされている Excel のバージョンによってインポートできるファイルが異なります。
(Excel2007 以上: .xlsx / Excel2003 以下: .xls)
2 行目から読み込みます。
列は 5 列で、1 列目はプロダクト ID、2 列目はシリアル No.、3 列目はライセンス数、4 列目は有効期限利用、5 列目は有効期限です。4 列目の有効期限利用は「0: 利用しない、1: 利用する」として設定しておきます。
複数のファイルを繰り返しインポートできますが、インポートした順に最下行に追加されます。
Excel のブックの 1 シート目のみがインポートの対象となります。

「参照」ボタン:
インポートするファイルを選択します。

「インポート」ボタン:
インポートを実行します。

「作成」ボタン:
インポートされた内容で、<認証キーテーブル>に追加します。

「認証キー一覧」ボタン:
登録されている<認証キーテーブル>の一覧を検索条件を指定して表示します。

■ 認証キー編集（表形式）

製品の認証キー情報を検索して編集します。

検索

プロダクトID: ? (未入力:指定なし)

シリアルNo. 先頭指定文字列: ? (未入力:指定なし)

有効期限利用のみ表示

特定の有効期限のみ表示 2016/03/28 ~ 2016/03/28

検索実行

※既に認証データが存在する行や、入力不可項目はグレーで表示され編集はできません。
ただし、「有効期限」は更新のため変更可能です。

| | プロダクトID | シリアル No. | ライセンス数 | プラス許可数 | 有効期限利用 | 有効期限 (例: 2016/03/28) | 作成日時 |
|-----|-------------------|----------|--------|--------|-------------------------------------|----------------------------|---------------------|
| ▶ 1 | 00001-00001-00001 | A0000001 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 2 | 00001-00001-00001 | A0000002 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 3 | 00001-00001-00001 | A0000003 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 4 | 00001-00001-00001 | A0000004 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:39:00 |
| 5 | 00001-00001-00001 | A0000005 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:39:00 |
| 6 | 12345-12345-12345 | 1234ABCD | 5 | 0 | <input checked="" type="checkbox"/> | 2018/03/31 | 2014/03/25 18:17:00 |

有効期限の一括設定
上表内の認証キーすべての「有効期限」を一括設定する場合は、次の共有有効期限を指定して「一括設定」ボタンを押します。

共有有効期限: 2016/03/28

一括設定 登録 Excelファイル出力 終了

既に存在する認証キーを表形式で編集します。

※ライセンス形態が有効期限形式の場合で、特定のプロダクト ID とシリアル No に新しい有効期限を設定する場合は、当処理を利用します。

「検索実行」ボタン:

入力された検索条件で、＜認証キーテーブル＞の該当するレコードを表示します。

検索結果に対して表上で編集できます。

その際、既に(子データである)認証データが存在する認証キーの行や、入力不可項目はグレーで表示され編集はできません。

「登録」ボタン:

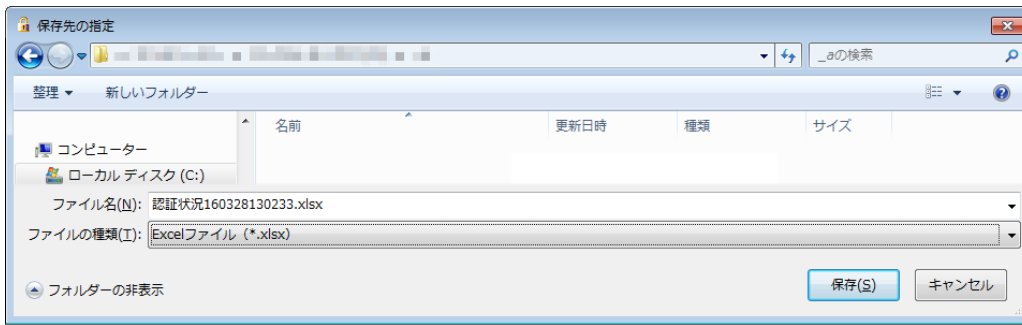
表に入力された内容で、＜認証キーテーブル＞を更新します。

「有効期限の一括設定」

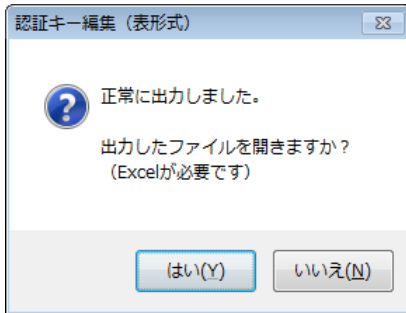
現在表示されている表内の認証キーすべての「有効期限」を一括で同日に設定できます。

「Excel ファイル出力」ボタン:

