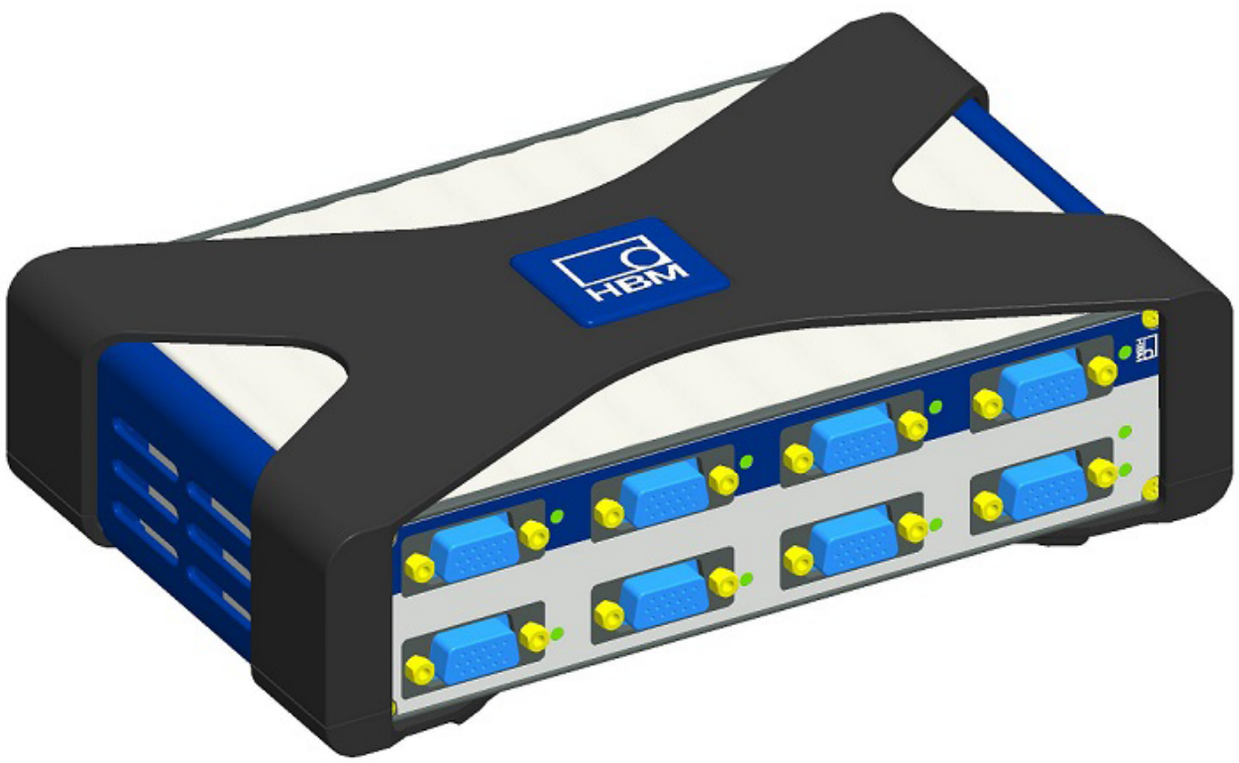
catman EasyScript Version 5.6.1
日本語訳

2023年9月 初版
2023年12月 第二版: リンクの修正



注意

- EasyScript日本語マニュアル(以下本書)は、Hottinger Brüel & Kjaer GmbHのQuantumX用catmanEasy/APに含まれているEasyScriptのヘルプ(EasyScriptHelp.chm)を日本語化したものです。
- 本書の著作権はアドバン オートメーション株式会社に属します。
- 本書に記述されている内容について、Hottinger Brüel & Kjaer GmbHおよびスペクトリス株式会社 HBK事業部(以下HBK社)は一切関与しません。本書についての問い合わせはアドバン オートメーション株式会社が受け付けます。
- 本書の内容に基づいて発生した負傷や損害に対しては、HBK社およびアドバン オートメーション株式会社は一切責任を負いません。
- 本書は、特定のアプリケーションに対する適合を保証していません。また、製品がどのような環境下でも正しく機能することを保証していません。製品とアプリケーションに関連したリスクを最小限に抑えるため、ユーザが適切な設計および保護対策を用意する必要があります。
- 本書に記載されている製品について将来予告することなしに変更することがあります。また本書の記述も予告なしに変更することがあります。
- 本書の一部または全部を複製、複写、翻訳、転載、テープ化などをすることはできません。
- QuantumXはHottinger Brüel & Kjaer GmbHの商標です。日本国内においてはスペクトリス株式会社 HBK事業部が保有しています。その他の製品名または商標は各社に帰属しています。

