

OLE オートメーションガイド

OLE オートメーションガイド

商標登録

Copyright © 2008 Symantec Corporation. All rights reserved.

Symantec、Symantec ロゴは Symantec Corporation または関連会社の米国およびその他の国における商標または登録商標です。その他の会社名、製品名は各社の商標または登録商標です。

このシマンテック製品には、サードパーティ（「サードパーティプログラム」）の所有物であることを示す必要があるサードパーティソフトウェアが含まれている場合があります。一部のサードパーティプログラムは、オープンソースまたはフリーソフトウェアライセンスで利用できます。本ソフトウェアに含まれる本使用許諾契約は、オープンソースまたはフリーソフトウェアライセンスでお客様が有する権利または義務は変更されないものとします。サードパーティプログラムについて詳しくは、この文書のサードパーティの商標登録の付属資料、またはこのシマンテック製品に含まれる TPIP ReadMe File を参照してください。

本書に記載されている製品は、その使用、コピー、頒布、逆コンパイルおよびリバースエンジニアリングを制限するライセンスに基づいて頒布されています。本書のいかなる部分も、Symantec Corporation およびそのライセンサーからの事前の文書による許諾を得ることなく、いかなる方法によっても無断で複写、複製してはならないものとします。

本書は、「現状のまま」提供されるものであり、Symantec Corporation は、商品価値を有すること、お客様の特定の目的にかなうこと、権利を侵害していないことに対する暗黙的な保証を含む、明示的あるいは暗黙的な条件、表明、および保証すべてから免責されるものとします。ただし、これらの免責が法的に無効であるとされる場合を除きます。SYMANTEC CORPORATION は、提供されるパフォーマンスまたは本書の使用に関連した付随的、または、結果的な損害に対して、一切責を負わないものとします。本書の内容は、事前の通知なく、変更される可能性があります。

ライセンス対象ソフトウェアおよび資料は、FAR 12.212 の規定によって商業用コンピュータソフトウェアと見なされ、FAR 52.227-19 「Commercial Computer Software - Restricted Rights」、DFARS 227.7202 「Rights in Commercial Computer Software or Commercial Computer Software Documentation」、その他の後継規則の規定により制限権利の対象となります。米国政府によるライセンス対象ソフトウェアおよび資料の使用、修正、複製のリリース、実演、表示、開示は、本使用許諾契約の条項に従ってのみ行われるものとします。

第 1 章	Symantec pcAnywhere における OLE オートメーションの使い方	5
	OLE オートメーションについて	5
	pcAnywhere Automation Server について	6
	pcAnywhere Automation Server でできること	6
	始める前に	7
	リモートエンジンの自動登録	7
	リモートエンジンの手動登録	7
	pcAnywhere Automation Server へのアクセス	8
	Visual Basic による pcAnywhere Automation Server へのアクセス	8
	Visual C++ による pcAnywhere Automation Server へのアクセス	9
	ホストとリモートの OLE オブジェクトの起動	11
	詳しい情報の参照先	11
第 2 章	Visual Basic のオブジェクト定義	13
	Visual Basic のオブジェクトについて	13
	CRemoteDataManager メソッド	14
	CRemoteData プロパティ	17
	接続の種類のプロパティ	19
	ダイヤルのプロパティ	20
	COM デバイスのプロパティ	21
	NetBIOS デバイスのプロパティ	24
	ISDN via CAPI 2.0 デバイスのプロパティ	25
	CRemoteDataEx オブジェクト	26
	リモート機能のための Visual Basic サンプルコード	27
	CHostDataManager メソッド	29
	CHostData プロパティ	31
	接続の種類のプロパティ	33
	AssignConnection(ConnectionType) メソッド	35
	UnassignConnection(ConnectionType) メソッド	36
	ダイヤルのプロパティ	37
	COM デバイスのプロパティ	37
	NetBIOS デバイスのプロパティ	40

ISDN via CAPI 2.0 デバイスのプロパティ 41

CHostDataEx オブジェクト 42

 ホスト機能のための Visual Basic サンプルコード 48

Awrem32 関数 50

第 3 章

Visual C++ のオブジェクト定義 53

Visual C++ オブジェクトについて 53

CRemoteDataManager メソッド 54

CRemoteData オブジェクト 57

 Get メソッドと Set メソッド 57

 リモートオブジェクトの詳細メソッド 60

 リモートオブジェクトのメソッド 62

CRemoteDataEx オブジェクト 64

 リモート機能のための Visual C++ サンプルコード 64

CHostDataManager メソッド 65

CHostData オブジェクト 69

 Get メソッドと Set メソッド 69

 ホストオブジェクトの詳細メソッド 70

 ホストオブジェクトのメソッド 74

CHostDataEx オブジェクト 76

 ホスト機能のための Visual C++ サンプルコード 78

Awrem32 関数 79

索引 81

Symantec pcAnywhere における OLE オートメーションの使い方

この章では以下の項目について説明しています。

- [OLE オートメーションについて](#)
- [pcAnywhere Automation Server について](#)
- [pcAnywhere Automation Server でできること](#)
- [始める前に](#)
- [詳しい情報の参照先](#)

OLE オートメーションについて

OLE オートメーションは、アプリケーションの中でエクスポートされた任意の機能の制御とオートメーションを行う外部アプリケーションや他の開発ツール（スクリプトやマクロなど）の作成を可能にする技術です。

OLE オートメーションは次のコンポーネントで構成されます。

OLE オートメーションサーバー 他のアプリケーションや開発ツールからアクセスまたは制御できるように自身の機能をエクスポートするアプリケーションやソフトウェアコンポーネントです。

pcAnywhere Automation Server は OLE オートメーションサーバーの一例です。

OLE オートメーションコントローラ OLE オートメーションサーバーによってエクスポートされたコンポーネントにアクセスして制御するアプリケーションや開発ツールです。

OLE オートメーションをサポートする任意のプログラミング言語を使えます。最も一般的なプログラミング言語は **Microsoft Visual Basic** と **Microsoft Visual C++** の 2 つです。

外部アプリケーションは、サーバーに接続してから、公開されたインターフェースの 1 つ以上へのアクセスを要求することによって OLE オートメーションサーバーにアクセスします。インターフェースは、関連する 1 つ以上のメソッドまたはプロパティへのアクセスを許可するエントリポイントです。アプリケーションがサーバーへのインターフェースを取得すると、任意の内部インターフェースメソッドを外部アプリケーションの一部であるかのように呼び出すことができます。

pcAnywhere Automation Server について

pcAnywhere Automation Server は、外部アプリケーションが pcAnywhere のリモートとホストのファイルを管理して、リモート制御とファイル転送のタスクのオートメーションを行うことを可能にします。pcAnywhere Automation Server は Symantec pcAnywhere ユーザーインターフェースのプログラミング可能な代替物として機能し、そのデフォルトの挙動の多くを模倣します。

たとえば、pcAnywhere でホストオブジェクトを作成すると、最初に利用可能なモデムの TAPI デバイスがデフォルトで割り当てられます。同様に、pcAnywhere Automation Server を使ってホストオブジェクトを作成してから割り当て済み接続のリストを通して見ると、最初に利用可能なモデムの TAPI デバイスがすでに割り当て済みになっています。

pcAnywhere Automation Server でできること

pcAnywhere Automation Server は、さまざまな管理性や生産性のタスクのオートメーションを可能にします。

pcAnywhere Automation Server を使うと次のことができます。

- ネットワークを通して複数のコンピュータにソフトウェアの更新を自動的に配布、インストールする
- 監査やアーカイブのために複数のコンピュータ間の自動的なファイル転送をスケジュール設定する
- ネットワーク上のすべての pcAnywhere ホストにある許可された呼び出し側リストで名前を自動的に追加または削除する

このマニュアルには、pcAnywhere Automation Server の接続方法と使い方を示すいくつかの例を、Visual Basic 版と Visual C++ 版の両方で収録しています。

p.13 の「[Visual Basic のオブジェクトについて](#)」を参照してください。

p.53 の「[Visual C++ オブジェクトについて](#)」を参照してください。

始める前に

pcAnywhere Automation Server とそのインターフェースへの接続中、GUID (Globally Unique Identifier) と呼ばれる識別子パラメータがオートメーションライブラリの API 関数に渡されます。pcAnywhere Automation Server とエクスポートされた各インターフェースには別個の GUID が割り当てられます。外部アプリケーションを pcAnywhere Automation Server とそのインターフェースに接続するには、これらの GUID がシステムレジストリの中に存在する必要があります。

Symantec pcAnywhere がインストールされたコンピュータで外部アプリケーションを実行する場合は、GUID エントリを自動的に登録できます。そうでない場合には GUID エントリを手動で登録する必要があります。

p.7 の「[リモートエンジンの自動登録](#)」を参照してください。

p.7 の「[リモートエンジンの手動登録](#)」を参照してください。

リモートエンジンの自動登録

OLE クライアントを使って別のコンピュータに初めて接続する前に、リモートエンジンを自分で登録する必要があります。この登録作業は、Symantec pcAnywhere でリモートオブジェクトを起動すると自動的に行われます。

リモートエンジンを自動的に登録するには

- 1 Symantec pcAnywhere を開くためには、次のいずれかの操作を行います。
 - デスクトップで Symantec pcAnywhere プログラムアイコンをダブルクリックします。
 - Windows のタスクバーで[スタート]、[プログラム]、[Symantec pcAnywhere]の順に選択します。
- 2 pcAnywhere マネージャウィンドウで[リモート]をクリックします。
- 3 リモート接続項目をダブルクリックします。

これでリモートエンジンが登録されます。接続を完了する必要はありません。

リモートエンジンの手動登録

外部アプリケーションを実行するコンピュータに pcAnywhere をインストールしていない場合、pcAnywhere Automation Server の実行可能ファイル (Winawsvr.exe) を実行して GUID を手動で登録する必要があります。この実行可能ファイルを 1 回実行するだ

けで GUID がレジストリに追加されます。Winawsvr.exe ファイルはインストール先ディレクトリにあります。

pcAnywhere Automation Server へのアクセス

OLE オートメーションをサポートする任意の言語プラットフォームを使って **pcAnywhere Automation Server** にアクセスできます。OLE オートメーションをサポートする最も一般的な言語プラットフォームは **Visual Basic** と **Visual C++** の 2 つです。

この 2 つのプラットフォームのコーディング原理は似ていますが、**Visual Basic** 環境では低水準の操作の多くが **Visual Basic** ランタイムシステムによってバックグラウンドで実行されます。

Visual Basic による pcAnywhere Automation Server へのアクセス

Visual Basic プログラミング言語は **pcAnywhere Automation Server** などの OLE オートメーションサーバーと通信を行うための組み込みサポートを持っています。**Standard Exe** プロジェクトを作成して、**pcAnywhere Automation Server** にアクセスするための各メソッドにコードを入力します。**Visual Basic** はソースファイルで高水準メソッド呼び出しを受け取り、その呼び出しを内部的に展開して対応する低水準 OLE オートメーションメソッド呼び出しに変換します。

Visual Basic で **pcAnywhere Automation Server** にアクセスするには

- 1 アクセスしたい **pcAnywhere** オブジェクトごとに一对の **Object** 変数を追加します。
たとえば、リモートオブジェクトを操作するときには、**RemoteDataManager** と **RemoteDataObject** を **Object** として **Dim** で定義します。
- 2 リモートオブジェクトのデータマネージャにアタッチするには **RemoteDataManager** を使います。
たとえば、パラメータとして **WINAWSVR.REMOTEDATAMANAGER** を指定して **CreateObject** メソッドを呼び出します。
Visual Basic はテキストのパラメータを使ってレジストリ内にあるマネージャの識別子を見つけ、そのマネージャにインターフェースを返します。
- 3 有効なデータマネージャオブジェクトが確立されれば、それを使って次のことが可能になります。
 - カレントディレクトリを決定する
 - 別のディレクトリに移動する
 - カレントディレクトリにある関連付けされたデータオブジェクトファイルを列挙する

- データオブジェクトファイルの作成、取得、削除を行う

- 4 データオブジェクトを作成または取得した後に、オブジェクトのプロパティの取得または設定が可能になります。

Visual Basic の構文は、プロパティの名前を使ってその値の取得と設定を区別することはありません。その代わり、代入演算子に対するプロパティの位置によって元になったメソッドの呼び出しが **Get** であるか **Set** であるかが決まります。

次に示すのは **Get** と **Set** の例です。

- オブジェクトの電話番号値を取得するには、そのプロパティ名を代入演算子の右に置きます。
たとえば、`s = RemoteData.PhoneNumber()` というようにします。s は文字列変数です。
- 電話番号を設定するには、そのプロパティ名を代入演算子の左に置きます。
たとえば、`RemoteData.PhoneNumber = "555-1212"` というようにします。

Visual C++ による pcAnywhere Automation Server へのアクセス

pcAnywhere Automation Server はタイプライブラリを使ってそのインターフェースとメソッドについての情報を **Visual C++** で書かれたオートメーションクライアントにエクスポートします。これらのタイプライブラリは **MFC (Microsoft Foundation Classes)** を使います。これは、**Visual C++** の **ClassWizard** を使ってアプリケーションにインポートできます。

タイプライブラリで提供されるデータマネージャクラスは **pcAnywhere Automation Server** へのインターフェースの取得に必要な機能を提供し、そのインターフェースの関連付けされたオブジェクト型で高水準の操作を実行します。

データマネージャオブジェクトを使うと次のことが可能になります。

- カレントディレクトリを決定または変更する
- カレントディレクトリにあるデータオブジェクトファイルをリストで列挙する
- 名前付きオブジェクトの作成、取得、削除を行う

オブジェクトが作成または取得されると、そのオブジェクトは関連付けされたデータオブジェクトクラスを使ってそのエクスポートされたプロパティのいずれかを試験または修正します。このプロパティのほとんどは **Get** または **Set** で始まる一対のメソッドを介してエクスポートされます。たとえば、オブジェクトの現在の電話番号のプロパティを試験するには **GetPhoneNumber** メソッドを呼び出し、そのプロパティを設定するには **SetPhoneNumber** を呼び出します。

クラスのインポートと表示

pcAnywhere Automation Server は次のタイプライブラリを使います。

Winawsvr.tlb pcAnywhere Automation Server に接続してそのインターフェースにアクセスするために必要な情報を提供する

Awrem32.tlb pcAnywhere 接続を制御するために必要な情報を提供する

次に説明するのは、**pcAnywhere Automation Server** タイプライブラリから **MFC** アプリケーションにクラス定義をインポートし、アプリケーションに追加されたクラスを表示する手順です。

クラスをインポートするには

- 1 Visual C++ で MFC アプリケーションを作成します。
- 2 表示メニューで **[ClassWizard]** を選択します。
- 3 **[MFC ClassWizard]** ダイアログボックスで、**[クラスの追加]**、**[タイプライブラリから]** の順にクリックします。
- 4 **winawsvr.tlb** をダブルクリックします。
- 5 **[クラスの確認]** ダイアログボックスで **[OK]** をクリックして、すべてのクラス定義をインポートします。
- 6 **[MFC ClassWizard]** ダイアログボックスで、**[クラスの追加]**、**[タイプライブラリから]** の順にクリックします。
- 7 **awrem32.tlb** をダブルクリックします。
- 8 **[クラスの確認]** ダイアログボックスで **[OK]** をクリックして、すべてのクラス定義をインポートします。
- 9 **[MFC ClassWizard]** ダイアログボックスで **[OK]** をクリックして、インポート処理を完了します。

これでアプリケーションにクラスが追加されました。これらのクラスを使ってオブジェクトの操作や接続の管理ができます。

タイプライブラリからクラス定義をインポートすると、サポートファイルがアプリケーションに追加されます。これらのファイルには **pcAnywhere Automation Server** のクラス定義と実装ソースコードが入っています。

p.10 の「[クラス定義と実装ファイルの表示](#)」を参照してください。

追加したクラスを表示するには

- 1 Visual C++ で MFC アプリケーションを開きます。
- 2 作業領域ウィンドウで **[ClassView]** タブをクリックします。

クラス定義と実装ファイルの表示

pcAnywhere Automation Server のタイプライブラリをアプリケーションにインポートすると、次のファイルが追加されます。

- Winawsvr.h
- Winawsvr.cpp
- Awrem32.h
- Awrem32.cpp

これらのファイルには **pcAnywhere Automation Server** のクラス定義と実装ソースコードが入っています。これらのファイルを編集する必要はありませんが、**pcAnywhere Automation Server** のメソッドの呼び出しを含む各アプリケーションソースファイルに **Winawsvr.h** をインクルードする必要があります。

クラス定義と実装ファイルを表示するには

- 1 Visual C++ で MFC アプリケーションを開きます。
- 2 作業領域ウィンドウで [FileView] タブをクリックします。

ホストとリモートの OLE オブジェクトの起動

Symantec pcAnywhere ではホストとリモートのすべてのオブジェクトをデフォルトのデータフォルダに格納する必要があります。OLE を使って作成したホストまたはリモートのオブジェクトを起動する前に、そのオブジェクトが **pcAnywhere** のデフォルトのデータフォルダにあることを確認してください。デフォルトフォルダは、Windows XP の場合は All Users、Windows Vista の場合は ProgramData です。

詳しい情報の参照先

OLE オートメーションについて詳しくは次の資料を参照してください。

- Box, Don.1998.『Essential COM』Reading, Mass.:Addison-Wesley.
- Horton, Ivor.1997.『Beginning MFC Programming』Birmingham, UK.:Wrox Press.
- Rogerson, Dale.1997.『Inside COM』Redmond, Wash.:Microsoft Press.
- Templeman, Julian.1997.『Beginning MFC COM Programming』Birmingham, UK.:Wrox Press.