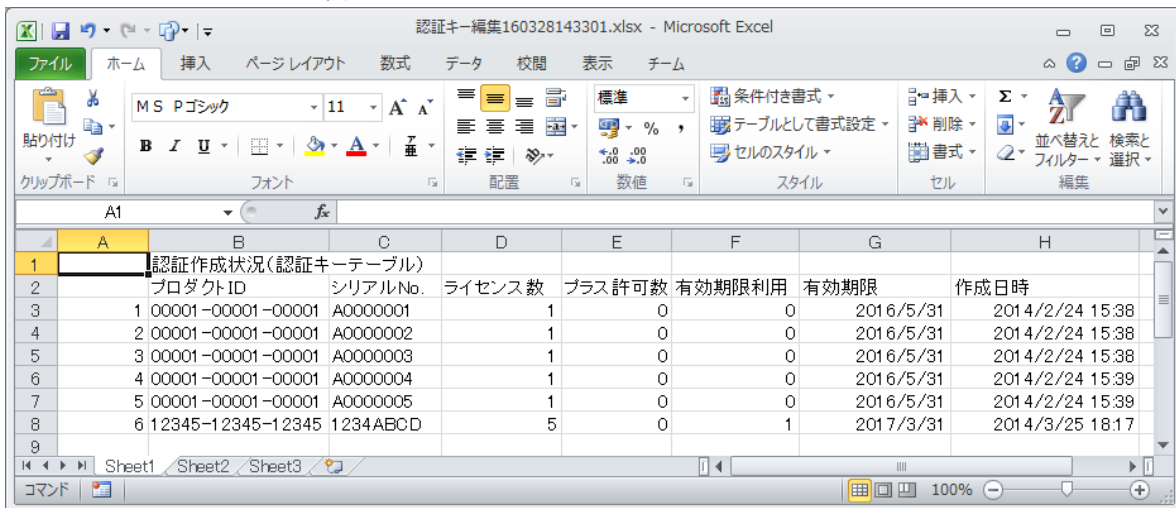
現在の表示内容を Excel ファイル(.xlsx)として出力します。ボタンを押すと出力先を指定するダイアログが表示されます。



出力した Excel ファイルを開くか選択します。



出力した Excel ファイルの例:



■ 認証キー削除（表形式）

認証キー削除（表形式）

製品の認証キー情報を検索して削除します。

検索

プロダクトID: ? (未入力:指定なし)

シリアルNo: 先頭指定文字列: ? (未入力:指定なし) 検索実行

| | プロダクトID | シリアルNo. | ライセンス数 | プラス許可数 | 有効期限利用 | 有効期限 (例: 2016/03/28) | 作成日時 |
|-----|-------------------|----------|--------|--------|-------------------------------------|----------------------------|---------------------|
| ▶ 1 | 00001-00001-00001 | A0000001 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 2 | 00001-00001-00001 | A0000002 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 3 | 00001-00001-00001 | A0000003 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 4 | 00001-00001-00001 | A0000004 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 5 | 00001-00001-00001 | A0000005 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 6 | 12345-12345-12345 | 1234ABCD | 5 | 0 | <input checked="" type="checkbox"/> | 2018/03/31 | 2014/03/25 18:17:00 |

選択削除
全行削除
終了

現在選択されている行を削除します。
検索で表示されている現在のデータをすべて削除します。

「検索実行」ボタン:

入力された検索条件で、＜認証キーテーブル＞の該当するレコードを表示します。
検索結果に対して表上で削除する行を指定します。

「選択削除」ボタン:

現在選択されている行を削除します。複数行の選択も可能です。

「全行削除」ボタン:

検索結果として表示されている現在のデータをすべて削除します。

■ 認証キー削除（個別）

作成した認証キーを削除します。

※入力した「プロダクトID+シリアルNo.」が認証データテーブルに存在する場合、そのレコードもすべて削除されます。

プロダクトID: ?

シリアルNo: ?

削除 認証キー一覧 終了

※作成した認証キー情報の一覧を表示します。

既存の認証キーを<認証キーテーブル>から削除します。
この際、入力した「プロダクト ID+シリアル No.」が(子データの)<認証データテーブル>に存在する場合は、そのレコードも削除されます。

「削除」ボタン：
入力された内容の認証キーを削除します。

「認証キー一覧」ボタン：
登録されている<認証キーテーブル>の一覧を検索条件を指定して表示します。

■ 認証状況

パッケージ出荷前に作成した認証キー情報と、現在のユーザーの認証登録状況を表示します。

検索

プロダクトID: ? (未入力: 指定なし)

シリアルNo: 先頭指定文字列: ? (未入力: 指定なし)

認証作成状況日付: 指定する 2016/03/28 ~ 2016/03/28

認証登録状況日付: 指定する 2016/03/28 ~ 2016/03/28

検索実行

認証作成状況(認証キーテーブル一覧)

| | プロダクトID | シリアルNo. | ライセンス数 | プラス許可数 | 有効期限利用 | 有効期限 (例: 2016/03/28) |
|-----|-------------------|----------|--------|--------|-------------------------------------|----------------------------|
| ▶ 1 | 00001-00001-00001 | A0000001 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 |
| 2 | 00001-00001-00001 | A0000002 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 |
| 3 | 00001-00001-00001 | A0000003 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 |
| 4 | 00001-00001-00001 | A0000004 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 |
| 5 | 00001-00001-00001 | A0000005 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 |
| 6 | 12345-12345-12345 | 1234ABCD | 5 | 0 | <input checked="" type="checkbox"/> | 2018/03/31 |