目次

注意・・・

AddIn ・・・

EasyScriptの概要

EasyScriptは、catmanEasy/APで監視および制御することのできるプログラミング言語です。EasyScriptは以下が行えます:

1. 独自の関数により、catmanEasy/APを拡張します。
   特定の時期に独自の関数を実行することができます。例えば、DAQジョブの開始や計測データの評価前に、Excelに計算値を転送します。関数は、より大きなEasyScriptプロジェクト内に属したり、ボタン、カスタム制御要素または Automatic Procedure Executionに割り当てることができます。後者のメカニズムはDirect Scriptと呼ばれています。DirectScriptコードは、小さく使いやすいエディタで記述されます。

2. 完全なプログラムを作成します。
   完全なデータ収集および解析プロセスを制御したい場合、このアプローチを使用します。このようなプログラムは、例えばボタンまたはプロジェクトの読み込み時に自動的に開始します。このようなプログラムはEasyScript 開発環境で作成します。

EasyScriptの実行方法は?

EasyScriptは、MicrosoftのOfficeパッケージで使用されているVBA標準(Visual Basic for Applications)に基づいています。VBAには一般的な言語要素が用意されています: 変数、制御フロー、数学など。EasyScriptは、必要なオブジェクトが利用可能で、catmanEasy/APの動作の制御を可能にします。VBAのオープンアーキテクチャであるため、他のプログラム、例えばExelやWordもオブジェクトとして対処することができます。

EasyScriptプロジェクトは、通常のプログラムと同様に実行することもでき、またユーザがボタンをクリックした場合に呼び出される個々の関数を含むコンテナとして見ることもできます。catmanの多くのオブジェクト(可視化オブジェクト、リボンのカスタマイズやAutomatic Procedure Execution)は、_Click eventにインスタンスの関数を指定する機会を得られます。

クリックに反応するように割り当てられたスクリプトプロシージャを持つボタンを有効にするには、スクリプトプロジェクトを開始する必要があります(**EasyScript → Run script project**)。定義済みのアクションまたは、AutoSequencesが割り当てられている場合のみ、ボタンをクリックすると開始します。

必要な予備知識は?

Visual BasicまたはVBAの基本的な知識を持っている必要があります。既にWordやExcelでマクロを組んだ経験があれば、それで十分です(マクロの記録だけでは不十分です)。そうでない場合、まず基本的な知識を適切な文献、例えばMicrosoft Pressから取得することを推奨します。

EasyScriptにはどのような可能性がありますか?

EasyScriptを使用すると、VBAでプログラムするための追加機能をcatmanEasy/APに拡張させることができます。典型的な例として、実行中のDAQジョブの計測値に特定の操作(例えば特別な解析)や、解析モードで複数の高度な数学が行えます。対話モードで基本構成(例えばセンサの割り当て)を行うか、EA.OpenProjectで既存のプロジェクトを読み込みます。

スクリプトを使用して可視化オブジェクトを作成することはできません。

一般的なcatman関数

EAオブジェクトは、catmanEasy/APの一般的な関数とメソッド(エラーステータスのクエリ、通知の送信、作業ディレクトリの取得など)を提供します。

DAQジョブの構成

EA_Jobを使用するとDAQ ジョブを開始/停止、ジョブ設定の変更や最初からジョブを作成することができます。

実行中DAQジョブの計測値の観察

EA_DAQを使用して、ジョブの実行中に測定データにアクセスします。値の監視、イベントの検出や、特別な計算が行えます。このオブジェクトは他にストレージ制御に対応(例えばストレージのOn/Off)、またはデータ レコーダーのアクション(記録の開始/停止など)を処理します。

既存のテストデータを解析

テスト中に記録されたデータを扱うすべての関数は、EA_Testにグループ化されています。テストの読み込み、チャネルのプロット、トレーサビリティデータの調査、データの計算、データのエクスポートまたは目的の形式への変換を行うことができます。

対話および可視化

EA_Panelは、テキストボックス、テーブルやリストボックスなどの入力オブジェクトをクエリしたり、メッセージを表示したりするメソッドを提供します。このオブジェクトはまたパネルの参照、パネルのエクスポートと印刷、デジタル表示器の値表示なども行うことができます。

EA_Graphは、グラフを構成(タイトル、軸等)、カーソル位置の取得、グラフへの注釈の追加/削除、グラフのエクスポート/印刷、グラフへのデータのプロットを行うための多数の関数を提供します。