Visual Basic のオブジェクト定義

この章では以下の項目について説明しています。

- Visual Basic のオブジェクトについて
- CRemoteDataManager メソッド
- CRemoteData プロパティ
- CRemoteDataEx オブジェクト
- CHostDataManager メソッド
- CHostData プロパティ
- CHostDataEx オブジェクト
- Awrem32 関数

Visual Basic のオブジェクトについて

pcAnywhere Automation Server は OLE オートメーションをサポートするために次のコンポーネントを提供します。

Winawsvr	pcAnywhere Automation Server に接続してそのインターフェースにアクセスするために必要な情報を提供する
Awrem32	pcAnywhere 接続を制御するために必要な情報を提供する

表 2-1 は Winawsvr を構成するオブジェクトの説明です。

表 2-1 Winawsvr のオブジェクト

オブジェクト	説明	参照先
CRemoteDataManager	CRemoteData オブジェクトの作成、オープン、修正、保存、削除のためのメソッドを提供します。	p.14 の「CRemoteDataManager メソッド」を参照してください。
CRemoteData	pcAnywhere リモート機能のアクセスと制御のためのパラメータを定義します。	p.17 の「CRemoteData プロパティ」を参照してください。 p.26 の「CRemoteDataEx オブジェクト」を参照してください。
CHostDataManager	CHostData オブジェクトの作成、オープン、修正、保存、削除のためのメソッドを提供します。	p.29 の「CHostDataManager メソッド」を参照してください。
CHostData	pcAnywhere ホスト機能のアクセスと制御のためのパラメータを定義します。	p.31 の「CHostData プロパティ」を参照してください。 p.42 の「CHostDataEx オブジェクト」を参照してください。

Awrem32 には 1 つのオブジェクトがあり、このオブジェクトはリモート制御とファイル転送のセッションをサポートする 8 つのインターフェースから構成されます。

p.50 の「Awrem32 関数」を参照してください。

一部の機能はすでにサポートされていません。ただし、pcAnywhere の以前のバージョンで使う場合に備えてオブジェクト定義は提供されています。

パスワードが必要な関数については、パスワード値は設定できますが取得はできません。これはセキュリティ上の理由によるものです。

CRemoteDataManager メソッド

CRemoteDataManager メソッドは CRemoteData オブジェクトにアクセスして制御するためのパラメータと戻り値を提供します。

表 2-2 CRemoteDataManager メソッド

CRemoteDataManager メソッド	パラメータまたは戻り値	説明
CurrentDirectory()	戻り値 = String	pcAnywhere リモートオブジェクトが格納されているカレントディレクトリの絶対パス名を返します。

CRemoteDataManager メソッド	パラメータまたは戻り値	説明
ChangeDirectory(NewDirectory)	パラメータ = NewDirectory	pcAnywhere リモートオブジェクトが格納されているカレントディレクトリを変更します。
FindFirst(Pattern)	パラメータ = Pattern (文字列)	カレントディレクトリで最初の pcAnywhere リモートオブジェクトファイル (*.chf) を見つけます。ファイル名パターンでは、指定されたファイル名パターンに基づいてフィルタ処理が行われます。アスタリスク (*) はカレントディレクトリのすべてのファイルを見つけてます。
FindFirst(Name string)	パラメータ = Name (文字列)	指定されたパターンに一致するリモートオブジェクトファイル (*.chf) の絶対パス名のバッファを返します。
FindNext(Name)	パラメータ = Name (文字列)	FindFirst() に対する元の呼び出しで指定したパターンに一致するリモートオブジェクトファイルの絶対パス名のバッファを返します。
FindNext(Name)	戻り値 = Boolean	FindFirst() の呼び出しで指定したパターンに一致する別のリモートオブジェクトファイルが見つかった場合に TRUE。一致するファイルの絶対パス名が Name に格納されます。
RetrieveObject(Name)	パラメータ = Name (文字列)	ロードするリモートオブジェクトファイルの完全修飾名です。
RetrieveObject(AccessMode)	パラメータ = AccessMode (整数)	このオブジェクトの使い方を指定します。これはパスワード保護に関係しません。 次のオプションがあります。 ■ 0 = 指定なし ■ 1 = 表示のみ ■ 2 = 表示と修正 ■ 3 = 実行
RetrieveObject>Password)	パラメータ = Password (文字列)	オブジェクトのパスワードです。NULL にできます。

CRemoteDataManager メソッド	パラメータまたは戻り値	説明
RetrieveObjectEx(Name)	パラメータ = Name (文字列)	ロードするリモートオブジェクトファイルの完全修飾名です。
RetrieveObjectEx(AccessMode)	パラメータ = AccessMode (整数)	このオブジェクトの使い方を指定します。これはパスワード保護に関係します。 次のオプションがあります。 ■ 0 = 指定なし ■ 1 = 表示のみ ■ 2 = 表示と修正 ■ 3 = 実行
RetrieveObjectEx(Password)	パラメータ = Password (文字列)	オブジェクトのパスワードです。NULLにできます。
RetrieveObjectEx	戻り値 = Object	指定されたファイルの CRemoteDataEx オブジェクトです。
CreateObject(Name)	パラメータ = Name (文字列)	CRemoteData オブジェクトを作成してそれを指す LPDISPATCH ポインタを返します。パラメータは新しいオブジェクトのリモートオブジェクトファイルの完全修飾名です。
CreateObject	戻り値 = Object	CRemoteData
CreateObjectEx(Name)	パラメータ = Name (文字列)	CRemoteDataEx オブジェクトを作成してそれを指す LPDISPATCH ポインタを返します。パラメータは新しいオブジェクトのリモートオブジェクトファイルの完全修飾名です。
CreateObjectEx	戻り値 = Object	CRemoteDataEx
DeleteObject(Name)	パラメータ = Name (文字列)	削除するオブジェクトのリモートオブジェクトファイルの完全修飾名です。
DeleteObject(Password)	パラメータ = Password (文字列)	オブジェクトのパスワードです。
DeleteObject	戻り値 = Boolean	オブジェクトが削除されれば TRUE。

CRemoteData プロパティ

CRemoteData オブジェクトで利用可能なプロパティとパラメータを以下に示します。

表 2-3 における三角カッコの中の情報は実際値に置き換えてください。

表 2-3 CRemoteData のプロパティとパラメータ

プロパティ	パラメータ	説明
<CRemoteData>.ComputerName(String)	String	ホストコンピュータのコンピュータ名または IP アドレスを設定します。
<CRemoteData>.PhoneNumber(String)	String	ホストコンピュータの電話番号を設定します。
<CRemoteData>.UseDialingProperties(Bool)	Bool	システムのダイヤルのプロパティを設定します。
<CRemoteData>.RedialCount(Integer)	Integer	呼び出しを中止するまでに行うリダイヤルの試行回数を設定します。
<CRemoteData>.RedialDelay(Integer)	Integer	リダイヤルの試行間の待ち時間(秒単位)を設定します。
<CRemoteData>.AutoLoginName(String)	String	自動ログインのためのユーザー名を設定します。 ドメインログインの使い方について詳しくは、 p.26 の「 CRemoteDataEx オブジェクト 」を参照してください。
<CRemoteData>.AutoLoginPassword(String)	String	自動ログインのためのパスワードをリモートオブジェクトに設定します。 セキュリティ上の理由から、 pcAnywhere Automation Server はパスワード値を読み込む機能を提供していません。パスワード値は返されません。
<CRemoteData>.Password(String)	String	ExecuteProtection 、 ReadProtection 、 WriteProtection の設定で使うリモートオブジェクトのパスワードを設定します。 セキュリティ上の理由から、 pcAnywhere Automation Server はパスワード値を読み込む機能を提供していません。パスワード値は返されません。
<CRemoteData>.ExecuteProtection(Bool)	Bool	オブジェクトを実行するためのパスワードの必要条件を設定します。 Password で設定します。

プロパティ	パラメータ	説明
<CRemoteData>.ReadProtection(Bool)	Bool	リモートオブジェクトのプロパティを表示するためのパスワードの必要条件を設定します。Password で設定します。
<CRemoteData>.WriteProtection(Bool)	Bool	リモートオブジェクトに対する変更を保存するためのパスワードの必要条件を設定します。Password で設定します。
<CRemoteData>.LogSession(Bool)	Bool	セッションのログ記録を有効または無効にします。
<CRemoteData>.RecordFile(String)	String	アクティブセッションを記録するファイルの場所のパスとファイルの完全修飾名を設定します。
<CRemoteData>.RecordSession(Bool)	Bool	自動セッション記録を有効または無効にします。
<CRemoteData>.ReadObject(String)	String	オブジェクトのパスワードを設定します。このプロパティはリモートオブジェクトのローカルデータコピーを更新するために使います。
<CRemoteData>.WriteObject(String)	String	オブジェクトのパスワードを設定します。このプロパティはリモートオブジェクトの作成またはリモートオブジェクトに対する変更の書き込みに使います。

CRemoteData のプロパティと戻り値を次の表に示します。

表 2-4 における三角カッコの中の情報は実際値に置き換えてください。

表 2-4 CRemoteData のプロパティと戻り値

プロパティ	戻り値	説明
String = <CRemoteData>.ComputerName	String	ホストコンピュータのコンピュータ名または IP アドレスを返します。
String = <CRemoteData>.PhoneNumber	String	ホストコンピュータの電話番号を返します。
Bool = <CRemoteData>.UseDialingProperties	Bool	リモートオブジェクトに設定されているシステムのダイヤルのプロパティを返します。
Integer = <CRemoteData>.RedialCount	Integer	リモートオブジェクトに設定されているリダイヤルの試行回数を返します。
Integer = <CRemoteData>.RedialDelay	Integer	リダイヤルの試行間の秒数を返します。
String = <CRemoteData>.AutoLoginName	String	自動ログインに使われるログイン名を返します。
Bool = <CRemoteData>.ExecuteProtection	Bool	ExecuteProtection 設定の値を返します。

プロパティ	戻り値	説明
Bool = <CRemoteData>.ReadProtection	Bool	ReadProtection 設定の値を返します。
Bool = <CRemoteData>.WriteProtection	Bool	WriteProtection 設定の値を返します。
Bool = <CRemoteData>.LogSession	Bool	セッションのログ記録が有効ならば TRUE を返します。
String = <CRemoteData>.RecordFile	String	セッション記録ファイルとパスの完全修飾名を返します。
Bool = <CRemoteData>.RecordSession	Bool	セッション記録設定の値を返します。

接続の種類のプロパティ

接続の種類のプロパティとパラメータの定義を次の表に示します。

表 2-5 における三角カッコの中の情報は実際値に置き換えてください。

表 2-5 接続の種類のプロパティとパラメータ

プロパティ	パラメータ	説明
<CRemoteData>.ConnectionType(String)	String	リモートの接続の種類を設定します。渡される値は FirstConnectionType() 関数と NextConnectionType() 関数で定義されている有効な接続の種類である必要があります。 有効な接続の種類の例を次に示します。 ■ COM1 ■ COM2 ■ COM3 ■ COM4 ■ TCP/IP ■ SPX ■ NetBIOS ■ ISDN via CAPI 2.0 (日本の環境ではご利用いただけません) ■ モデム名 (コンピュータに表示されるモデム名)

接続の種類のプロパティと戻り値の定義を次の表に示します。

表 2-6 における三角カッコの中の情報は実際値に置き換えてください。

表 2-6 接続の種類のプロパティと戻り値

プロパティ	戻り値	説明
String = <CRemoteData>.ConnectionTypes	String	リモートオブジェクトの接続の種類を返します。
Integer = <CRemoteData>.ConnectionType	Integer	利用可能な接続の種類の数を返します。 有効な接続の種類の例を次に示します。 ■ COM1 ■ COM2 ■ COM3 ■ COM4 ■ TCP/IP ■ SPX ■ NetBIOS ■ ISDN via CAPI 2.0 (日本の環境ではご利用いただけません) ■ モデム名 (コンピュータに表示されるモデム名)
String = <CRemoteData>.FirstConnectionType	String	最初に利用可能な接続の種類を返します。
String = <CRemoteData>.NextConnectionType	String	次に利用可能な接続の種類を返します。このプロパティは <CRemoteData>.ConnectionType に設定された接続の種類の数に対して順次呼び出され、すべての接続の種類が列挙されます。
Bool = <CRemoteData>.FindConnectionType(ConnectionType)	Bool	利用可能な接続の種類のリストの中で指定された接続の種類が見つかった場合は TRUE を返します。

ダイヤルのプロパティ

モデム接続のダイヤルプロパティを設定するためのプロパティとパラメータの定義を次の表に示します。

表 2-7 における三角カッコの中の情報は実際値に置き換えてください。

表 2-7 ダイヤルのプロパティを設定するためのプロパティとパラメータ

プロパティ	パラメータ	説明
<CRemoteData>.AreaCode(String)	String	モデム接続のダイヤルのプロパティとして市外局番を設定します。
<CRemoteData>.CountryCode(String)	String	モデム接続のダイヤルのプロパティとして国番号を設定します。

モデムのダイヤルのプロパティに使われるプロパティと戻り値の定義を次の表に示します。

表 2-8 における三角カッコの中の情報は実際値に置き換えてください。

表 2-8 ダイヤルのプロパティを設定するためのプロパティと戻り値

プロパティ	戻り値	説明
String = <CRemoteData>.AreaCode	String	ダイヤルのプロパティとして市外局番を返します。
String = <CRemoteData>.CountryCode	String	ダイヤルのプロパティとして国番号を返します。
Integer = <CRemoteData>.CountryCodes	Integer	利用可能な国番号の数を返します。
String = <CRemoteData>.FirstCountryCode	String	オペレーティングシステムのリストにある最初に利用可能な国番号を返します。
String = <CRemoteData>.NextCountryCode	String	次に利用可能な国番号を返します。これは <CRemoteData.CountryCodes> に設定された国番号の数に対して順に呼び出され、すべての国番号が列挙されます。