認証登録状況(認証データテーブル一覧)

| | プロダクトID | シリアルNo. | 認証ID | ライセンスキー | 作成日時 | MACアドレス 1 | MACアドレス 2 |
|-----|-------------------|----------|-------------|-----------------|---------------------|--------------|--------------|
| ▶ 1 | 00001-00001-00001 | A0000001 | 50583-21818 | 606126478578998 | 2014/03/24 15:42:00 | E840F260C430 | |
| 2 | 00001-00001-00001 | A0000002 | 17958-26503 | 660128081831738 | 2014/03/24 15:42:00 | E840F260C430 | |
| 3 | 00001-00001-00001 | A0000003 | 78359-54685 | 082259796100439 | 2014/03/24 15:42:00 | E840F260C430 | |
| 4 | 12345-12345-12345 | 1234ABCD | 28158-08530 | 281263669417894 | 2016/03/28 14:04:51 | BC5FF4B0E9DA | |

Excelファイル出力 終了

パッケージ出荷前に作成した認証キー情報(上表)と現在のエンドユーザーによるライセンスの認証登録状況(下表)を表示します。

「検索実行」ボタン:

入力された検索条件で、該当するデータを表示します。

「Excel ファイル出力」ボタン:

現在の表示内容を Excel ファイル(.xlsx)として出力します。ボタンを押すと出力先を指定するダイアログが表示されます。

保存先の指定

新しいフォルダー

名前 更新日時

コンピューター

ローカル ディスク (C:)

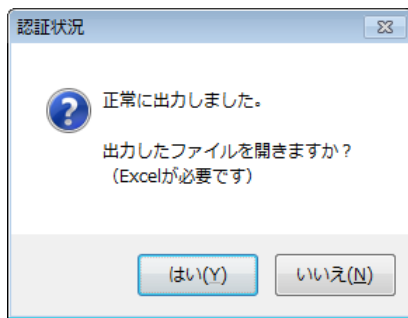
ファイル名(N): 認証状況140323220233.xlsx

ファイルの種類(I): Excelファイル (*.xlsx)

フォルダーの非表示

保存(S) キャンセル

出力した Excel ファイルを開くか選択します。



出力した Excel ファイルの例:

| | A | B | C | D | E | F | G | H | I | J | K | L |
|----|---|-------------------|----------|-------------|-----------------|-----------------|--------------|-----------------|------|------|------|---------|
| 1 | | 認証作成状況(認証キーテーブル) | | | | | | | | | | |
| 2 | | プロダクトID | シリアルNo. | ライセンス数 | プラス許可数 | 有効期限利用 | 有効期限 | 作成日時 | | | | |
| 3 | 1 | 00001-00001-00001 | A0000001 | 1 | 0 | 0 | 2016/5/31 | 2014/2/24 15:38 | | | | |
| 4 | 2 | 00001-00001-00001 | A0000002 | 1 | 0 | 0 | 2016/5/31 | 2014/2/24 15:38 | | | | |
| 5 | 3 | 00001-00001-00001 | A0000003 | 1 | 0 | 0 | 2016/5/31 | 2014/2/24 15:38 | | | | |
| 6 | 4 | 00001-00001-00001 | A0000004 | 1 | 0 | 0 | 2016/5/31 | 2014/2/24 15:39 | | | | |
| 7 | 5 | 00001-00001-00001 | A0000005 | 1 | 0 | 0 | 2016/5/31 | 2014/2/24 15:39 | | | | |
| 8 | 6 | 12345-12345-12345 | 1234ABCD | 5 | 0 | 1 | 2017/3/31 | 2014/3/25 18:17 | | | | |
| 9 | | | | | | | | | | | | |
| 10 | | | | | | | | | | | | |
| 11 | | 認証登録状況(認証データテーブル) | | | | | | | | | | |
| 12 | | プロダクトID | シリアルNo. | 認証ID | ライセンスキー | 作成日時 | MACアドレス1 | MACアドレス2 | MAC7 | MAC7 | MAC7 | CPU情報 |
| 13 | 1 | 00001-00001-00001 | A0000001 | 50533-21818 | 606126473573993 | 2014/3/24 15:42 | E840F260C430 | | | | | Intel(F |
| 14 | 2 | 00001-00001-00001 | A0000002 | 17958-26503 | 660128081831738 | 2014/3/24 15:42 | E840F260C430 | | | | | Intel(F |
| 15 | 3 | 00001-00001-00001 | A0000003 | 78359-54685 | 082259796100439 | 2014/3/24 15:42 | E840F260C430 | | | | | Intel(F |
| 16 | 4 | 12345-12345-12345 | 1234ABCD | 28158-08530 | 281263669417894 | 2016/3/28 14:04 | EC5FF4B0E9DA | | | | | Intel(F |
| 17 | | | | | | | | | | | | |

■ ログの表示

ログの表示

認証登録、認証解除、認証キー作成、認証キー削除時のログを表示します。

検索

日付: 指定する 2016/12/28 ~ 2016/12/28

プロダクトID: ? (未入力:指定なし)