EA_Ribbonは、リボン バーシステムに独自タブの追加、グループおよびコントロールを追加したり、ポップアップメニューの作成といった、多くの関数を提供します。これはEA_Panelの強力な代替案です。トピックの詳細はEasyScriptアプリケーションのユーザインタフェースを作成を参照してください。

独自のポップアップダイアログを作成するには、UserDialog editorを使用します。

機器の構成

EasyScriptの主なタスクではない場合でも、EA_IOオブジェクトを使用して、デバイスのスキャン、チャネル名の変更、チャネル属性の取得、新しいセンサの作成やセンサの割り当て、計算チャネルの作成/変更、または単一の測定値の取得を行うことができます。一部のデバイスに対してはローレベルコマンドを送信することができます(EA_IO.Command)。

EA_IOオブジェクトの二つの特別な関数は、QuantumXモジュールと直接通信を可能にします。

デバイスやファイルサーバーとの通信

EA_Commオブジェクトを使用すると、さまざまなインタフェース(TCP/IP、RS232、GPIB など)を介してデバイスと直接通信し、NTP/PTP同期ステータスを取得し、FTP/SFTPサーバにファイルをアップロードできます。このグループでは、基本的なファイル操作を実行するための多数の関数も利用できます。

数学ライブラリにアクセス

EA_Mathオブジェクトは、catmanEasy/APの数学ライブラリから多数の数学関数を提供します。

ビデオカメラ

EA_Videoオブジェクトを使用すると、ビデオキャプチャ(ビデオ/オーディオ、手動フレーム、または単一画像)を開始/停止したり、カメラ設定を変更したりできます。

基本的なDataView関数

EA_DataViewオブジェクトは、DataViewウィンドウを操作するための基本的な関数を提供します。DataViewにチャネルを追加します。

Webページサービス

EA_Webオブジェクトを使用すると、HTMLを使用してWebページを管理できます。

どのようにスクリプトを記述しますか?

必要な入出力のパネルを含む、計画的に計測デバイスと変換器を組み合わせた、計測プロジェクトを作成します。そして、**EasyScript → Editor**からエディタを起動します。

モジュールおよびプロジェクトには意味のある名前をつけます(または**File → Save project as**); 古いファイルは新規プロジェクトの作成により、上書きされる可能性があるため、デフォルト名のMODULE1.BASおよびPROJECT1.ESPは使用しないでください。

完全なスクリプトを実行するか、単一プロシージャを起動(例えばボタンをクリックして)するかに応じて、`Sub Main … End Sub`内に独自のサブプログラムを書き込むか(下図のサブプログラムの例を参照)、空白の`Sub Main … End Sub`セクションにします。これはもちろん可能であり(強く推奨しています)、完全な実行のために意図したサブプログラム(プロシージャ)を使用します: その場合、`Sub Main … End Sub`セクションには少なくとも一つのプロシージャが含まれています。プロシージャ内から他のプロシージャ、または同じプロシージャを再度呼び出すことが可能です(再帰呼出し)。

命令文を作成する場合、Code Builder(ウィンドウの左側)が助けになります: 関数またはアクションがクリックされた場合、カーソル位置に対応するコードを生成します。同時にScript Help(ウィンドウの下)に、それぞれのプロパティやメソッドの説明が表示さます。catmanEasy/APのオブジェクトは、その関数をさまざまなオブジェクトのスクリプトで利用できるようにします。各オブジェクトは特定の関数グループを意味し、スクリプトに対応するプロパティおよびメソッドを持ちます。すべてのオブジェクトは、EAsyScriptを表す接頭辞`EA_`から始まります。

これらのオブジェクトはスクリプトで自動的に利用可能で、VBAスクリプトの命令の`CreateObject`で作成する必要はありません。

すべてのVBAオブジェクトは、プロパティ、メソッドおよび`For … Next`や`If … End If`などの命令は、本書のVBAリファレンスに記載されています。

他のプログラムやオブジェクトにアクセスするには、それぞれのソフトウェア製造元よりオブジェクトのリファレンスを入手する必要があります。MicrosoftのOfficeプログラムは、Microsoft OfficeのWebページを参照してください。

テストの実行前に、スクリプトのプロジェクトを保存します。選択したスクリプトの形式に応じて、完全なEasyScriptプロジェクトの実行や、単一プロシージャを実行することができます。

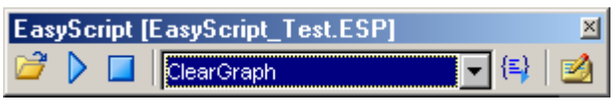
スクリプトでcatmanEasy/APを起動、プロジェクトおよびモジュール、EasyScript editorも参照してください。