COM デバイスのプロパティ

モデムと他の COM ベースの接続のためのポート設定のカスタマイズを可能にするプロパティとパラメータの説明を次の表に示します。

表 2-9 における三角カッコの中の情報は実際値に置き換えてください。

表 2-9 COM デバイスのプロパティとパラメータ

プロパティ	パラメータ	説明
<CRemoteData>.ComParity(String)	String	通信パリティを設定します。 次の値が有効です。 ■ <None> ■ Odd ■ Even ■ Mark ■ Space

プロパティ	パラメータ	説明
<CRemoteData>.ComFlowControl(String)	String	COM ベースの接続の種類フロー制御を設定します。 次の値が有効です。 ■ <None> ■ XONXOFF ■ RTS/CTS ■ Both
<CRemoteData>.ComStartedBy(String)	String	COM ベースの接続の種類開始設定を設定します。 次の値が有効です。 ■ Always connected ■ Carrier detect (DCD) ■ Clear to send (CTS) ■ Data set ready (DSR) ■ Ring indicator (RI) ■ Receive 2 <CR>'s ■ Modem response
<CRemoteData>.ComEndedBy(String)	String	COM ベースの接続の種類終了設定を設定します。 次の値が有効です。 ■ Always connected ■ Carrier detect (DCD) ■ Clear to send (CTS) ■ Data set ready (DSR) ■ Ring indicator (RI)

プロパティ	パラメータ	説明
<CRemoteData>.ComSpeed(Long)	Long	最大 COM 速度の設定です。 次の値が有効です。 ■ 110 ■ 300 ■ 600 ■ 1200 ■ 2400 ■ 4800 ■ 9600 ■ 38400 ■ 57600 ■ 115200

COM デバイスのプロパティと戻り値を次の表に示します。

表 2-10 における三角カッコの中の情報は実際値に置き換えてください。

表 2-10 COM デバイスのプロパティと戻り値

プロパティ	戻り値	説明
String = <CRemoteData>.ComParity	String	通信パリティとして次の値のいずれかを返します。 ■ <None> ■ Odd ■ Even ■ Mark ■ Space
String = <CRemoteData>.ComFlowControl	String	リモートオブジェクトの Com Flow 設定を返します。 次の値が有効です。 ■ <None> ■ XONXOFF ■ RTS/CTS ■ Both

プロパティ	戻り値	説明
String = <CRemoteData>.ComStartedBy	String	COM ベースの接続の種類の Com Start 制御を返します。 次の値が有効です。 ■ Always connected ■ Carrier detect (DCD) ■ Clear to send (CTS) ■ Data set ready (DSR) ■ Ring indicator (RI) ■ Receive 2 <CR>'s ■ Modem response
String = <CRemoteData>.ComEndedBy	String	COM ベースの接続の種類の Com End 制御を返します。 次の値が有効です。 ■ Always connected ■ Carrier detect (DCD) ■ Clear to send (CTS) ■ Data set ready (DSR) ■ Ring indicator (RI)
Long = <CRemoteData>.ComSpeed	Long	リモートオブジェクトの Com Speed の現在の設定を返します。 次の値が有効です。 ■ 110 ■ 300 ■ 600 ■ 1200 ■ 2400 ■ 4800 ■ 9600 ■ 38400 ■ 57600 ■ 115200

NetBIOS デバイスのプロパティ

NetBIOS ネットワークデバイスのプロパティとパラメータの定義を次の表に示します。

表 2-11 における三角カッコの中の情報は実際値に置き換えてください。

表 2-11 NetBIOS のプロパティとパラメータ

プロパティ	パラメータ	説明
<CRemoteData>.LanaNumber(Integer)	Integer	NetBIOS 接続の LAN アダプタ (LANA) 番号を設定します。

NetBIOS のプロパティと戻り値の定義を次の表に示します。

表 2-12 における三角カッコの中の情報は実際値に置き換えてください。

表 2-12 NetBIOS のプロパティと戻り値

プロパティ	戻り値	説明
Integer = <CRemoteData>.LanaNumber	Integer	NetBIOS 接続の LAN アダプタ (LANA) 番号の現在の設定を返します。

ISDN via CAPI 2.0 デバイスのプロパティ

European ISDN 接続のプロパティとパラメータの定義を次の表に示します。

表 2-13 における三角カッコ内の情報は実際の値に置き換えてください。

表 2-13 ISDN via CAPI 2.0 のプロパティとパラメータ

プロパティ	パラメータ	説明
<CRemoteData>.CapiChannelBonding(Bool)	Bool	ISDN CAPI デバイスのチャネル結合を有効または無効にします。
<CRemoteData>.CapiExtensions(String)	String	通信に必要な追加の CAPI エクステンションを設定します。

European ISDN 接続のプロパティと戻り値の定義を次の表に示します。

表 2-14 における三角カッコの中の情報は実際値に置き換えてください。

表 2-14 ISDN via CAPI 2.0 のプロパティと戻り値

プロパティ	戻り値	説明
Bool = <CRemoteData>.CapiChannelBonding	Bool	リモートオブジェクトの現在の ISDN CAPI チャネル結合の設定を返します。
String = <CRemoteData>.CapiExtensions	String	リモートオブジェクトから CAPI エクステンションの現在のリストを返します。

CRemoteDataEx オブジェクト

CRemoteDataEx オブジェクトには、CRemoteData オブジェクトと同じ機能と、いくつかの追加機能が含まれています。

プロパティとパラメータを次の表に示します。

表 2-15 における三角カッコの中の情報は実際値に置き換えてください。

表 2-15 CRemoteDataEx のパラメータ

プロパティ	パラメータ	説明
<CRemoteData>.PrivateKey(String)	String	使う秘密キーコンテナの名前を設定します。
<CRemoteData>.CertificateName(String)	String	使う秘密キーの共通名を設定します。
<CRemoteData>.EncryptionLevel(Byte)	Byte	暗号化レベルを設定します。 次の値が有効です。 ■ -1: なし ■ 0: pcAnywhere ■ 1: 対称 ■ 2: 公開キー
<CRemoteData>.DenyLowerEncrypt(Bool)	Bool	低い暗号化レベルを使っているホストコンピュータへの接続をリモートコンピュータで許可するかどうかを定義します。
<CRemoteData>.AutoDomain(String)	String	自動ログインのためのドメイン名を設定します。このオプションは、NT 認証と Windows 認証で使われます。

CRemoteDataEx オブジェクトのプロパティと戻り値を次の表に示します。

表 2-16 における三角カッコの中の情報は実際値に置き換えてください。

表 2-16 CRemoteDataEx のプロパティと戻り値

プロパティ	戻り値	説明
String = <CRemoteData>.PrivateKey	String	現在アクティブな秘密キーコンテナの名前を返します。
String = <CRemoteData>.CertificateName	String	アクティブな秘密キーコンテナの共通名を返します。

プロパティ	戻り値	説明
Byte = <CRemoteData>.EncryptionLevel	Byte	次のいずれかの暗号化設定を返します。 ■ -1: なし ■ 0: pcAnywhere ■ 1: 対称 ■ 2: 公開キー
Bool = <CRemoteData>.DenyLowerEncrypt	Bool	低い暗号化を拒否する設定値を返します。
String = <CRemoteData>.AutoDomain	String	自動ログインのためのドメイン名の設定を返します。 このオプションは、NT 認証と Windows 認証で使われます。

リモート機能のための Visual Basic サンプルコード

次の Visual Basic サンプルコードではリモートデータオブジェクトを取得してそのプロパティを修正します。

```
Private Sub Command1_Click()
Dim RemoteDataManager as Object
Dim RemoteData as Object
Dim s as string

' CRemoteDataManager オブジェクトの作成
Set RemoteDataManager = CreateObject(WINAWSVR.REMOTEDATAMANAGER)

' カレントディレクトリの表示と変更
s = RemoteDataManager.CurrentDirectory()
MsgBox ( s )
RemoteDataManager.ChangeDirectory ( "C:¥dev¥bin.w32¥data" )
s = RemoteDataManager.CurrentDirectory()
MsgBox ( s )

' リモートデータオブジェクトを取得
Set RemoteData = RemoteDataManager.RetrieveObjectEx ("pod.CHF",
2, 0)

' 一部のプロパティを表示
```

```
s = RemoteData.AreaCode()
MsgBox (s)
s = RemoteData.PhoneNumber()
MsgBox (s)
```

一部のプロパティを設定

```
RemoteData.AreaCode = "212"
RemoteData.PhoneNumber = "555-5555"
```

オブジェクトをディスクに書き込み

```
RemoteData.WriteObject (0)
End Sub
```

次のように **FindFirst** メソッドと **FindNext** メソッドを使ってディレクトリのリモートファイルを表示します。

```
Private Sub Command5_Click()

Dim RemoteDataManager as Object
Dim RemoteData as Object
Dim s as string

Set RemoteDataManager =
CreateObject ("WINAWSVR.REMOTEDATAMANAGER")
RemoteDataManager.ChangeDirectory
("C:¥dev¥bin.w32¥data")
RemoteDataManager.FindFirst "*", s
MsgBox (s)
RemoteDataManager.FindNext s
MsgBox (s)
End Sub
```

リモートオブジェクトを作成します。次のように接続の種類を **TCP/IP** に設定してコンピュータ名を **Host1** に設定してからリモートオブジェクトを起動します。

```
Private Sub Command6_Click()
Dim RemoteDataManager as Object
Dim RemoteData as Object
Dim s as string
```

```

Set RemoteDataManager =
CreateObject ("WINAWSVR.REMOTEDATAMANAGER")
MsgBox (RemoteDataManager.CurrentDirectory ())

RemoteDataManager.ChangeDirectory ("C:¥dev¥bin.w32¥data")
MsgBox (RemoteDataManager.CurrentDirectory ())
Set RemoteData = RemoteDataManager.CreateObject ("test")
RemoteData.ConnectionType = "TCP/IP"
RemoteData.ComputerName = "Host1"
s = RemoteData.ConnectionType
MsgBox (s)
s = RemoteData.ComputerName
MsgBox (s)
RemoteData.WriteObject (0)
End Sub

```

CHostDataManager メソッド

CHostDataManager メソッドは、CHostData オブジェクトにアクセスしてそれらを制御するためのパラメータと戻り値を提供します。

CHostDataManager メソッド	パラメータまたは戻り値	説明
CurrentDirectory()	戻り値 = String	pcAnywhere ホストオブジェクトが格納されている現在の pcAnywhere データディレクトリの絶対パス名です。
FindFirst(Pattern)	パラメータ = Pattern (文字列)	オブジェクトファイルをフィルタ処理するファイル名パターンです。アスタリスク(*)はカレントディレクトリのすべてのファイルを見つけます。
FindFirst(Name string)	パラメータ = Name (文字列)	指定されたパターンに一致するホストオブジェクトファイルの絶対パス名のバッファを返します。
FindFirst	戻り値 = Boolean	指定されたパターンに一致するホストオブジェクトファイルが見つかった場合に TRUE 。一致するファイルの絶対パス名が Name に格納されます。
FindNext(Name)	パラメータ = Name (文字列)	FindFirst() に対する元の呼び出しで指定したパターンに一致するホストオブジェクトファイルの絶対パス名のバッファを返します。

CHostDataManager メソッド	パラメータまたは戻り値	説明
FindNext	戻り値 = Boolean	FindFirst() の呼び出しで指定したパターンに一致する別のホストオブジェクトファイルが見つかった場合の戻り値は TRUE です。一致するファイルの絶対パス名が Name に格納されます。
RetrieveObject(Name)	パラメータ = Name (文字列)	ロードするホストオブジェクトファイルの完全修飾名です。
RetrieveObject(AccessMode)	パラメータ = AccessMode (整数)	このオブジェクトの使い方を指定します。これはパスワード保護に関係します。次のオプションがあります。 ■ 0 = 指定なし ■ 1 = 表示のみ ■ 2 = 表示と修正 ■ 3 = 実行
RetrieveObject(Password)	パラメータ = Password (文字列)	オブジェクトのパスワードです。NULL にできません。
RetrieveObject	戻り値 = Object	指定されたファイルの CHostData オブジェクトです。
RetrieveObjectEx(Name)	パラメータ = Name (文字列)	ロードするホストオブジェクトファイルの完全修飾名です。
RetrieveObjectEx(AccessMode)	パラメータ = AccessMode (整数)	このオブジェクトの使い方を指定します。これはパスワード保護に関係します。次のオプションがあります。 ■ 0 = 指定なし ■ 1 = 表示のみ ■ 2 = 表示と修正 ■ 3 = 実行
RetrieveObjectEx(Password)	パラメータ = Password (文字列)	オブジェクトのパスワードです。NULL にできません。
RetrieveObjectEx	戻り値 = Object	指定されたファイルの CHostDataEx オブジェクトです。

CHostDataManager メソッド	パラメータまたは戻り値	説明
CreateObject(Name)	パラメータ = Name (文字列)	CHostData オブジェクトを作成してそれを指す LPDISPATCH ポインタを返します。パラメータは新しいオブジェクトのホストオブジェクトファイルの完全修飾名です。
CreateObject	戻り値 = Object	CHostData
CreateObjectEx(Name)	パラメータ = Name (文字列)	CHostDataEx オブジェクトを作成してそれを指す LPDISPATCH ポインタを返します。パラメータは新しいオブジェクトのホストオブジェクトファイルの完全修飾名です。
CreateObjectEx	戻り値 = Object	CHostDataEx
DeleteObject(Name)	パラメータ = Name (文字列)	削除するオブジェクトのホストオブジェクトファイルの完全修飾名です。
DeleteObject(Password)	パラメータ = Password (文字列)	オブジェクトのパスワードです。
DeleteObject	戻り値 = Boolean	オブジェクトが削除されれば TRUE。
Launch(Name)	パラメータ = Name (文字列)	ホストオブジェクトファイルを起動して pcAnywhere ホスト端末ウィンドウを開きます。パラメータは、起動するオブジェクトのホストオブジェクトファイルの完全修飾名です。
Launch	戻り値 = Boolean	オブジェクトが正常に起動した場合は TRUE。

CHostData プロパティ

CHostData オブジェクトで利用可能なプロパティとパラメータの説明を次の表に示します。

表 2-17 における三角カッコの中の情報は実際値に置き換えてください。

表 2-17 CHostData のプロパティとパラメータ

プロパティ	パラメータ	説明
<CHostData>.PhoneNumber(String)	String	ホストコンピュータの電話番号を設定します。
<CHostData>.UseDialingProperties(Bool)	Bool	システムのダイヤルのプロパティを設定します。

プロパティ	パラメータ	説明
<CHostData>.RedialCount(Integer)	Integer	呼び出しを中止するまでに行うリダイヤルの試行回数を設定します。
<CHostData>.RedialDelay(Integer)	Integer	リダイヤルの試行間の待ち時間 (秒単位) を設定します。
<CHostData>.Password(String)	String	ExecuteProtection、ReadProtection、WriteProtection の設定で使うホストオブジェクトのパスワードを設定します。 セキュリティ上の理由から、pcAnywhere Automation Server はパスワード値を読み込む機能を提供していません。パスワード値は返されません。
<CHostData>.ExecuteProtection(Bool)	Bool	オブジェクトを実行するためのパスワードの必要条件を設定します。Password で設定します。
<CHostData>.ReadProtection(Bool)	Bool	ホストオブジェクトのプロパティを表示するためのパスワードの必要条件を設定します。Password で設定します。
<CHostData>.WriteProtection(Bool)	Bool	ホストオブジェクトに対する変更を保存するためのパスワードの必要条件を設定します。Password で設定します。
<CHostData>.LogSession(Bool)	Bool	セッションのログ記録を有効または無効にします。
<CHostData>.RecordFile(String)	String	アクティブセッションを記録するファイルの場所のパスとファイルの完全修飾名を設定します。
<CHostData>.RecordSession(Bool)	Bool	自動セッション記録を有効または無効にします。
<CHostData>.ReadObject(String)	String	オブジェクトのパスワードを設定します。このプロパティはホストオブジェクトのローカルデータコピーを更新するために使います。
<CHostData>.WriteObject(String)	String	オブジェクトのパスワードを設定します。このプロパティはホストオブジェクトの作成またはホストオブジェクトに対する変更の書き込みに使います。