シリアルNo.: 先頭指定文字列: ? (未入力:指定なし) 検索実行

ログの削除

全削除 日付指定 2016/12/28 分まで ログの削除実行 終了

| 自動No. | 作成日時 | 処理 | ステータス | 認証区分 | メモ | プロダクトID | シリアルNo. |
|-------|---------------------|-----------|--------|--------|------------|-------------------|----------|
| 7 | 2014/03/24 15:42:00 | 3認証登録 | 1:正常登録 | 1オンライン | | 00001-00001-00001 | A0000003 |
| 8 | 2014/03/24 15:42:00 | 3認証登録 | 1:正常登録 | 1オンライン | | 00001-00001-00001 | A0000002 |
| 9 | 2014/03/25 18:17:00 | 1キー作成 | 1:正常登録 | 1オンライン | | 12345-12345-12345 | 1234ABCD |
| 10 | 2016/12/02 17:44:44 | 3認証登録 | 1:正常登録 | 1オンライン | | 12345-12345-12345 | 1234ABCD |
| 11 | 2016/12/02 17:45:05 | 4認証解除 | 1:正常解除 | 1オンライン | | 12345-12345-12345 | 1234ABCD |
| 12 | 2016/12/03 17:25:12 | 3認証登録 | 1:正常登録 | 1オンライン | | 12345-12345-12345 | 1234ABCD |
| 13 | 2016/12/03 17:25:23 | 4認証解除 | 1:正常解除 | 1オンライン | | 12345-12345-12345 | 1234ABCD |
| 14 | 2016/12/19 14:59:27 | 7キー作成(編集) | 1:正常登録 | | 【検索条件】指定なし | - | - |
| 15 | 2016/12/19 15:00:03 | 7キー作成(編集) | 1:正常登録 | | 【検索条件】指定なし | - | - |
| 16 | 2016/12/19 15:00:54 | 7キー作成(編集) | 1:正常登録 | | 【検索条件】指定なし | - | - |
| 17 | 2016/12/21 15:07:04 | 3認証登録 | 1:正常登録 | 1オンライン | | 12345-12345-12345 | 1234ABCD |
| 18 | 2016/12/21 15:07:18 | 4認証解除 | 1:正常解除 | 1オンライン | | 12345-12345-12345 | 1234ABCD |
| 19 | 2016/12/22 14:49:41 | 3認証登録 | 1:正常登録 | 1オンライン | | 12345-12345-12345 | 1234ABCD |
| 20 | 2016/12/22 14:49:51 | 4認証解除 | 1:正常解除 | 1オンライン | | 12345-12345-12345 | 1234ABCD |
| 21 | 2016/12/22 14:54:20 | 3認証登録 | 1:正常登録 | 1オンライン | | 12345-12345-12345 | 1234ABCD |
| 22 | 2016/12/22 14:54:45 | 4認証解除 | 1:正常解除 | 1オンライン | | 12345-12345-12345 | 1234ABCD |

※「IPアドレス」IPv4で「127.0.0.1」、IPv6で「::1」はlocalhost

Excelファイル出力

認証登録、認証解除、認証キー作成、認証キー削除時のログを表示します。また、ログの削除もできます。

「検索実行」ボタン:

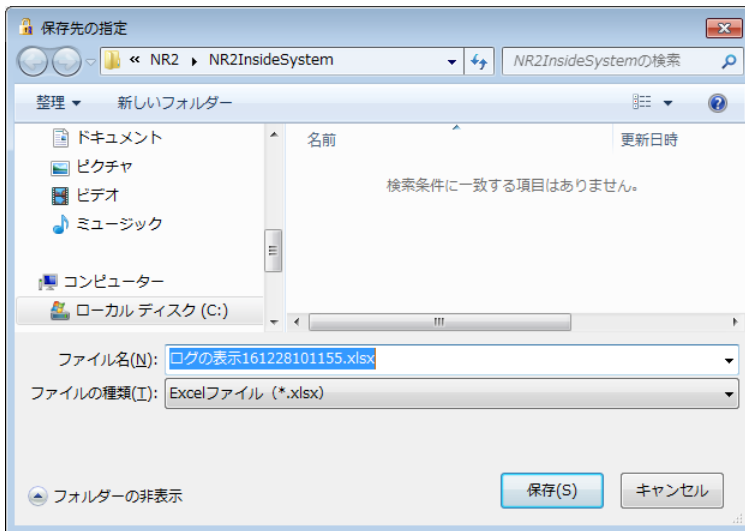
入力された検索条件で、該当するデータを表示します。

「ログの削除実行」ボタン:

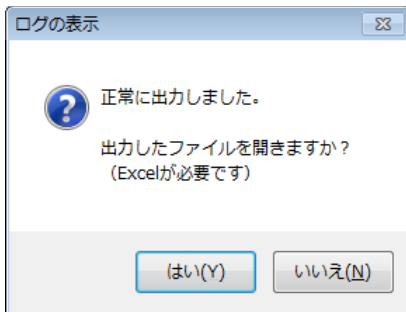
削除する条件を、「全削除」か「日付指定」から選択し、「日付指定」の場合はいつまでのログを削除するかを年月で指定後、この「ログの削除実行」ボタンを押すと指定した条件のログデータが削除されます。

「Excel ファイル出力」ボタン:

現在の表示内容を Excel ファイル(.xlsx)として出力します。ボタンを押すと出力先を指定するダイアログが表示されます。



出力した Excel ファイルを開くか選択します。



出力した Excel ファイルの例:

| 自動No. | 作成日時 | 処理 | ステータス | 認証区分 | メモ | プロダクトID | シリアルNo. | 認証ID | MACアドレス |
|-------|------------------|--------|--------|---------|--------------|-----------------|--------------------|-----------|-----------|
| 1 | 2014/2/24 15:38 | 1 キー作成 | 1 正常登録 | 1 オンライン | | 00001-00001-000 | A00000001 | | E840F260K |
| 2 | 2014/2/24 15:38 | 1 キー作成 | 1 正常登録 | 1 オンライン | | 00001-00001-000 | A00000002 | | E840F260K |
| 3 | 2014/2/24 15:38 | 1 キー作成 | 1 正常登録 | 1 オンライン | | 00001-00001-000 | A00000003 | | E840F260K |
| 4 | 2014/2/24 15:39 | 1 キー作成 | 1 正常登録 | 1 オンライン | | 00001-00001-000 | A00000004 | | E840F260K |
| 5 | 2014/2/24 15:39 | 1 キー作成 | 1 正常登録 | 1 オンライン | | 00001-00001-000 | A00000005 | | E840F260K |
| 6 | 2014/3/24 15:42 | 3 認証登録 | 1 正常登録 | 1 オンライン | | 00001-00001-000 | A00000001 | 50533-21E | E840F260K |
| 7 | 2014/3/24 15:42 | 3 認証登録 | 1 正常登録 | 1 オンライン | | 00001-00001-000 | A00000003 | 17958-26E | E840F260K |
| 8 | 2014/3/24 15:42 | 3 認証登録 | 1 正常登録 | 1 オンライン | | 00001-00001-000 | A00000002 | 78359-54E | E840F260K |
| 9 | 2014/3/25 18:17 | 1 キー作成 | 1 正常登録 | 1 オンライン | | 12345-12345-123 | 1234ABCE | | E840F260K |
| 10 | 2016/12/2 17:44 | 3 認証登録 | 1 正常登録 | 1 オンライン | | 12345-12345-123 | 1234ABCE77941-31E | | BC5FF4BC |
| 11 | 2016/12/2 17:45 | 4 認証解除 | 1 正常解除 | 1 オンライン | | 12345-12345-123 | 1234ABCE77941-31E | | BC5FF4BC |
| 12 | 2016/12/3 17:25 | 3 認証登録 | 1 正常登録 | 1 オンライン | | 12345-12345-123 | 1234ABCE04592-86E | | BC5FF4BC |
| 13 | 2016/12/3 17:25 | 4 認証解除 | 1 正常解除 | 1 オンライン | | 12345-12345-123 | 1234ABCE04592-86E | | BC5FF4BC |
| 14 | 2016/12/19 14:59 | 7 キー作成 | 1 正常登録 | | 【検索条件】指定なし | | | | BC5FF4BC |
| 15 | 2016/12/19 15:00 | 7 キー作成 | 1 正常登録 | | 【検索条件】指定なし | | | | BC5FF4BC |
| 16 | 2016/12/19 15:00 | 7 キー作成 | 1 正常登録 | | 【検索条件】指定なし | | | | BC5FF4BC |
| 17 | 2016/12/21 15:07 | 3 認証登録 | 1 正常登録 | 1 オンライン | | 12345-12345-123 | 1234ABCE29930-40E | | BC5FF4BC |
| 18 | 2016/12/21 15:07 | 4 認証解除 | 1 正常解除 | 1 オンライン | | 12345-12345-123 | 1234ABCE29930-40E | | BC5FF4BC |
| 19 | 2016/12/22 14:49 | 3 認証登録 | 1 正常登録 | 1 オンライン | | 12345-12345-123 | 1234ABCE47355-14E | | BC5FF4BC |
| 20 | 2016/12/22 14:49 | 4 認証解除 | 1 正常解除 | 1 オンライン | | 12345-12345-123 | 1234ABCE47355-14E | | BC5FF4BC |
| 21 | 2016/12/22 14:54 | 3 認証登録 | 1 正常登録 | 1 オンライン | | 12345-12345-123 | 1234ABCE16288-427E | | BC5FF4BC |
| 22 | 2016/12/22 14:54 | 4 認証解除 | 1 正常解除 | 1 オンライン | | 12345-12345-123 | 1234ABCE16288-427E | | BC5FF4BC |
| 23 | 2016/12/22 14:58 | 7 キー作成 | 1 正常登録 | | 【検索条件】指定なし | | | | BC5FF4BC |
| 24 | 2016/12/22 14:59 | 3 認証登録 | 1 正常登録 | 1 オンライン | 有効期限:2017/12 | 12345-12345-123 | 1234ABCE82565-70E | | BC5FF4BC |
| 25 | 2016/12/22 15:00 | 4 認証解除 | 1 正常解除 | 1 オンライン | | 12345-12345-123 | 1234ABCE82565-70E | | BC5FF4BC |
| 26 | 2016/12/22 15:10 | 3 認証登録 | 1 正常登録 | 1 オンライン | 有効期限:2017/12 | 12345-12345-123 | 1234ABCE22642-34E | | BC5FF4BC |
| 27 | 2016/12/22 15:10 | 4 認証解除 | 1 正常解除 | 1 オンライン | | 12345-12345-123 | 1234ABCE22642-34E | | BC5FF4BC |

■電話認証登録の対応

電話認証登録の対応

お客様がインターネットを使わずに電話でライセンス認証登録を依頼してきた場合の作業です。

① 以下の項目を(電話で)お客様から聞いて入力後、「登録」ボタンを押してください。

認証ID: ?

プロダクトID: ?

シリアルNo.: ?

登録

結果

ライセンスキー:

② 正常に登録された場合は結果に表示された「ライセンスキー」をお客様に伝えて、お客様画面のライセンスキーボックスに入力してもらい「登録」ボタンを押してもらいます。

終了

本処理(貴社の認証業務用 PC)の画面

電話で認証登録

インターネットを使わずに電話でライセンス認証登録を行います。

認証ID: 78285-13266

① 下記「プロダクトID」と「シリアルNo.」を入力します。

プロダクトID:

シリアルNo.:

② 0258-24-7900 に電話して「電話でのライセンス認証登録」を依頼してください。その後は電話担当者の指示に従ってください。

ライセンスキー:

登録

閉じる

エンドユーザ PC 上の画面

エンドユーザがインターネットを使わずに電話でライセンス認証登録を依頼してきた場合の作業です。

上記の項目を電話でユーザから聞いて入力し、「登録」ボタンを押します。その後、表示されたライセンスキーをエンドユーザに伝えて、エンドユーザ画面の「ライセンスキーボックス」に入力してもらい「登録」ボタンを押してもらいます。

※上図のように貴社の認証業務用の PC とエンドユーザ用の PC の画面を並べて表示すると、ユーザとの電話対応がイメージできます。

■電話認証解除の対応

・オフライン(ユーザの PC が動作している場合)

本処理(貴社の認証業務用 PC)の画面

お客様がインターネットを使わずに電話でライセンス認証解除を依頼してきた場合の作業です。

オフライン (お客様のPCが動作している場合) クラッシュ (お客様のPCが動作不能になった場合)

① 以下の項目について(電話で)お客様から聞いて入力後「解除キーの表示」ボタンを押します。

認証ID: ?

プロダクトID: ?

シリアルNo.: ?

解除キーの表示

解除キー:

② 次に、表示された「解除キー」をお客様に伝えて、お客様画面上の解除キーボックスに入力してもらい「解除」ボタンを押してもらいます。

③ お客様から「解除ステータス」を聞いて次の解除ステータスボックスに入力します。

解除ステータス:

④ 次の解除ボタンを押します。

解除

⑤ 正常に解除できた場合は、お客様に「閉じる」ボタンで処理を終了してもらいます。

① 以下の項目について(電話で)お客様から聞いて入力後「解除」ボタンを押してください。

プロダクトID: ?

シリアルNo.: ?

解除

② お客様に次のようにお話しください。

「こちらでクラッシュしたPCの解除を行いましたので電話の後でもう一度、認証登録を行ってください。」

終了

エンドユーザ PC 上の画面

インターネットを使わずに電話でライセンス認証解除を行います。

認証ID: 88167-21617

プロダクトID: 12345-12345-12345

シリアルNo.: 1234ABCD

① 0259-24-7900 に電話して「電話でのライセンス認証解除」を依頼してください。その後、電話担当者から聞いた「解除キー」を次のボックスに入力します。

解除キー: 956-11749-60711

② 次の「解除」ボタンを押してください。

解除

解除ステータス: 162-80797-49245

③ 上で表示された「解除ステータス」を電話担当者に伝えてください。

電話担当者に「解除ステータス」を伝えました。

閉じる

エンドユーザがインターネットを使わずに電話でライセンス認証解除を依頼してきた場合の作業です。

上記の項目を電話でエンドユーザから聞いて入力し、「解除キーの表示」ボタンを押します。その後、表示された解除キーをユーザに伝えて、エンドユーザ画面上の「解除キーボックス」に入力してもらい「解除」ボタンを押してもらいます。エンドユーザが正常に解除できたか確認したら社内用 PC 上の「解除」ボタンを押します。

「解除キー」、「解除ステータス」について

この解除キーと解除ステータスは、この処理でしか使用しません。また、データベースやエンドユーザ PC のレジストリにも保存しません。

解除キーはインターネット以外で認証解除をする場合に、必ず貴社に電話をかけさせるための手段として用意されています。

「電話認証解除の対応」の画面の「オフライン(ユーザの PC が動作している場合)」を見ると分かりますが、この解除キーの入力が無いとすると、「解除」ボタンを押すだけでエンドユーザが勝手に PC 上で認証解除ができてしまいます。この場合、エンドユーザ PC 上は認証解除状態になっても、貴社のデータベース上は認証解除にはなりません。情報のやり取りがないのですから当たり前です。そのままでは、再度エンドユーザが認証登録をしない場合に貴社のデータベースは解除されていないので登録は拒否されます。

そこでこのようなトラブルを避けるため、オフラインでの認証解除にはこの解除キーを必要とし、エンドユーザは貴社へ電話をして解除キーを聞かなければいけない仕組みになっているのです。

また、解除ステータスはエンドユーザがこの「電話で認証解除」を利用した不正ライセンスの流用防止のために使われます。

・クラッシュ(ユーザの PC が動作不能になった場合)

電話認証解除の対応

お客様がインターネットを使わずに電話でライセンス認証解除を依頼してきた場合の作業です。

オフライン (お客様のPCが動作している場合)
 クラッシュ (お客様のPCが動作不能になった場合)

① 以下の項目について(電話で)お客様から聞いて入力後「解除キーの表示」ボタンを押します。

認証ID: ?