CHostData のプロパティと戻り値の説明を次の表に示します。

表 2-18 における三角カッコの中の情報は実際値に置き換えてください。

表 2-18 CHostData のプロパティと戻り値

プロパティ	戻り値	説明
String = <CHostData>.PhoneNumber	String	ホストコンピュータの電話番号を返します。
Bool = <CHostData>.UseDialingProperties	Bool	ホストオブジェクトに設定されているシステムのダイヤルプロパティの使用を返します。
Integer = <CHostData>.RedialCount	Integer	ホストオブジェクトに設定されているリダイヤルの試行回数を返します。
Integer = <CHostData>.RedialDelay	Integer	リダイヤルの試行間の秒数を返します。
Bool = <CHostData>.ExecuteProtection	Bool	ExecuteProtection 設定の値を返します。
Bool = <CHostData>.ReadProtection	Bool	ReadProtection 設定の値を返します。
Bool = <CHostData>.WriteProtection	Bool	WriteProtection 設定の値を返します。
Bool = <CHostData>.LogSession	Bool	セッションのログ記録が有効ならば TRUE を返します。
String = <CHostData>.RecordFile	String	セッション記録ファイルとパスの完全修飾名を返します。
Bool = <CHostData>.RecordSession	Bool	セッション記録設定の値を返します。

接続の種類のプロパティ

接続の種類のプロパティとパラメータを次の表に示します。

表 2-19 における三角カッコの中の情報は実際値に置き換えてください。

表 2-19 接続の種類のプロパティとパラメータ

プロパティ	パラメータ	説明
<CHostData>.ConnectionType(String)	String	ホストの接続の種類を設定します。渡される値は FirstConnectionType() 関数と NextConnectionType() 関数で定義されている有効な接続の種類である必要があります。 有効な接続の種類の例を次に示します。 ■ COM1 ■ COM2 ■ COM3 ■ COM4 ■ TCP/IP ■ SPX ■ NetBIOS ■ ISDN via CAPI 2.0 (日本の環境ではご利用いただけません) ■ モデム名 (コンピュータに表示されるモデム名)
<CHostData>.AssignConnection(String)	String	この文字列で定義された接続の種類をアクティブな状態に設定します。
<CHostData>.UnassignConnection(String)	String	この文字列で定義された接続の種類を非アクティブな状態に設定します。

接続の種類のプロパティと戻り値を次の表に示します。

表 2-20 における三角カッコの中の情報は実際値に置き換えてください。

表 2-20 接続の種類のプロパティと戻り値

プロパティ	戻り値	説明
String = <CHostData>.ConnectionTypes	String	ホストオブジェクトの接続の種類を返します。

プロパティ	戻り値	説明
Integer = <CHostData>.ConnectionType	Integer	利用可能な接続の種類を返します。 有効な接続の種類の例を次に示します。 ■ COM1 ■ COM2 ■ COM3 ■ COM4 ■ TCP/IP ■ SPX ■ NetBIOS ■ ISDN via CAPI 2.0 (日本の環境ではご利用いただけません) ■ モデム名 (コンピュータに表示されるモデム名)
String = <CHostData>.FirstConnectionType	String	最初に利用可能な接続の種類を返します。
String = <CHostData>.NextConnectionType	String	次に利用可能な接続の種類を返します。このプロパティは <CHostData>.ConnectionTypes に設定された接続の種類の数に対して順次呼び出され、すべての接続の種類が列挙されます。
Bool = <CHostData>.FindConnectionType(ConnectionType)	Bool	利用可能な接続の種類のリストの中で指定された接続の種類が見つかった場合は TRUE を返します。
Integer = <CHostData>.MaxAssignedConnections	Integer	このホストでアクティブにできる接続の種類の最大数を返します。
String = <CHostData>.FirstAssignedConnection	String	最初に割り当てられたアクティブな接続の種類を返します。
String = <CHostData>.NextAssignedConnection	String	次に割り当てられたアクティブな接続の種類を返します。
Bool = <CHostData>.FindAssignedConnection(ConnectionType)	Bool	渡された接続の種類がアクティブな接続の種類のいずれかと一致する場合は TRUE を返します。

AssignConnection(ConnectionType) メソッド

AssignConnection(ConnectionType) メソッドは要求された接続の種類をホストオブジェクトの割り当て済み接続の種類のリストに入れ、以降のデバイス固有のメソッド呼び出しを処理するときにその接続の種類を現在の接続の種類にします。

要求された接続の種類が割り当て済み接続のリストにすでにある場合、割り当て済み接続のリストは変更されません。現在の接続の種類が要求された種類に変わるだけです。

接続固有の値の取得や設定をするときに同じオブジェクトで **AssignConnection** メソッドを複数回呼び出すことは正常です。

AssignConnection は渡された接続の種類がコンピュータに存在しその割り当てに成功した場合またはそれがすでに割り当てられている場合は **TRUE** を返します。要求された接続の種類がコンピュータに存在しない場合または現在の割り当て済み接続数がすでに最大許容レベルになっている場合は **FALSE** を返します。

pcAnywhere ホストオブジェクトは割り当て済み接続の種類を 2 つまでサポートできます。**AssignConnection** メソッドはこの限度を超える試行を検出した場合は **FALSE** を返します。

AssignConnection のパラメータの定義を次の表に示します。

パラメータ	説明
ConnectionType as string	割り当てる接続デバイスの種類の名前です。

AssignConnection の戻り値の定義を次の表に示します。

戻り値	説明
Boolean	このデバイスの種類が利用可能で割り当て済み接続の最大許容数にまだ達していない場合は TRUE です。

UnassignConnection(ConnectionType) メソッド

UnassignConnection(ConnectionType) メソッドは渡された接続の種類が割り当て済み接続の種類のリストから正常に削除された場合に **TRUE** を返します。

UnassignConnection のパラメータの定義を次の表に示します。

パラメータ	説明
ConnectionType as string	割り当て解除する接続デバイスの種類の名前です。

UnassignConnection の戻り値の定義を次の表に示します。

戻り値	説明
Boolean	このデバイスの種類が正常に割り当て解除された場合は TRUE です。

ダイヤルのプロパティ

モデム接続にダイヤルのプロパティを設定するためのプロパティと戻り値の定義を次の表に示します。

表 2-21 における三角カッコの中の情報は実際値に置き換えてください。

表 2-21 モデムのダイヤルのプロパティに使われるプロパティとパラメータ

プロパティ	パラメータ	説明
<CHostData>.AreaCode(String)	String	モデム接続の市外局番を設定します。
<CHostData>.CountryCode(String)	String	モデム接続の国番号を設定します。

モデムのダイヤルのプロパティに使われるプロパティと戻り値の定義を次の表に示します。

表 2-22 における三角カッコの中の情報は実際値に置き換えてください。

表 2-22 モデムのダイヤルのプロパティに使われるプロパティと戻り値

プロパティ	戻り値	説明
String = <CHostData>.AreaCode	String	市外局番を返します。
String = <CHostData>.CountryCode	String	国番号を返します。
Integer = <CHostData>.CountryCodes	Integer	利用可能な国番号の数を返します。
String = <CHostData>.FirstCountryCode	String	オペレーティングシステムのリストにある最初に利用可能な国番号を返します。
String = <CHostData>.NextCountryCode	String	次に利用可能な国番号を返します。これは <CHostData>.CountryCodes に設定された国番号の数に対して順に呼び出され、すべての国番号が列挙されます。

COM デバイスのプロパティ

モデム接続などの COM ベースの接続を目的としたポートの設定をカスタマイズできるようにするプロパティとパラメータの定義を次の表に示します。

表 2-23 における三角カッコの中の情報は実際値に置き換えてください。

表 2-23COM デバイスのプロパティとパラメータ

プロパティ	パラメータ	説明
<CHostData>.ComParity(String)	String	通信パリティを設定します。 次の値が有効です。 ■ <None> ■ Odd ■ Even ■ Mark ■ Space
<CHostData>.ComFlowControl(String)	String	COM ベースの接続の種類のフロー制御を設定します。 次の値が有効です。 ■ <None> ■ XONXOFF ■ RTS/CTS ■ Both
<CHostData>.ComStartedBy(String)	String	COM ベースの接続の種類の開始設定を設定します。 次の値が有効です。 ■ Always connected ■ Carrier detect (DCD) ■ Clear to send (CTS) ■ Data set ready (DSR) ■ Ring indicator (RI) ■ Receive 2 <CR>'s ■ Modem response
<CHostData>.ComEndedBy(String)	String	COM ベースの接続の種類の終了設定を設定します。 次の値が有効です。 ■ Always connected ■ Carrier detect (DCD) ■ Clear to send (CTS) ■ Data set ready (DSR) ■ Ring indicator (RI)

プロパティ	パラメータ	説明
<CHostData>.ComSpeed(Long)	Long	最大 COM 速度の設定です。 次の値が有効です。 ■ 110 ■ 300 ■ 600 ■ 1200 ■ 2400 ■ 4800 ■ 9600 ■ 38400 ■ 57600 ■ 115200

COM デバイスのプロパティと戻り値を次の表に示します。

表 2-24 における三角カッコの中の情報は実際値に置き換えてください。

表 2-24 COM デバイスのプロパティと戻り値

プロパティ	戻り値	説明
String = <CHostData>.ComParity	String	通信パリティとして次の値のいずれかを返します。 ■ <None> ■ Odd ■ Even ■ Mark ■ Space
String = <CHostData>.ComFlowControl	String	ホストオブジェクトの Com Flow の設定を返します。 次の値が有効です。 ■ <None> ■ XONXOFF ■ RTS/CTS ■ Both

プロパティ	戻り値	説明
String = <CHostData>.ComStartedBy	String	COM ベースの接続の種類の Com Start 制御を返します。 次の値が有効です。 ■ Always connected ■ Carrier detect (DCD) ■ Clear to send (CTS) ■ Data set ready (DSR) ■ Ring indicator (RI) ■ Receive 2 <CR>'s ■ Modem response
String = <CHostData>.ComEndedBy	String	COM ベースの接続の種類の Com End 制御を返します。 次の値が有効です。 ■ Always connected ■ Carrier detect (DCD) ■ Clear to send (CTS) ■ Data set ready (DSR) ■ Ring indicator (RI)
Long = <CHostData>.ComSpeed	Long	ホストオブジェクトの Com Speed の現在の設定を返します。 次の値が有効です。 ■ 110 ■ 300 ■ 600 ■ 1200 ■ 2400 ■ 4800 ■ 9600 ■ 38400 ■ 57600 ■ 115200

NetBIOS デバイスのプロパティ

NetBIOS ネットワークデバイスのプロパティとパラメータの定義を次の表に示します。

表 2-25 における三角カッコの中の情報は実際値に置き換えてください。

表 2-25 NetBIOS のプロパティとパラメータ

プロパティ	パラメータ	説明
<CHostData>.LanaNumber(Integer)	Integer	NetBIOS 接続の LAN アダプタ (LANA) 番号を設定します。

NetBIOS ネットワークデバイスのプロパティと戻り値の定義を次の表に示します。

表 2-26 における三角カッコの中の情報は実際値に置き換えてください。

表 2-26 NetBIOS の戻り値

プロパティ	戻り値	説明
Integer = <CHostData>.LanaNumber	Integer	NetBIOS 接続の LAN アダプタ (LANA) 番号の現在の設定を返します。

ISDN via CAPI 2.0 デバイスのプロパティ

European ISDN 接続のプロパティとパラメータの定義を次の表に示します。

表 2-27 における三角カッコの中の情報は実際値に置き換えてください。

表 2-27 ISDN via CAPI 2.0 のプロパティとパラメータ

プロパティ	パラメータ	説明
<CHostData>.CapiChannelBonding(Bool)	Bool	ISDN CAPI デバイスのチャネル結合を有効または無効にします。
<CHostData>.CapiExtensions(String)	String	通信に必要な追加の CAPI エクステンションを設定します。

European ISDN 接続のプロパティと戻り値の定義を表 2-28 に示します。

表 2-28 ISDN via CAPI 2.0 のプロパティと戻り値

プロパティ	戻り値	説明
Bool = <CHostData>.CapiChannelBonding	Bool	ホストオブジェクトの現在の ISDN CAPI チャネル結合の設定を返します。
String = <CHostData>.CapiExtensions	String	ホストオブジェクトから CAPI エクステンションの現在のリストを返します。

CHostDataEx オブジェクト

CHostDataEx オブジェクトには、CHostData オブジェクトと同じ機能と、いくつかの追加機能が含まれています。

プロパティとパラメータを次の表に示します。

表 2-29 における三角カッコの中の情報は実際値に置き換えてください。

表 2-29 CHostDataEx のプロパティとパラメータ

プロパティ	パラメータ	説明
<CHostData>.CryptPrivateKey(String)	String	使う秘密キーコンテナの名前を設定します。
<CHostData>.CryptCommonName(String)	String	使う秘密キーの共通名を設定します。
<CHostData>.CryptReqLevel(Byte)	Byte	暗号化レベルを設定します。 次の値が有効です。 ■ -1: なし ■ 0: pcAnywhere ■ 1: 対称 ■ 2: 公開キー
<CHostData>.CryptRefuseLower(Bool)	Bool	低い暗号化レベルを使っているリモートコンピュータからの接続をホストコンピュータで受け入れるかどうかを定義します。
<CHostData>.CallersPath(String)	String	呼び出し側ファイルの完全修飾パスを設定します。
<CHostData>.ConfirmConnect(Bool)	Bool	ホストユーザーに接続確認のメッセージ表示するかどうかを定義します。
<CHostData>.ConfirmTimeout(Byte)	Byte	接続の確認がタイムアウトになるまでの秒数を設定します。
<CHostData>.ConfirmDeny(Bool)	Bool	接続の確認がタイムアウトになった場合に接続を終了するかどうかを定義します。
<CHostData>.PwCaseSensitive(Bool)	Bool	大文字と小文字を区別するパスワードの使用を義務付けます。
<CHostData>.PwAttempts(Byte)	Byte	ログオンに連続で何回失敗したら接続を終了するかを設定します。
<CHostData>.PwTimeout(Byte)	Byte	何分以内にログオンを完了しないと接続が切断されるかを設定します。

プロパティ	パラメータ	説明
<CHostData>.ActiveKbds(Byte)	Byte	接続時にどのマウスとキーボードをアクティブにするかを定義します。 次の値が有効です。 ■ 0: ホストとリモート ■ 1: ホスト ■ 2: リモート
<CHostData>.InactiveTimeout(Byte)	Byte	キーボードとマウスが活動なしになってから何分間経過したら接続を終了するかを設定します。
<CHostData>.LockSystemWhileWait(Bool)	Bool	起動時にホストコンピュータをロックするオプションを設定します。
<CHostData>.MinimizeOnLaunch(Bool)	Bool	最小化して実行するホスト起動オプションを設定します。
<CHostData>.RunAsService(Bool)	Bool	ホストをサービスとして実行できるようにします。
<CHostData>.ConnLostWait(Byte)	Byte	別の呼び出し側からの接続を許可するまでの待機時間を分単位で設定します。
<CHostData>.ConnLostHostOpts(Bool)	Bool	セッションが異常終了した場合に別の接続を待つかホストを中止するかを定義します。 次の値が有効です。 ■ FALSE: 待機 ■ TRUE: ホストを中止
<CHostData>.EnableConnLostSecurity(Bool)	Bool	異常終了したセッションに対するセッションの終了セキュリティオプションを有効または無効にします。

プロパティ	パラメータ	説明
<CHostData>.AuthenticationType(Byte)	Byte	認証の種類を設定します。 次の値が有効です。 ■ 0: pcAnywhere ■ 1: pcAnywhere ■ 2: Windows ■ 3: NT ■ 4: pcAnywhere ■ 5: pcAnywhere ■ 6: ADS (Active Directory Services) ■ 7: Microsoft LDAP ■ 8: FTP ■ 9: HTTP ■ 10: HTTPS ■ 11: RSA SecurID 認証の値が設定されていない場合や、設定されている値が無効な場合(その認証の種類が利用可能でない場合など)は、デフォルトで pcAnywhere 認証が使われます。
<CHostData>.ConnLostSecurity(Byte)	Byte	セッションの異常終了を処理するためのセキュリティオプションを設定します。 次の値が有効です。 ■ 1: ユーザーをログオフ ■ 2: ホストコンピュータを再起動 ■ 3: コンピュータをロック
<CHostData>.CallbkDelay(Byte)	Byte	ホストモデムがリモートをコールバックするまでの待機時間を秒単位で設定します。
<CHostData>.EndSessHostOpts(Bool)	Bool	セッションの正常終了後に別の接続を待つかホストを中止するかを定義します。 次の値が有効です。 ■ FALSE: 次の接続を待機 ■ TRUE: ホストを中止
<CHostData>.EnableEndSessSecurity(Bool)	Bool	セッションの正常終了のセキュリティオプションを有効または無効にします。

プロパティ	パラメータ	説明
<CHostData>.EndSessSecurity(Byte)	Byte	セッションの正常終了のセキュリティオプションを設定します。 次の値が有効です。 ■ 1: ユーザーをログオフ ■ 2: ホストコンピュータを再起動 ■ 3: コンピュータをロック
<CHostData>.BlankHost(Bool)	Bool	ホストコンピュータの画面の消去を設定します。一部のビデオカードはこのオプションをサポートしません。
<CHostData>.AllowRemoteMouse(Bool)	Bool	セッション中にリモートコンピュータでマウスを使えないようにします。
<CHostData>.RebootOnDisconnect(Bool)	Bool	TRUE の場合、セッションの終了後にコンピュータを強制的に再起動します。
<CHostData>.PasswordAfterDisc(Bool)	Bool	セッションの終了後にユーザーをログオフします。
<CHostData>.LogFailures(Bool)	Bool	失敗したパスワードの試行をログに記録します。
<CHostData>.AllowDriveSecurity(Bool)	Bool	ドライブのセキュリティオプションを有効にします。この設定は NTFS ファイルシステムでのみ有効です。
<CHostData>.UseDirectoryServices(Bool)	Bool	認証でディレクトリサービスを使えるようにします。
<CHostData>.DirectoryServiceEntry(String)	String	ディレクトリサービスの設定を行います。

CHostDataEx オブジェクトのプロパティと戻り値を次の表に示します。

表 2-30 における三角カッコの中の情報は実際値に置き換えてください。

表 2-30 CHostDataEx のプロパティと戻り値

プロパティ	戻り値	説明
String = <CHostDataEx>.CryptPrivateKey	String	現在アクティブな秘密キーコンテナを返します。
String = <CHostDataEx>.CryptCommonName	String	アクティブな秘密キーコンテナの共通名を返します。

プロパティ	戻り値	説明
Byte = <CHostDataEx>.CryptReqLevel	Byte	次のいずれかの暗号化設定を返します。 ■ -1: なし ■ 0: pcAnywhere ■ 1: 対称 ■ 2: 公開キー
Bool = <CHostDataEx>.CryptRefuseLower	Bool	低い暗号化を拒否する設定値を返します。
String = <CHostDataEx>.CallersPath	String	呼び出し側ファイルの現在アクティブなパスを返します。
Bool = <CHostDataEx>.ConfirmConnect	Bool	接続時の確認の設定値を返します。
Byte = <CHostDataEx>.ConfirmTimeout	Byte	低い暗号化を拒否する設定値を返します。
Bool = <CHostDataEx>.ConfirmDeny	Bool	時間切れならば接続解除の設定値を返します。
Bool = <CHostDataEx>.PwCaseSensitive	Bool	パスワードの大文字と小文字を区別する設定値を返します。
Byte = <CHostDataEx>.PwAttempts	Byte	ログインの再試行回数の設定値を返します。
Byte = <CHostDataEx>.PwTimeout	Byte	ログイン入力の制限時間の設定値(分単位)を返します。
Byte = <CHostDataEx>.ActiveKbds	Byte	アクティブなキーボードとマウスの設定を返します。 次の値が有効です。 ■ 0: ホストとリモート ■ 1: ホスト ■ 2: リモート
Byte = <CHostDataEx>.InactiveTimeout	Byte	活動なしになってから何分間経過すると接続が切断されるかを返します。
Bool = <CHostDataEx>.LockSystemWhileWait	Bool	起動時にホストコンピュータをロックするオプションの設定を返します。
Bool = <CHostDataEx>.MinimizeOnLaunch	Bool	最小化して実行するホスト起動オプションの設定を返します。
Bool = <CHostDataEx>.RunAsService	Bool	ホストをサービスとして実行する設定を返します。
Byte = <CHostDataEx>.ConnLostWait	Byte	別の接続を許可するまでの待機時間(分単位)を返します。

プロパティ	戻り値	説明
Bool = <CHostDataEx>.ConnLostHostOpts	Bool	セッションが異常終了した場合に別の接続を待つ かホストを中止するかを返します。 次の値が有効です。 ■ FALSE: 待機 ■ TRUE: ホストを中止
Bool = <CHostDataEx>.ConnLostWait	Bool	セッションの異常終了の処理に対して設定されて いるセキュリティオプションの値を返します。
Byte = <CHostDataEx>.AuthenticationType	Byte	認証の種類を表す番号を返します。 次の値が有効です。 ■ 0: pcAnywhere ■ 1: pcAnywhere ■ 2: Windows ■ 3: NT ■ 4: pcAnywhere ■ 5: pcAnywhere ■ 6: ADS (Active Directory Services) ■ 7: Microsoft LDAP ■ 8: FTP ■ 9: HTTP ■ 10: HTTPS ■ 11: RSA SecurID 認証の値が設定されていない場合や、設定されて いる値が無効な場合(その認証の種類が利用可能 でない場合など)は、デフォルトで pcAnywhere 認証が使われます。
Byte = <CHostDataEx>.ConnLostSecurity	Byte	セッションの異常終了の処理に対して設定されて いるセキュリティレベルを表す数値を返します。 次の値が有効です。 ■ 1: ユーザーをログオフ ■ 2: ホストコンピュータを再起動 ■ 3: コンピュータをロック
Byte = <CHostDataEx>.CallbkDelay	Byte	ホストモデムがリモートをコールバックするまでの待 機時間(秒)を返します。

プロパティ	戻り値	説明
Bool = <CHostDataEx>.EndSessHostOpts	Bool	セッションの正常終了後に別の接続を待つかホストを中止するかを返します。 次の値が有効です。 ■ FALSE: 次の接続を待機 ■ TRUE: ホストを中止
Bool = <CHostDataEx>.EnableEndSessSecurity	Bool	セッションの正常終了に対してセッションの終了セキュリティオプションが有効になっているかどうかを返します。
Byte = <CHostDataEx>.EndSessSecurity	Byte	セッションの正常終了に対して設定されているセキュリティオプションを返します。 次の値が有効です。 ■ 1: ユーザーをログオフ ■ 2: ホストコンピュータを再起動 ■ 3: コンピュータをロック
Bool = <CHostDataEx>.BlankHost	Bool	設定されている画面の消去のオプションを返します。
Bool = <CHostDataEx>.AllowRemoteMouse	Bool	セッション中にリモートユーザーがキーボードとマウスを制御できるかどうかを返します。
Bool = <CHostDataEx>.RebootOnDisconnect	Bool	セッションの終了後にホストコンピュータを再起動するかどうかを返します。
Bool = <CHostDataEx>.PasswordAfterDisc	Bool	セッションの終了後にホストユーザーをログオフするかどうかを返します。
Bool = <CHostDataEx>.LogFailures	Bool	パスワードの失敗をログに記録するオプションが設定されているかどうかを返します。
Bool = <CHostDataEx>.AllowDriveSecurity	Bool	ドライブのセキュリティオプションが有効かどうかを返します。
Bool = <CHostDataEx>.UseDirectoryServices	Bool	ディレクトリサービスのオプションが有効かどうかを返します。
String = <CHostDataEx>.DirectoryServiceEntry	String	ディレクトリサービスの設定を返します。

ホスト機能のための Visual Basic サンプルコード

次の Visual Basic サンプルコードではホストデータオブジェクトを取得してそのプロパティを修正します。

```
Private Sub Command1_Click()  
Dim HostDataManager as Object  
Dim HostData as Object  
Dim s as string
```

'CHostDataManager オブジェクトを作成

```
Set HostDataManager = CreateObject(WINAWSVR.BEHOSTDATAMANAGER)
```

'カレントディレクトリの表示と変更

```
s = HostDataManager.CurrentDirectory()  
MsgBox ( s )  
HostDataManager.ChangeDirectory ("C:¥dev¥bin.w32¥data")  
s = HostDataManager.CurrentDirectory()  
MsgBox ( s )
```

'ホストデータオブジェクトを取得

```
Set HostData = HostDataManager.RetrieveObject("pod.BHF", 2, 0)
```

'一部のプロパティを表示

```
s = HostData.AreaCode()  
MsgBox (s)  
s = HostData.PhoneNumber()  
MsgBox (s)
```

'一部のプロパティを設定

```
RemoteData.HostData = "212"  
RemoteData.HostData = "555-5555"
```

'オブジェクトをディスクに書き込み

```
HostData.WriteObject (0)  
End Sub
```

次のように **FindFirst** メソッドと **FindNext** メソッドを使ってディレクトリのホストファイルを表示します。

```
Private Sub Command5_Click()  
  
  
Dim HostDataManager as Object  
Dim HostData as Object
```

```
Dim s as string

Set HostDataManager = CreateObject("WINAWSVR.BEHOSTDATAMANAGER")
HostDataManager.ChangeDirectory ("C:¥dev¥bin.w32¥data")
HostDataManager.FindFirst "*", s
MsgBox (s)
HostDataManager.FindNext s
MsgBox (s)

End Sub
```

Awrem32 関数

Awrem32 関数は、ホストコンピュータとリモートコンピュータの間の接続を処理するためのパラメータと戻り値を提供します。

Awrem32 関数	パラメータまたは戻り値	説明
awConnect(FileName)	パラメータ = Name(文字列)	ホストコンピュータについての情報を含む.chf ファイルの完全修飾名です。
awConnect	戻り値 = Boolean	コマンドが実行された場合は TRUE です。
awDisconnect()	戻り値 = Boolean	呼び出し側のプログラムはこの関数の呼び出し後にオブジェクトを削除(C++ の場合は IAwrem32X* を削除、VB の場合は ObjectName = Nothing; を設定)する必要があります。
FileXferFromHost(HostFile)	パラメータ = HostFile(文字列)	コピー元のホストコンピュータのパスとファイルの完全修飾名です。
FileXferFromHost(RemoteFile)	パラメータ = RemoteFile(文字列)	コピー先のパスとファイルの完全修飾名です。HostFile と RemoteFile の文字列は異なる必要があります。
FileXferFromHost	戻り値 = Boolean	コマンドが実行された場合は TRUE です。
FileXferToHost(HostFile)	パラメータ = HostFile(文字列)	リモートコンピュータからホストコンピュータにファイルをコピーします。パラメータにはワイルドカード文字を使うことができます。 コピー先のパスとファイルの完全修飾名です。

Awrem32 関数	パラメータまたは戻り値	説明
FileXferToHost(RemoteFile)	パラメータ = RemoteFile (文字列)	コピー元のリモートコンピュータのパスとファイルの完全修飾名です。HostFile と RemoteFile の文字列は異なる必要があります。
CreateFolderOnHost(FolderName)	パラメータ = FolderName (文字列)	ホストコンピュータ上に作成するフォルダのドライブとパスです。
CreateFolderOnHost	戻り値 = Boolean	コマンドが実行された場合は TRUE です。
ExecuteHostFile(FileName)	パラメータ = FileName (文字列)	ホストコンピュータ上で既存のファイルを実行します。この関数が実行するのは、バッチファイル、コマンドファイル、実行可能ファイルのみです。実行可能ファイルに関連付けされたファイルは実行しません。たとえば、.doc ファイルを実行しても Microsoft Word は起動しません。 ホストコンピュータ上のファイルの完全修飾パスです。
ExecuteHostFile	戻り値 = Boolean	コマンドが実行された場合は TRUE です。
GetError()	戻り値 = String	Awrem32 が生成した最後のエラーを返します。
ConnectionStatus()	戻り値 = Short	ホストコンピュータに対する接続の現在の状態を返します。 有効な値は次のとおりです。 ■ -1 = 接続の消失 ■ 0 = 接続なし ■ 1 = 接続セッション

Visual C++ のオブジェクト定義

この章では以下の項目について説明しています。

- Visual C++ オブジェクトについて
- CRemoteDataManager メソッド
- CRemoteData オブジェクト
- CRemoteDataEx オブジェクト
- CHostDataManager メソッド
- CHostData オブジェクト
- CHostDataEx オブジェクト
- Awrem32 関数

Visual C++ オブジェクトについて

pcAnywhere Automation Server は OLE オートメーションをサポートするために次のコンポーネントを提供します。

Winawsvr	pcAnywhere Automation Server に接続してそのインターフェースにアクセスするために必要な情報を提供する
Awrem32	pcAnywhere 接続を制御するために必要な情報を提供する

Winawsvr を構成するオブジェクトを表 3-1 に示します。

表 3-1 Winawsvr のオブジェクト

オブジェクト	説明	参照先
CRemoteDataManager	CRemoteData オブジェクトの作成、オープン、修正、保存、削除のためのメソッドを提供します。	p.54 の「CRemoteDataManager メソッド」を参照してください。
CRemoteData	pcAnywhere リモート機能のアクセスと制御のためのパラメータを定義します。	p.57 の「CRemoteData オブジェクト」を参照してください。 p.64 の「CRemoteDataEx オブジェクト」を参照してください。
CHostDataManager	CHostData オブジェクトの作成、オープン、修正、保存、削除のためのメソッドを提供します。	p.65 の「CHostDataManager メソッド」を参照してください。
CHostData	pcAnywhere ホスト機能のアクセスと制御のためのパラメータを定義します。	p.57 の「CRemoteData オブジェクト」を参照してください。 p.64 の「CRemoteDataEx オブジェクト」を参照してください。

Awrem32 には 1 つのオブジェクトがあり、このオブジェクトはリモート制御とファイル転送のセッションをサポートする 8 つのインターフェースから構成されます。

p.79 の「Awrem32 関数」を参照してください。

一部の機能はすでにサポートされていません。ただし、以前のバージョンでの使用のためにオブジェクト定義については説明しています。

パスワードが必要な関数については、パスワード値は設定できますが取得はできません。これはセキュリティ上の理由によるものです。

CRemoteDataManager メソッド

CRemoteDataManager メソッドは CRemoteData オブジェクトにアクセスして制御するためのパラメータと戻り値を提供します。

メソッド	パラメータまたは 説明 戻り値
BSTR CurrentDirectory();	パラメータ = BSTR pcAnywhere リモートオブジェクトが格納されている現在の pcAnywhere データディレクトリの絶対パス名です。