プロダクトID: ?

シリアルNo.: ?

解除キーの表示

解除キー:

② 次に、表示された「解除キー」をお客様に伝えて、お客様画面上の解除キーボックスに入力してもらい「解除」ボタンを押してもらいます。

③ お客様が正常に解除できた場合は下の「解除」ボタンを押してください。

解除

① 以下の項目について(電話で)お客様から聞いて入力後「解除」ボタンを押してください。

プロダクトID: ?

シリアルNo.: ?

解除

② お客様に次のようにお話してください。
「こちらでクラッシュしたPCの解除を行いましたので電話の後でもう一度、認証登録を行ってください。」

終了

エンドユーザの PC がクラッシュしてしまい、OS などを入れなおした PC や別の PC に貴社のパッケージを再インストールする場合で、そのために以前の認証解除をしなければいけない状況への対応処理です。

この場合、エンドユーザはパッケージに添付されているプロダクト ID とシリアル No.しか分かりません。

この処理では、上記の項目を電話でエンドユーザから聞いて入力後、「解除」ボタンを押します。

前述の「<認証キーテーブル>の「プラス許可数」項目の必要性」に書きましたが、この処理によりデータベースには新たに認証登録できる猶予が作成されます。

●有効期限機能の利用方法

認証レスキュー！では、貴社はお客様(エンドユーザ)に対して通常の認証登録による無期限のライセンスとは別に、有効期限によるライセンスを設定することができます。
以下にその基本的な利用手順を示します。

■貴社側作業① - 認証キー作成処理時に有効期限を設定する

「認証管理システム」内の認証キー作成処理には、自動ナンバリング、表形式、個別、ランダム生成、インポートの5種類があります。

これらの認証キー作成処理を使用して出荷前に最初の有効期限を設定します。

各処理の画面には指定したプロダクトIDとシリアルNo.に対する「有効期限利用(する)」と「有効期限」の項目があります。

有効期限を設定するには、「有効期限利用(する)」をチェックして「有効期限」項目に有効期限としたい日付を設定します。

<認証キー作成(個別)処理の画面例>

上記の画面例では5ライセンス(マルチライセンス)に対して共通な有効期限を設定しています。

■エンドユーザ(お客様)側作業① - 認証登録を行う

エンドユーザに配布された貴社のアプリケーションから、通常の有効期限のない認証登録と同様に、認証UIライブラリ(DLL)を呼び出すことにより認証登録を実行してもらいます。

認証登録・解除の方法としてインターネットでの認証登録・解除、電話での認証登録・解除、代理認証登録・解除の3種類がありますが、有効期限によるライセンスの場合は「電話での認証登録・解除」は利用できませんので、ご注意ください。

<「インターネットで認証登録」の画面例>

<貴社のアプリケーションから「認証状況表示」を呼び出した場合の画面例>

下図のように、貴社が出荷前に「認証管理システム」の「認証キー作成」処理で設定した有効期限がエンドユーザの PC にも設定されます。

■貴社側作業② - 有効期限によるライセンスの有効期限更新に備え、新しい有効期限を設定する

有効期限によるライセンスで使用していたエンドユーザの有効期限が迫ってきていて、貴社とエンドユーザの間で継続して使用するライセンスを更新する契約が合意されたとします。

その場合、貴社はエンドユーザが使用しているプロダクトIDとシリアルNo.に対し新しい有効期限を設定する必要があります。

「認証管理システム」の「認証キー編集(表形式)」処理で新しい有効期限を設定します。

<認証キー編集(表形式)の画面例>

製品の認証キー情報を検索して編集します。

検索

プロダクトID: ? (未入力:指定なし)

シリアルNo: 先頭指定文字列: ? (未入力:指定なし)

有効期限利用のみ表示

特定の有効期限のみ表示 ~

※既に認証データが存在する行や、入力不可項目はグレーで表示され編集はできません。
ただし、「有効期限」は更新のため変更可能です。

| | プロダクトID | シリアル No. | ライセンス数 | プラス許可数 | 有効期限利用 | 有効期限 (例: 2016/03/28) | 作成日時 |
|-----|-------------------|----------|--------|--------|-------------------------------------|----------------------------|---------------------|
| 1 | 00001-00001-00001 | A0000001 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 2 | 00001-00001-00001 | A0000002 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 3 | 00001-00001-00001 | A0000003 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:38:00 |
| 4 | 00001-00001-00001 | A0000004 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:39:00 |
| 5 | 00001-00001-00001 | A0000005 | 1 | 0 | <input type="checkbox"/> | 2016/05/31 | 2014/02/24 15:39:00 |
| ▶ 6 | 12345-12345-12345 | 1234ABCD | 5 | 0 | <input checked="" type="checkbox"/> | 2017/03/31 | 2014/03/25 18:17:00 |