メソッド	パラメータまたは 戻り値	説明
BOOL ChangeDirectory(LPCTSTR lpszNewDirectory);	パラメータ = LPCTSTR lpszNewDirectory 戻り値 = BOOL	pcAnywhere リモートオブジェクトが格納さ れているカレントディレクトリを変更します。 パラメータは既存ディレクトリの名前です。 成功した場合の戻り値は TRUE です。
BOOL FindFirst(LPCTSTR lpszPattern, BSTR FAR*);	パラメータ = LPCTSTR lpszPattern	オブジェクトファイルをフィルタ処理するファ イル名パターンです。アスタリスク(*)はカレ ントディレクトリのすべてのファイルを見つけ ます。
BOOL FindFirst(pbstrFullQualName);	パラメータ = BSTR FAR * pbstrFullQualName	指定されたパターンに一致するリモートオブ ジェクトファイルの絶対パス名のバッファを 返します。
BOOL FindFirst	戻り値 = BOOL	指定されたパターンに一致するリモートオブ ジェクトファイルが見つかった場合に TRUE 。 一致するファイルの絶対パス名は pbstrFullQualName に格納されます。
BOOL FindNext(BSTR FAR* pbstrFullQualName);	パラメータ = BSTR FAR * pbstrFullQualName 戻り値 = BOOL	FindFirst() の呼び出しが成功してカレント ディレクトリにあるリモートオブジェクトファ イルの名前を取得したら、 FindNext() を呼び 出してパターンに一致する次のファイル (も しあれば) を検索できます。 FindFirst() の呼び出しで指定したパターン に一致する別のリモートオブジェクトファイ ルが見つかった場合の戻り値は TRUE です。 一致するファイルの絶対パス名は pbstrFullQualName に格納されます。
LPDISPATCH RetrieveObject(LPCTSTR lpszFQName);	パラメータ = LPCTSTR lpszFQName	CRemoteData オブジェクトをファイル名で 取得します。 パラメータは、ロードするリモートオブジェ トファイルの完全修飾名です。
LPDISPATCH RetrieveObject(short wAccessMode);	パラメータ = short wAccessMode	このオブジェクトの使い方を指定します。こ れはパスワード保護に関係します。次のオ プションがあります。 ■ 0 = 指定なし ■ 1 = 表示のみ ■ 2 = 表示と修正 ■ 3 = 実行

メソッド	パラメータまたは 戻り値	説明
LPDISPATCH RetrieveObject(LPCTSTR lpszPassword);	パラメータ = LPCTSTR lpszPassword	オブジェクトのパスワードです。NULL にで きます。
LPDISPATCH RetrieveObjectEx(LPCTSTR lpszFQName);	パラメータ = LPCTSTR lpszFQName	CRemoteDataEx オブジェクトをファイル名 で取得します。 パラメータは、ロードするリモートオブジェク トファイルの完全修飾名です。
LPDISPATCH RetrieveObjectEx(short wAccessMode);	パラメータ = short wAccessMode	このオブジェクトの使い方を指定します。こ れはパスワード保護に関係します。次のオ プションがあります。 ■ 0 = 指定なし ■ 1 = 表示のみ ■ 2 = 表示と修正 ■ 3 = 実行
LPDISPATCH RetrieveObjectEx(LPCTSTR lpszPassword);	パラメータ = LPCTSTR lpszPassword	オブジェクトのパスワードです。NULL にで きます。
LPDISPATCH RetrieveObjectEx	戻り値 = LPDISPATCH	OLE ディスパッチオブジェクトを指すポイン タ。オブジェクトは CRemoteDataEx オブ ジェクトです。 「リモート機能のための Visual C++ サンプ ルコード」を参照してください。
LPDISPATCH CreateObject(LPCTSTR lpszFQName);	パラメータ = LPCTSTR lpszFQName 戻り値 = LPDISPATCH	CRemoteData オブジェクトを作成し、その オブジェクトを指す LPDISPATCH ポインタ を返します。 パラメータは、新しいオブジェクトのリモート オブジェクトファイルの完全修飾名です。 戻り値は、OLE ディスパッチオブジェクトを 指すポインタです。オブジェクトは CRemoteData オブジェクトです。 「リモート機能のための Visual C++ サンプ ルコード」を参照してください。

メソッド	パラメータまたは 戻り値	説明
LPDISPATCH CreateObjectEx(LPCTSTR lpszFQName);	パラメータ = LPCTSTR lpszFQName 戻り値 = LPDISPATCH	CRemoteDataEx オブジェクトを作成してそ れを指す LPDISPATCH ポインタを返しま す。 パラメータは、新しいオブジェクトのリモート オブジェクトファイルの完全修飾名です。 戻り値は、OLE ディスパッチオブジェクトを 指すポインタです。オブジェクトは CRemoteDataEx オブジェクトです。 「リモート機能のための Visual C++ サンプ ルコード」 を参照してください。
BOOL DeleteObject(LPCTSTR lpszFQName);	パラメータ = LPCTSTR lpszFQName	リモートオブジェクトファイルを削除します。 パラメータは、削除するオブジェクトのリモート オブジェクトファイルの完全修飾名です。
BOOL DeleteObject(LPCTSTR lpszPassword);	パラメータ = LPCTSTR lpszPassword	オブジェクトのパスワードです。NULL にで きます。
BOOL DeleteObject	BOOL	オブジェクトが削除されれば TRUE。
BOOL Launch(LPCTSTR lpszFQName);	パラメータ = LPCTSTR lpszFQName 戻り値 = BOOL	リモートオブジェクトファイルを起動して pcAnywhere リモート端末ウィンドウを開き ます。 パラメータは、起動するオブジェクトのリモート オブジェクトファイルの完全修飾名です。 オブジェクトが正常に起動した場合の戻り値 は TRUE です。

CRemoteData オブジェクト

このオブジェクトはリモートオブジェクトデータを修正するために使います。

Get メソッドと Set メソッド

次に示すメソッドは CRemoteData オブジェクトのプロパティを取得または設定するために使います。

コンピュータ名はリモートオブジェクトが起動したときに呼び出される **pcAnywhere** ホストコンピュータの名前です。

```
BSTR GetComputerName();
void SetComputerName(LPCTSTR lpszNewValue);
```

電話番号は **pcAnywhere** ホストコンピュータとのモデム接続を確立するためにダイヤルする番号です。

```
BSTR GetPhoneNumber();
void SetPhoneNumber(LPCTSTR lpszNewValue);
```

TAPI ダイヤルプロパティを使うか(所在地情報) (**TRUE**)、電話番号の文字列を表示どおりにそのまま使うか (**FALSE**) を示します。

```
BOOL GetUseDialingProperties();
void SetUseDialingProperties(BOOL bNewValue);
```

ダイヤルプロパティを使う場合、この値が呼び出し番号の市外局番になります。

```
BSTR GetAreaCode();
void SetAreaCode(LPCTSTR lpszNewValue);
```

ダイヤルプロパティを使う場合、この値が呼び出し番号の国番号になります。

```
BSTR GetCountryCode();
void SetCountryCode(LPCTSTR lpszNewValue);
```

呼び出しに失敗したときにこの番号のダイヤルを再試行する回数です。

```
short GetRedialCount();
void SetRedialCount(short nNewValue);
```

リダイヤルの試行間の待ち時間(秒単位)です。

```
short GetRedialDelay();
void SetRedialDelay(short nNewValue);
```

接続するときにホストに送信されるログイン名です。この値が空のままの場合は接続時にユーザーにログイン名を要求するメッセージが表示されます。

```
BSTR GetAutoLoginName();
void SetAutoLoginName(LPCTSTR lpszNewValue);
```

接続するときにホストに送信されるログインパスワードです。この値が空のままの場合は接続時にユーザーにパスワードを要求するメッセージが表示されます。

```
BSTR GetAutoLoginPassword();
void SetAutoLoginPassword(LPCTSTR lpszNewValue);
```

このオブジェクトのパスワードです。

```
BSTR GetPassword();  
void SetPassword(LPCTSTR lpszNewValue);
```

パスワードを使った場合にのみオブジェクトを起動できます (TRUE)。

```
BOOL GetExecuteProtection();  
void SetExecuteProtection(BOOL bNewValue);
```

正しいパスワードを入力した場合にのみオブジェクトを表示できます (TRUE)。

```
BOOL GetReadProtection();  
void SetReadProtection(BOOL bNewValue);
```

正しいパスワードを入力した場合にのみオブジェクトを書き込むことができます (TRUE)。

```
BOOL GetWriteProtection();  
void SetWriteProtection(BOOL bNewValue);
```

このオブジェクトを使うセッションをログに記録するかどうかを制御します。

```
BOOL GetLogSession();  
void SetLogSession(BOOL bNewValue);
```

このオブジェクトを使うセッションを最初から記録するかどうかを制御します。

```
BOOL GetRecordSession();  
void SetRecordSession(BOOL bNewValue);
```

このオブジェクトを使うセッションの記録ファイルの名前です。

```
BSTR GetRecordFile();  
void SetRecordFile(LPCTSTR lpszNewValue);
```

接続の種類を設定します。

```
BOOL GetRunOnConnect();  
void SetRunOnConnect(BOOL bNewValue);
```

利用可能な接続の種類は次のとおりです。

- COM1
- COM2
- COM3
- COM4
- ISDN via CAPI 2.0 (日本の環境ではご利用いただけません)
- NetBIOS

- SPX
- DEFAULT TAPI
- TCP/IP

TAPI デバイスの名前も接続の種類として使えます。DEFAULT TAPI はシステムで見つかった最初の TAPI デバイスを使います。特定の TAPI デバイスを使うには利用可能なデバイスを `FirstConnectionType()` または `NextConnectionType()` で検索します。

リモートオブジェクトの詳細メソッド

リモートオブジェクトに接続の種類を割り当てると、デバイスの詳細が有効なデフォルト値に設定されます。

次に示す接続の種類にはアプリケーションで設定できる高度な設定オプションがあります。

- COM デバイス
- NetBIOS デバイス
- ISDN via CAPI 2.0 (European ISDN のみ) デバイス

COM デバイスの詳細

通信パリティレベルを設定します。

```
BSTR GetComParity();  
void SetComParity(LPCTSTR lpszNewValue);
```

通信パリティ値は次のとおりです。

- None
- Odd
- Even
- Mark
- Space

フロー制御レベルを設定します。

```
BSTR GetComFlowControl();  
void SetComFlowControl(LPCTSTR lpszNewValue);
```

フロー制御値は次のとおりです。

- <None>
- XONXOFF

■ RTS/CTS

■ BOTH

接続開始設定を設定します。

```
BSTR GetComStartedBy();  
void SetComStartedBy(LPCTSTR lpszNewValue);
```

接続開始値は次のとおりです。

- Always connected
- Carrier detect (DCD)
- Clear to send (CTS)
- Data set ready (DSR)
- Ring indicator (RI)
- Receive 2 <CR>'s
- Modem response

接続終了値を設定します。

```
BSTR GetComEndedBy();  
void SetComEndedBy(LPCTSTR lpszNewValue);
```

接続終了値は次のとおりです。

- Always connected
- Carrier detect (DCD)
- Clear to send (CTS)
- Data set ready (DSR)
- Ring indicator (RI)

接続速度を設定します。

```
long GetComSpeed();  
void SetComSpeed(long nNewValue);
```

接続速度の値は次のとおりです。

- 110
- 300
- 600
- 1200

- 2400
- 4800
- 9600
- 19200
- 38400
- 57600
- 115200

NetBIOS デバイスの詳細

この接続に使う LAN アダプタ (LANA) 番号を設定します。

```
short GetLanaNumber();  
void SetLanaNumber(short nNewValue);
```

ISDN via CAPI 2.0 デバイスの詳細

TRUE の場合はチャネル結合 (1 つの接続に 2 つの ISDN チャネルを使う) を有効にします。

```
BOOL GetCapiChannelBonding();  
void SetCapiChannelBonding(BOOL bNewValue);
```

通信に必要な追加の CAPI エクステンションを設定します。

```
BSTR GetCapiExtensions();  
void SetCapiExtensions(LPCTSTR lpszNewValue);
```

リモートオブジェクトのメソッド

リモートオブジェクトの通常メソッドを次の表に示します (これらのメソッドではプロパティの取得と設定はできません)。

メソッド	パラメータまたは 戻り値	説明
short ConnectionTypes();	戻り値 = Short	このコンピュータで利用可能な接続の種類 の数です。

メソッド	パラメータまたは 戻り値	説明
BSTR FirstConnectionType(); と BSTR NextConnectionType();	戻り値 = BSTR	FirstConnectionType() と NextConnectionType() は利用可能な接続 の種類を通して繰り返すために使います。こ の関数は利用可能な接続の種類の名前で ある BSTR を返します。返された接続の種 類は SetConnectionType() 関数で使えま す。 戻り値はサポートされる接続デバイスの種類 の名前です。
BOOL FindConnectionType(LPCTSTR lpszConnectionType);	パラメータ = LPCTSTR lpszConnectionTyp 戻り値 = BOOL	接続デバイスの種類の名前です。 渡された接続の種類がコンピュータに存在 する場合には戻り値は TRUE です。
short CountryCodes();	戻り値 = Short	利用可能な国番号の数を返します。
BSTR FirstCountryCode(); と BSTR NextCountryCode();	戻り値 = BSTR	FirstCountryCode() と NextCountryCode() は利用可能な国番号を通して繰り返すため に使います。この関数は利用可能な国番号 の名前である BSTR を返します。戻り値は SetCountryCode() 関数で使えます。 戻り値は最初または次の国番号の文字列で す。
BOOL ReadObject(LPCTSTR lpszPassword);	パラメータ = LPCTSTR lpszPassword 戻り値 = BOOL	リモートオブジェクトファイルからオブジェクト データを読み込みます。 パラメータはこのオブジェクトのパスワードで す。 オブジェクトの読み込みが正常に完了した 場合には戻り値は TRUE です。
BOOL WriteObject(LPCTSTR lpszPassword);	パラメータ = LPCTSTR lpszPassword 戻り値 = BOOL	リモートオブジェクトファイルにオブジェクト データを書き込みます。 パラメータはこのオブジェクトのパスワードで す。 オブジェクトの書き込みが正常に完了した場 合には戻り値は TRUE です。

CRemoteDataEx オブジェクト

CRemoteDataEx オブジェクトは CRemoteData オブジェクトと同じ機能と次に示す追加の Get メソッドと Set メソッドを備えています。

```
BSTR GetPrivateKey(); // PrivateKey 情報を返します
void SetPrivateKey(LPCTSTR lpszNewValue);
BSTR GetCertificationName(); // 認証名を返します
void SetCertificationName(LPCTSTR lpszNewValue);
short GetEncryptionLevel(); // 暗号化レベルの値を返します
void SetEncryptionLevel(short nNewValue);
BOOL GetDenyLowerEncrypt(); //DenyLowerEncrypt の値を返します
void SetDenyLowerEncrypt(BOOL bNewValue);
BSTR GetAutoDomain(); //AutoDomain の値を返します
void SetAutoDomain(LPCTSTR lpszNewValue);
```

リモート機能のための Visual C++ サンプルコード

次に示す C++ 関数の例はリモートオブジェクトを作成し、その接続の種類を TCP/IP に設定し、コンピュータ名を関数に渡される TCP/IP アドレスに設定してからリモートオブジェクトを起動します。

```
BOOL LaunchTCPRemote(LPCTSTR lpszAddress)
{
    BOOL bReturn = FALSE;

    CRemoteDataManager remoteDM;

    CRemoteData remoteData;

    // まず、CRemoteDataManager を作成
    remoteDM.CreateDispatch( _T( "WINAHSV.R.RemoteDataManager" ) );
    // 次に、CRemoteData を作成してアタッチ
    remoteData.AttachDispatch( remoteDM.CreateObject( "Test", 0 ) );

    // ここで必要なプロパティを設定
    remoteData.SetConnectionType( "TCP/IP" );

    remoteData.SetComputerName( lpszAddress );
```

```
// オブジェクトデータを保存
if (remoteData.WriteObject(0))
{
// そのオブジェクトを起動
if (remoteData.Launch())
    bReturn = TRUE;
}
// リモートオブジェクトを解放
remoteData.ReleaseDispatch();

remoteDM.ReleaseDispatch( _T( "WINAWSVR.RemoteDataManager" ) );

return bReturn;
}
```

CHostDataManager メソッド

次の表に CHostDataManager メソッドを示します。これらのメソッドには、CHostData オブジェクトにアクセスしてそれらを制御するためのパラメータと戻り値があります。

メソッド	パラメータまたは戻り値	説明
BSTR CurrentDirectory();	戻り値 = BSTR	pcAnywhere ホストオブジェクトが格納されているカレントディレクトリの絶対パス名を返します。
BOOL ChangeDirectory(LPCTSTR lpszNewDirectory);	パラメータ = LPCTSTR lpszNewDirectory 戻り値 = BOOL	pcAnywhere ホストオブジェクトが格納されているカレントディレクトリを変更します。 パラメータは既存ディレクトリの名前です。 成功した場合の戻り値は TRUE です。

メソッド	パラメータまたは戻り値	説明
BOOL FindFirst(LPCTSTR lpszPattern);	パラメータ = LPCTSTR lpszPattern 戻り値 = BOOL	指定されたファイル名パターンに基づいてカレントディレクトリの最初の pcAnywhere ホストオブジェクトファイル(*.bhf)を見つけます。 パラメータは、オブジェクトファイルをフィルタ処理するファイル名パターンです。アスタリスク(*)はカレントディレクトリのすべてのホストファイルを見つけます。 指定されたパターンに一致するホストオブジェクトファイルが見つかった場合の戻り値は TRUE です。一致するファイルの絶対パス名は pbstrFullQualName に格納されます。
BOOL FindFirst(BSTR FAR* pbstrFullQualName);	パラメータ = BSTR FAR * pbstrFullQualName	指定されたパターンに一致するホストオブジェクトファイルの絶対パス名のバッファを返します。 指定されたパターンに一致するホストオブジェクトファイルが見つかった場合の戻り値は TRUE です。一致するファイルの絶対パス名は pbstrFullQualName に格納されます。
BOOL FindNext(BSTR FAR* pbstrFullQualName);	パラメータ = BSTR FAR * pbstrFullQualName 戻り値 = BOOL	FindFirst() の呼び出しが成功してカレントディレクトリにあるホストオブジェクトファイルの名前を取得したら、FindNext() を呼び出してパターンに一致する次のファイル (もしあれば) を検索できます。 FindFirst() に対する元の呼び出しで指定したパターンに一致するホストオブジェクトファイルの絶対パス名のバッファを返します。 FindFirst() の呼び出しで指定したパターンに一致する別のホストオブジェクトファイルが見つかった場合の戻り値は TRUE です。一致するファイルの絶対パス名は pbstrFullQualName に格納されます。
LPDISPATCH RetrieveObject(LPCTSTR lpszFQName);	パラメータ = LPCTSTR lpszFQName	CHostData オブジェクトをファイル名で取得します。 パラメータは、ロードするホストオブジェクトファイルの完全修飾名です。

メソッド	パラメータまたは戻り値	説明
LPDISPATCH RetrieveObject(short wAccessMode);	パラメータ = short wAccessMode	このオブジェクトの使い方を指定します。これはパスワード保護に関係します。次のオプションがあります。 ■ 0 = 指定なし ■ 1 = 表示のみ ■ 2 = 表示と修正 ■ 3 = 実行
LPDISPATCH RetrieveObject(LPCTSTR lpszPassword);	パラメータ = LPCTSTR lpszPassword	オブジェクトのパスワードです。NULL にできます。
LPDISPATCH RetrieveObject	戻り値 = LPDISPATCH	OLE ディスパッチオブジェクトを指すポインタ。オブジェクトは CHostData オブジェクトです。 「ホスト機能のための Visual C++ サンプルコード」を参照してください。
LPDISPATCH RetrieveObjectEx(LPCTSTR lpszFQName);	パラメータ = LPCTSTR lpszFQName	CHostDataEx オブジェクトをファイル名で取得します。 パラメータは、ロードするホストオブジェクトファイルの完全修飾名です。
LPDISPATCH RetrieveObjectEx(short wAccessMode);	パラメータ = short wAccessMode	このオブジェクトの使い方を指定します。これはパスワード保護に関係します。次のオプションがあります。 ■ 0 = 指定なし ■ 1 = 表示のみ ■ 2 = 表示と修正 ■ 3 = 実行
LPDISPATCH RetrieveObjectEx(LPCTSTR lpszPassword);	パラメータ = LPCTSTR lpszPassword	オブジェクトのパスワードです。NULL にできます。
LPDISPATCH RetrieveObjectEx	戻り値 = LPDISPATCH	OLE ディスパッチオブジェクトを指すポインタ。オブジェクトは CHostDataEx オブジェクトです。 「ホスト機能のための Visual C++ サンプルコード」を参照してください。

メソッド	パラメータまたは戻り値	説明
LPDISPATCH CreateObject(LPCTSTR lpszName);	パラメータ = LPCTSTR lpszFQName 戻り値 = LPDISPATCH	CHostData オブジェクトを作成してそれを 指す LPDISPATCH ポインタを返します。 パラメータは新しいオブジェクトのホストオ ブジェクトファイルの完全修飾名です。 戻り値は、OLE ディスパッチオブジェクトを 指すポインタです。オブジェクトは CHostData オブジェクトです。 「ホスト機能のための Visual C++ サンプル コード」 を参照してください。
LPDISPATCH CreateObjectEx(LPCTSTR lpszName);	パラメータ = LPCTSTR lpszFQName 戻り値 = LPDISPATCH	CHostDataEx オブジェクトを作成してそれ を指す LPDISPATCH ポインタを返します。 パラメータは新しいオブジェクトのホストオ ブジェクトファイルの完全修飾名です。 戻り値は、OLE ディスパッチオブジェクトを 指すポインタです。オブジェクトは CHostDataEx オブジェクトです。 「ホスト機能のための Visual C++ サンプル コード」 を参照してください。
BOOL DeleteObject(LPCTSTR lpszFQName);	パラメータ = LPCTSTR lpszFQName	ホストオブジェクトファイルを削除します。 パラメータは、削除するオブジェクトのホス トオブジェクトファイルの完全修飾名です。
BOOL DeleteObject(LPCTSTR lpszPassword);	パラメータ = LPCTSTR lpszPassword	オブジェクトのパスワードです。NULL にで きます。
BOOL DeleteObject	戻り値 = BOOL	オブジェクトが削除されれば TRUE。
BOOL Launch(LPCTSTR lpszFQName);	パラメータ = LPCTSTR lpszFQName 戻り値 = BOOL	ホストオブジェクトファイルを起動して pcAnywhere ホスト端末ウィンドウを開きま す。 パラメータは、起動するオブジェクトのホス トオブジェクトファイルの完全修飾名です。 オブジェクトが正常に起動した場合の戻り 値は TRUE です。

CHostData オブジェクト

このオブジェクトはホストオブジェクトデータを修正するために使います。

Get メソッドと Set メソッド

次に示すメソッドはCHostDataオブジェクトのプロパティを取得または設定するために使います。

電話番号は **pcAnywhere** リモートコンピュータとのモデム接続を確立するためにダイヤルする番号です。

```
BSTR GetPhoneNumber();  
void SetPhoneNumber(LPCTSTR lpszNewValue);
```

TAPI ダイヤルプロパティを使うか(所在地情報) (**TRUE**)、電話番号の文字列を表示どおりにそのまま使うか (**FALSE**) を示します。

```
BOOL GetUseDialingProperties();  
void SetUseDialingProperties(BOOL bNewValue);
```

ダイヤルプロパティを使う場合、この値が呼び出し番号の市外局番になります。

```
BSTR GetAreaCode();  
void SetAreaCode(LPCTSTR lpszNewValue);
```

ダイヤルプロパティを使う場合、この値が呼び出し番号の国番号になります。

```
BSTR GetCountryCode();  
void SetCountryCode(LPCTSTR lpszNewValue);
```

呼び出しに失敗したときにこの番号のダイヤルを再試行する回数です。

```
short GetRedialCount();  
void SetRedialCount(short nNewValue);
```

リダイヤルの試行間の待ち時間(秒単位)です。

```
short GetRedialDelay();  
void SetRedialDelay(short nNewValue);
```

このオブジェクトを使うセッションをログに記録するかどうかを制御します。

```
BOOL GetLogSession();  
void SetLogSession(BOOL bNewValue);
```

このオブジェクトを使うセッションを最初から記録するかどうかを制御します。

```
BOOL GetRecordSession();  
void SetRecordSession(BOOL bNewValue);
```

このオブジェクトを使うセッションの記録ファイルの名前です。

```
BSTR GetRecordFile();  
void SetRecordFile(LPCTSTR lpszNewValue);
```

ホストオブジェクトの詳細メソッド

ホストオブジェクトに接続の種類を割り当てると、デバイスの詳細が有効なデフォルト値に設定されます。

次に示す接続の種類にはアプリケーションで設定できる高度な設定オプションがあります。

- COM デバイス
- NetBIOS デバイス
- NASI/NCSI デバイス
- ISDN via CAPI 2.0 (European ISDN のみ) デバイス

COM デバイスの詳細

次のコードは要求された接続の種類をホストオブジェクトの割り当て済み接続の種類のリストに入れ、以降のデバイス固有のメソッド呼び出しを処理するときにその接続の種類を現在の接続の種類にします。

```
BOOL AssignConnection(LPCTSTR lpszNewValue);
```

要求された接続の種類が割り当て済み接続のリストにすでにある場合、割り当て済み接続のリストは変更されません。現在の接続の種類が要求された種類に変わるだけです。接続固有の値の取得や設定をするときに同じオブジェクトで **AssignConnection** メソッドを複数回呼び出すことは正常です。

AssignConnection は渡された接続の種類がコンピュータに存在しその割り当てに成功した場合またはそれがすでに割り当てられている場合は **TRUE** を返します。要求された接続の種類がコンピュータに存在しない場合または現在の割り当て済み接続数がすでに最大許容レベルになっている場合は **FALSE** を返します。

pcAnywhere ホストオブジェクトは割り当て済み接続の種類を 2 つまでサポートできません。**AssignConnection** メソッドはこの限度を超える試行を検出した場合は **FALSE** を返します。

利用可能な接続の種類は次のとおりです。

- COM1

- COM2
- COM3
- COM4
- SPX
- NetBIOS
- TCP/IP
- ISDN via CAPI 2.0 (日本の環境ではご利用いただけません)
- DEFAULT TAPI

TAPI デバイスの名前も接続の種類として使えます。DEFAULT TAPI はシステムで見つかった最初の TAPI デバイスを使います。特定の TAPI デバイスを使うには利用可能なデバイスを `FirstConnectionType()` と `NextConnectionType()` で検索します。

次のコードは接続の種類を割り当て解除します。接続の種類を割り当て解除後、以降のデバイス固有のメソッド呼び出しでは残りの割り当て済み接続 (存在する場合) が現在の接続の種類になります。

```
BOOL UnassignConnection(LPCTSTR lpszNewValue);
```

次のコードは通信パリティレベルを設定します。

```
BSTR GetComParity();  
void SetComParity(LPCTSTR lpszNewValue);
```

通信パリティ値は次のとおりです。

- None
- Odd
- Even
- Mark
- Space

次のコードはフロー制御レベルを設定します。

```
BSTR GetComFlowControl(); void SetComFlowControl(LPCTSTR  
lpszNewValue);
```

フロー制御値は次のとおりです。

- <None>
- XONXOFF
- RTS/CTS

■ BOTH

次のコードは接続開始値を設定します。

```
BSTR GetComStartedBy();  
void SetComStartedBy(LPCTSTR lpszNewValue);
```

接続開始値は次のとおりです。

- Always connected
- Carrier detect (DCD)
- Clear to send (CTS)
- Data set ready (DSR)
- Ring indicator (RI)
- Receive 2 <CR>'s
- Modem response

次のコードは接続終了値を設定します。

```
BSTR GetComEndedBy();  
void SetComEndedBy(LPCTSTR lpszNewValue);
```

接続終了値は次のとおりです。

- Always connected
- Carrier detect (DCD)
- Clear to send (CTS)
- Data set ready (DSR)
- Ring indicator (RI)

次のコードは接続速度を設定します。

```
long GetComSpeed();  
void SetComSpeed(long nNewValue);
```

接続速度の値は次のとおりです。

- 110
- 300
- 600
- 1200
- 2400

- 4800
- 9600
- 19200
- 38400
- 57600
- 115200

NetBIOS デバイスの詳細

この接続に使う LAN アダプタ (LANA) 番号を設定します。

```
short GetLanaNumber(); void SetLanaNumber(short nNewValue);
```

NASi/NCSI デバイスの詳細

NASi サーバーにユーザー名を設定します。

```
BSTR GetNasiUserName();  
void SetNasiUserName(LPCTSTR lpszNewValue);
```

NASi サーバーにユーザーパスワードを設定します。

```
BSTR GetNasiPassword();  
void SetNasiPassword(LPCTSTR lpszNewValue);
```

NASi セッション名を設定します。

```
BSTR GetNasiSessionName();  
void SetNasiSessionName(LPCTSTR lpszNewValue);  
BOOL GetNasiSessionNameAvailable();  
void SetNasiSessionNameAvailable(BOOL bNewValue);
```

使う NASi サーバーを指定します。

```
BOOL NasiServer();  
BSTR GetNasiServerName();  
void SetNasiServerName(LPCTSTR lpszNewValue);
```

使う NASi サービスを指定します。

```
BOOL NasiService();  
BSTR GetNasiServiceName();  
void SetNasiServiceName(LPCTSTR lpszNewValue);
```

使う NASi ポートを指定します。

```
BOOL NasiPort();  
BSTR GetNasiPortName();  
void SetNasiPortName(LPCTSTR lpszNewValue);  
BOOL GetNasiSelectOnConnect();  
void SetNasiSelectOnConnect(BOOL bNewValue);
```

ISDN via CAPI 2.0 デバイスの詳細

TRUE の場合はチャネル結合 (1 つの接続に 2 つの ISDN チャネルを使う) を有効にします。

```
BOOL GetCapiChannelBonding();  
void SetCapiChannelBonding(BOOL bNewValue);
```

通信に必要な追加の CAPI エクステンションを設定します。

```
BSTR GetCapiExtensions();  
void SetCapiExtensions(LPCTSTR lpszNewValue);
```

ホストオブジェクトのメソッド

オブジェクトの通常の方法を次の表に示します (これらの方法ではプロパティの取得と設定はできません)。

メソッド	パラメータまたは 説明 戻り値	
short ConnectionTypes();	戻り値 = Short	このコンピュータで利用可能な接続の種類の数を返します。
BSTR FirstConnectionType(); と BSTR NextConnectionType();	戻り値 = BSTR	FirstConnectionType() と NextConnectionType() は利用可能な接続の種類を通して繰り返すために使います。これらの関数は利用可能な接続の種類の名前である BSTR を返します。返された接続の種類は SetConnectionType() 関数で使えます。 戻り値はサポートされる接続デバイスの種類の名前です。
BOOL FindConnectionType(LPCTSTR lpszConnectionType);	パラメータ = LPCTSTR lpszConnectionType 戻り値 = BOOL	パラメータは、接続デバイスの種類の名前です。 戻り値は、このデバイスの種類が利用可能な場合は TRUE です。