↓
2018/03/31に更新

有効期限の一括設定
上表内の認証キーすべての「有効期限」を一括設定する場合は、次の共通有効期限を指定して「一括設定」ボタンを押します。

共通有効期限:

上図の例では、プロダクト ID「12345-12345-12345」、シリアル No.「1234ABCD」のライセンスに対して、2017/03/31 までだった有効期限を新しく 2018/03/31 までと、一年間更新しています。この処理後は、貴社はエンドユーザに対し有効期限を延長したことを何らかの形で通知します。

■エンドユーザ(お客様)側作業② - 有効期限の更新を行う

エンドユーザは、貴社からの有効期限延長の通知を受け取ります。次にエンドユーザに、貴社のアプリケーションから認証 UI ライブラリ(DLL)を呼び出すことにより「有効期限の更新」を実行してもらいます。

この処理は、エンドユーザ PC が有効期限による認証済みでない場合は、実行することはできません。

<「有効期限の更新」の画面例>

下図のように、プロダクト ID、シリアル No.、現在の有効期限が自動的に表示されます。エンドユーザが「更新」ボタンを押すと、貴社が「認証管理システム」の「認証キー編集(表形式)」処理で設定した新しい有効期限が「新しい有効期限」として表示されます。

有効期限の更新

インターネットを経由して有効期限の更新を行います。

プロダクトID: 12345-12345-12345

シリアルNo: 1234ABCD

現在の有効期限: 2017/03/31

「更新」ボタンを押してください。

更新

新しい有効期限: 2018/03/31

開じる

プロキシサーバー

プロキシサーバーを使用する

アドレス:
(例: xxx.xxx.xxx.xxx)

ポート:
(例: 8080)

ユーザ名:
(必要時)

パスワード:
(必要時)

＜貴社のアプリケーションから「認証状況表示」を呼び出した場合の画面例＞
 下図のように、新しい有効期限がエンドユーザの PC にも設定されます。

認証状況表示

有効期限:
2018年03月31日まで

認証ID: 37033-50921

プロダクトID: 12345-12345-12345

シリアルNo: 1234ABCD

開じる

◆Web アプリケーション（ASP.NET）から有効期限の更新を行う

Ver.2.6.0 以降で利用できる ASP.NET (Web アプリケーション) 用の DLL には Web ページから「有効期限の更新」を実行できる、[APIxEditOfExpirationDate](#) メソッドが用意されています。それを利用すれば、貴社が作成した Web ページの UI からエンドユーザーに「有効期限の更新」を行わせることができます。

具体的な利用方法は、次の ASP.NET (Web アプリケーション) 用のサンプルプロジェクトをご覧ください。

- ・SampleProject_Web フォルダ (ASP.NET 系サンプルプロジェクト)

●アプリケーション難読化の必要性

ここでは、貴社が考慮すべき大変重要な事項について記載いたします。
それは、.NET アプリケーションの難読化の必要性です。

.NET 用に作成したアプリケーションのアセンブリ(.EXE や.DLL など)は、中間言語(IL)で作成されているので、簡単にリバースエンジニアリングをされてしまいます。

.NET アプリケーションの逆コンパイルは、無償の逆コンパイラなどを利用していても簡単に実行できます。逆コンパイルにより、単にコードの内容を知られるだけにとどまらず、パスワードなどのログイン情報や暗号化情報などの文字列も明らかになります。これらを放置するとその危険性がシステム全体におよび、貴社のソフトウェアビジネスに大きな損害を与えかねません。

これらのことは「認証レスキュー！」に関わらず一般的な問題なのですが、「認証レスキュー！」の場合を考えてみます。

最終的にエンドユーザに配布されるのは「認証レスキュー！」で提供される認証 UI ライブラリ(DLL)とそれを利用して作成した貴社のアプリケーションです。

この内、認証 UI ライブラリ(DLL)は難読化を施した上で弊社(ニュートン)より出荷されています。しかし、貴社のアプリケーションが Visual Basic(.NET)や C#で作成された場合は貴社サイドで難読化が必要となります。

貴社のアプリケーションを難読化すれば、悪意を持ったエンドユーザ(または第三者)が逆コンパイルしてもパスワードなどのログイン情報や暗号化情報などが露呈することはありません。

不正逆コンパイル対策のために弊社の難読化ツールなどで.NET アプリケーションを「難読化」されることを強く推奨いたします。

「難読化ツール」の詳細につきましては、[弊社の「Spices.NET JP」Web サイト](#)をご覧ください。

「認証レスキュー !2」ユーザーズガイド (2.6.8)

2014年3月24日 初版発行

2023年10月18日 第27版発行

NEWTONE
株式会社ニュートン

著者 株式会社ニュートン

発行所 株式会社ニュートン

新潟県長岡市本町 2-2-15 シャングリラ本町 1F

www.newtone.co.jp

Copyright © Newton Corporation

本書は、法律に定めのある場合または権利者の承諾のある場合を除いて、いかなる方法においても複製・複写することはできません。