メソッド	パラメータまたは 戻り値	説明
short MaxAssignedConnections()	戻り値 = Short	同時に割り当てられる接続の種類の最大数 (現バージョンでは 2) を返します。
short AssignedConnections()	戻り値 = Short	割り当て済み接続の種類の数を返します。
BSTR FirstAssignedConnection(); と BSTR NextAssignedConnection();	戻り値 = BSTR	FirstAssignedConnection() と NextAssignedConnection() は割り当て済み 接続のリストを通して繰り返すために使いま す。これらの関数は割り当て済み接続の種類 の名前である BSTR を返します。返された接 続の種類は AssignConnection() 関数で使 えます。 戻り値はサポートされる接続デバイスの種類 の名前です。
BOOL FindAssignedConnection (LPCTSTR lpszConnectionType);	パラメータ = LPCTSTR lpszConnectionTyp 戻り値 = BOOL	パラメータは、接続デバイスの種類の名前で す。 戻り値は、このデバイスの種類が現在割り当 て済み場合は TRUE です。
short CountryCodes();	戻り値 = Short	利用可能な国番号の数を返します。
BSTR FirstCountryCode(); と BSTR NextCountryCode();	戻り値 = BSTR	FirstCountryCode() と NextCountryCode() は利用可能な国番号のリストを通して繰り返 すために使います。これらの関数は利用可能 な国番号の名前である BSTR を返します。戻 り値は SetCountryCode() 関数で使えます。 戻り値は最初または次の国番号の文字列で す。
BOOL ReadObject(LPCTSTR lpszPassword);	パラメータ = LPCTSTR lpszPassword 戻り値 = BOOL	ホストオブジェクトファイルからオブジェクトデー タを読み込みます。 パラメータはこのオブジェクトのパスワードで す。 オブジェクトの読み込みが正常に完了した場 合には戻り値は TRUE です。

メソッド	パラメータまたは 戻り値	説明
BOOL WriteObject(LPCTSTR lpszPassword);	パラメータ = LPCTSTR lpszPassword 戻り値 = BOOL	ホストオブジェクトファイルにオブジェクトデー タを書き込みます。 パラメータはこのオブジェクトのパスワードで す。 オブジェクトの書き込みが正常に完了した場 合には戻り値は TRUE です。

CHostDataEx オブジェクト

CHostDataEx オブジェクトは CHostData オブジェクトと同じ機能と次に示す追加の Get メソッドと Set メソッドを備えています。

```

BOOL GetReadProtection();
void SetReadProtection(BOOL bNewValue);
BOOL GetWriteProtection();
void SetWriteProtection(BOOL bNewValue);
BSTR GetPassword(); // "NOT IMPLEMENTED" を返します
void SetPassword(LPCTSTR lpszNewValue);
BSTR GetCallersPath();
void SetCallersPath(LPCTSTR lpszNewValue);
BOOL GetConfirmConnect();
void SetConfirmConnect(BOOL bNewValue);
short GetConfirmTimeout();
void SetConfirmTimeout(short nNewValue);
BOOL GetConfirmDeny();
void SetConfirmDeny(BOOL bNewValue);
BOOL GetPwCaseSensitive();
void SetPwCaseSensitive(BOOL bNewValue);
short GetPwAttempts();
void SetPwAttempts(short nNewValue);
short GetPwTimeout();
void SetPwTimeout(short nNewValue);
short GetActiveKbds();
void SetActiveKbds(short nNewValue); // ActiveKbds を設定します
short GetInactiveTimeout();
void SetInactiveTimeout(short nNewValue);
short GetCryptReqLevel();
void SetCryptReqLevel(short nNewValue);
BOOL GetCryptRefuseLower();

```

```
void SetCryptRefuseLower(BOOL bNewValue);
short GetAuthenticationType();
void SetAuthenticationType(short nNewValue);
BOOL GetLockSystemWhileWait();
void SetLockSystemWhileWait(BOOL bNewValue);
BOOL GetMinimizeOnLaunch();
void SetMinimizeOnLaunch(BOOL bNewValue);
BOOL GetRunAsService();
void SetRunAsService(BOOL bNewValue);
short GetConnLostWait();
void SetConnLostWait(short nNewValue);
BOOL GetConnLostHostOpts();
void SetConnLostHostOpts(BOOL bNewValue);
BOOL GetEnableConnLostSecurity();
void SetEnableConnLostSecurity(BOOL bNewValue);
short GetConnLostSecurity();
void SetConnLostSecurity(short nNewValue);
short GetCallbkDelay();
void SetCallbkDelay(short nNewValue);
BOOL GetEndSessHostOpts();
void SetEndSessHostOpts(BOOL bNewValue);
BOOL GetEnableEndSessSecurity();
void SetEnableEndSessSecurity(BOOL bNewValue);
short GetEndSessSecurity();
void SetEndSessSecurity(short nNewValue);
BSTR GetCryptPrivateKey();
void SetCryptPrivateKey(LPCTSTR lpszNewValue);
BSTR GetCryptCommonName();
void SetCryptCommonName(LPCTSTR lpszNewValue);
BOOL GetBlankHost();
void SetBlankHost(BOOL bNewValue);
BOOL GetAllowRemoteMouse();
void SetAllowRemoteMouse(BOOL bNewValue);
short GetRebootOnDisconnect();
void SetRebootOnDisconnect(short nNewValue);
BOOL GetPasswordAfterDisc();
void SetPasswordAfterDisc(BOOL bNewValue);
BOOL GetLogFailures();
void SetLogFailures(BOOL bNewValue);
BOOL GetAllowDriveSecurity();
void SetAllowDriveSecurity(BOOL bNewValue);
BOOL GetExecuteProtection();
void SetExecuteProtection(BOOL bNewValue);
```

ホスト機能のための Visual C++ サンプルコード

次に示す C++ 関数の例は、ホストオブジェクトを作成してその接続の種類を TCP/IP に設定し、コンピュータ名を関数に渡される TCP/IP アドレスに設定してからホストオブジェクトを起動します。

```
BOOL LaunchTCPHost(LPCTSTR lpszAddress)
{
    BOOL bReturn = FALSE;

    CHostDataManager hostDM;
    CHostData hostData;

    // まず、CHostDataManager を作成
    hostDM.CreateDispatch( _T( "WINAWSVR.BeHostDataManager" ) );

    // 次に、CRemoteData を作成してアタッチ
    hostData.AttachDispatch(hostDM.CreateObject("Test", 0) );

    // ここで必要なプロパティを設定
    hostData.SetConnectionType("TCP/IP");

    // オブジェクトデータを保存
    if (hostData.WriteObject(0))
    {
        // そのオブジェクトを起動
        if (hostData.Launch())
            bReturn = TRUE;
    }

    // ホストオブジェクトを解放
    hostData.ReleaseDispatch();

    return (bReturn);
}
```

Awrem32 関数

次の表に Awrem32 関数を示します。これらの関数には、ホストコンピュータとリモートコンピュータの間の接続を処理するためのパラメータと戻り値があります。

メソッド	パラメータまたは 戻り値	説明
<code>boolean awConnect(BSTR FileName);</code>	パラメータ = Name (文字列) 戻り値 = Boolean	ホストコンピュータへの接続を作成します。 パラメータは、ホストコンピュータについての 情報を含む .chf ファイルの完全修飾名で す。 戻り値によってコマンドが実行されます。
<code>boolean awDisconnect();</code>	戻り値 = Boolean	ホストコンピュータを接続解除します。 呼び出し側のプログラムはこの関数の呼び 出し後にオブジェクトを削除 (C++ の場合は IAwrem32X* を削除、VB の場合は ObjectName = Nothing; を設定) する必要 があります。
<code>boolean FileXferFromHost(BSTR HostFile);</code>	パラメータ = HostFile (文字列)	ホストコンピュータからリモートコンピュータ にファイルをコピーします。パラメータには ワイルドカード文字を使うことができます。 パラメータは、コピー元のホストコンピュータ のパスとファイルの完全修飾名です。 HostFile と RemoteFile の文字列は異な る必要があります。
<code>boolean FileXferFromHost(BSTR RemoteFile);</code>	パラメータ = RemoteFile (文字 列)	ホストコンピュータからリモートコンピュータ にファイルをコピーします。パラメータには ワイルドカード文字を使うことができます。 コピー先のパスとファイルの完全修飾名で す。 HostFile と RemoteFile の文字列は異 なる必要があります。
<code>boolean FileXferFromHost</code>	戻り値 = Boolean	コマンドが実行された場合は TRUE です。
<code>boolean FileXferToHost(BSTR HostFile);</code>	パラメータ = HostFile (文字列)	リモートコンピュータからホストコンピュータ にファイルをコピーします。パラメータには ワイルドカード文字を使うことができます。 コピー先のパスとファイルの完全修飾名で す。 HostFile と RemoteFile の文字列は異 なる必要があります。

メソッド	パラメータまたは 戻り値	説明
<code>boolean FileXferToHost(BSTR RemoteFile);</code>	パラメータ = <code>RemoteFile</code> (文字 列)	リモートコンピュータからホストコンピュータ にファイルをコピーします。パラメータには ワイルドカード文字を使うことができます。 パラメータは、コピー元のリモートコンピュ ータのパスとファイルの完全修飾名です。 <code>HostFile</code> と <code>RemoteFile</code> の文字列は異な る必要があります。
<code>boolean FileXferToHost</code>	戻り値 = <code>Boolean</code>	コマンドが実行された場合は <code>TRUE</code> です。
<code>boolean CreateFolderOnHost(BSTR FolderName);</code>	パラメータ = <code>FolderName</code> (文字 列) 戻り値 = <code>Boolean</code>	ホストコンピュータ上に新しいフォルダを作 成します。この関数は、リモートコンピュ ータ上に一時フォルダを作成してからそのフ ォルダをホストコンピュータにコピーします。 パラメータは、ホストコンピュータ上に作成 するフォルダのドライブとパスです。 コマンドが実行された場合の戻り値は <code>TRUE</code> です。
<code>boolean ExecuteHostFile(BSTR FileName);</code>	パラメータ = <code>FileName</code> (文字列) 戻り値 = <code>Boolean</code>	ホストコンピュータ上で既存のファイルを実 行します。この関数が実行するのは、バッチ ファイル、コマンドファイル、実行可能フ ァイルのみです。実行可能ファイルに関連付 けられたファイルは実行しません。たとえ ば、 <code>.doc</code> ファイルを実行しても <code>Microsoft Word</code> は起動しません。 ホストコンピュータ上のファイルの完全修飾 パスです。 コマンドが実行された場合の戻り値は <code>TRUE</code> です。
<code>BSTR GetError();</code>	戻り値 = <code>String</code>	<code>Awrem32</code> が生成した最後のエラーを文字 列として返します。
<code>short ConnectionStatus();</code>	戻り値 = <code>Short</code>	ホストコンピュータに対する接続の現在の 状態を返します。 有効な値は次のとおりです。 ■ -1 = 接続の消失 ■ 0 = 接続なし ■ 1 = 接続セッション

A

API ライブラリ 9
Awrem32
関数 50、79

C

C++。「Visual C++」を参照
CAPI 接続
ホストプロパティ 41、74
リモートプロパティ 25、62
CHostData
Visual Basic を使う 31
Visual C++ を使う 69
CHostDataEx
Visual Basic を使う 42
Visual C++ を使う 76
CHostDataManager
Visual Basic を使う 29
Visual C++ を使う 65
COM デバイス
ホストデバイスの詳細 37、70
リモートデバイスの詳細 21、60
CRemoteData
Visual Basic を使う 17
Visual C++ を使う 57
CRemoteDataEx
Visual Basic を使う 26
Visual C++ を使う 64
CRemoteDataManager
Visual Basic を使う 14
Visual C++ を使う 54

G

GUID 7

I

ISDN
CAPI 接続
ホストプロパティ 41、74
リモートプロパティ 25、62

N

NASI デバイス 73
NCSI デバイス 73
NetBIOS 接続
ホストプロパティ 40、73
リモートプロパティ 24、62

O

OLE オートメーション、。「pcAnywhere Automation Server」を参照

P

pcAnywhere Automation Server
GUID の登録 7
Visual Basic によるアクセス 8
Visual C++ によるアクセス 9
概要 6
使用例 6
タイプライブラリ 9

T

TAPI デバイス
ホスト上 71
リモート上 60

V

Visual Basic
Awrem32 関数 50
CHostDataEx オブジェクト 42
CHostDataManager オブジェクト 29
CHostData オブジェクト 31
CRemoteDataEx オブジェクト 26
CRemoteDataManager オブジェクト 14
CRemoteData オブジェクト 17
pcAnywhere Automation Server へのアクセス 8
コードサンプル 27、48
Visual C++
Awrem32 関数 79
CHostDataEx オブジェクト 76
CHostDataManager オブジェクト 65

- CHostData オブジェクト 69
- CRemoteDataEx オブジェクト 64
- CRemoteDataManager オブジェクト 54
- CRemoteData オブジェクト 57
- pcAnywhere Automation Server へのアクセス 9
- Winawsvr.h のインクルード 11
- クラスのインポート 10
- コードサンプル 64、78

W

Winawsvr

- オブジェクト
 - CHostData 31、69
 - CHostDataEx 42、76
 - CHostDataManager 29、65
 - CRemoteData 17、57
 - CRemoteDataEx 26、64
 - CRemoteDataManager 14、54
- 実行可能 8
- ヘッダーファイル 11

あ

暗号化の設定

- ホスト上 42、76
- リモート上 26～27、64

エラー

- 返す 51、80

オートメーションコントローラ

- Visual Basic 8
- Visual C++ 9

- オートメーションサーバー。「pcAnywhere Automation Server」を参照

か

国番号

- ホスト上 75
- リモート上 63

クラス定義

- インポート 10
- 表示 10

コードサンプル

- Visual Basic 27、48
- Visual C++ 64、78

さ

- 識別子。「GUID」を参照

状態

- 返す 51、80

接続

- エラーを返す 80
- 開始 31、50、57、79
- 終了 50、79
- 状態を返す 51、80

接続の種類

- ホストプロパティ 33、70、74
- リモートプロパティ 19、60、62
- 割り当て 35
- 割り当て解除 36

た

- タイプライブラリ 9

- デバイス。「接続の種類」を参照

な

ネットワーク接続

- ホスト上
 - NetBIOS のプロパティ 40、73
- リモート上
 - NetBIOS のプロパティ 24、62

は

パスワード

- ホストオブジェクト上 75
- リモートオブジェクト上 63

- バリティ 21、38、60、71

ファイル

- ホスト上での実行 51、80

ファイル転送

- ホストから 50、79
- ホストへ 50、79

フォルダ

- ホストへの作成 51、80

- フロー制御 22、38、60、71

ホストオブジェクト

- 起動 31、68
- 検索 29～30、66～67
- 削除 31、68
- 作成 68
- ダイヤルのプロパティを使ってモデム接続を設定する
るには 37
- ディレクトリ 29、65
- デバイスの詳細 70
- パスワード 75
- メソッド 74

ま

モデム接続

ホスト上

COM のプロパティ 37、70

TAPI デバイス 71

国番号 75

ダイヤルのプロパティを使ってモデム接続を設定するには 37

リモート上

COM のプロパティ 21、60

TAPI デバイス 60

国番号 63

ダイヤルのプロパティ 20

ら

リモートエンジン

自動登録 7

手動登録 8

リモートオブジェクト

削除 57

作成 56

ダイヤルのプロパティ 20

ディレクトリ 14、54

デバイスの詳細 60

パスワード 63

ファイル 15～16、29、55～56

メソッド 62