Quick Accessツールバー

頻繁に利用する機能をすぐに使用できるよう、**Cockpit** → **Quick Access toolbar**からQuick Accessツールバーを開きます:

- スクリプトのプロジェクトを開く
- スクリプトの実行
- スクリプトの実行を停止
- プロシージャを選択して実行
- EasyScript editorを開く

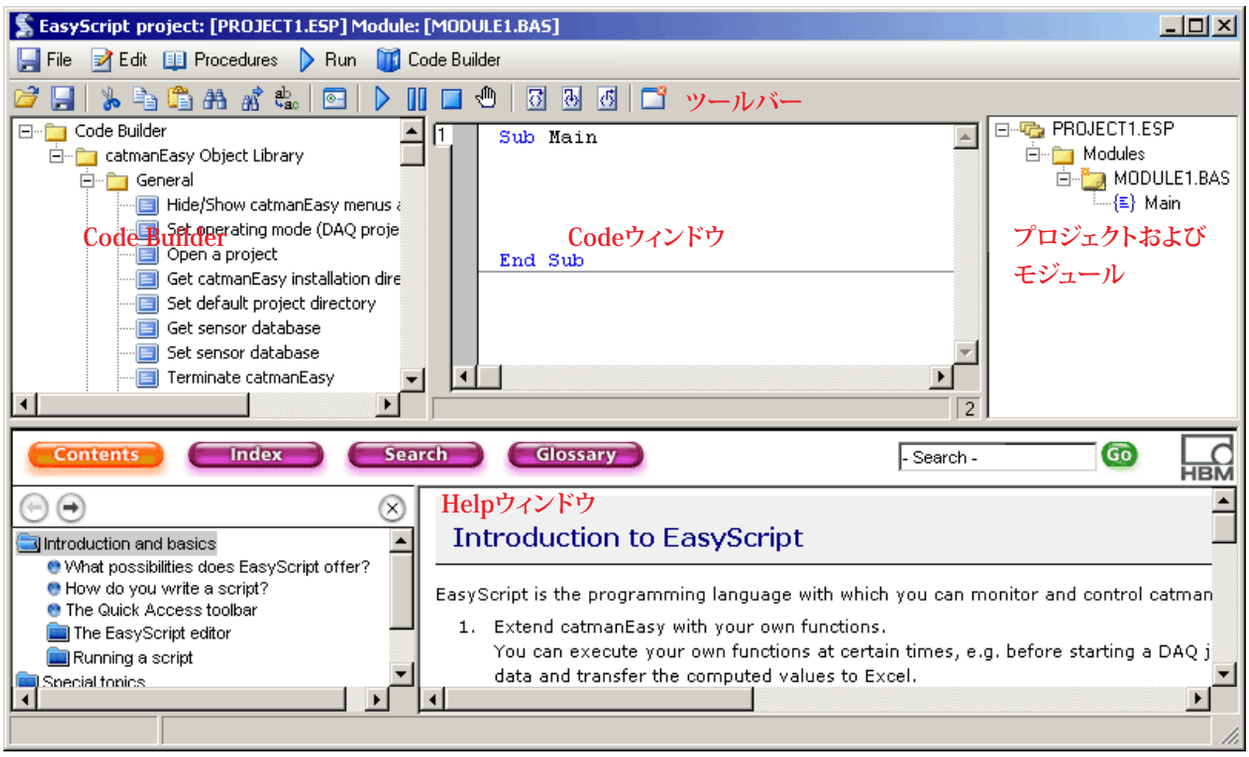
ツールバーのウィンドウキャプションには、現在読み込まれているEasyScriptのプロジェクト名が表示されます。



catmanEasy/APは、すぐに**Quick Access Toolbar**を表示することができます: **File** → **Options**, **Functions**タブ

EasyScript editor

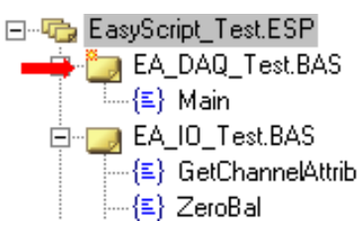
EasyScript editorは4つのセクションに分割されています。対応するトピックにアクセスするには、セクションをクリックしてください。



プロジェクトおよびモジュール

現在のEasyScriptプロジェクトはウィンドウの右側に表示されます。スクリプトプロジェクトは、少なくとも一つのコードモジュールから構成されています。モジュールのコードは、プロジェクトを開いたり、新規のプロジェクトを作成すると、EasyScriptエディタ中央のCodeウィンドウで見ることができます。

大規模なプロジェクトでは、全体を見渡しやすくするためや、コードを再利用しやすくするために、モジュールを複数に分割する必要があります(コンテキストメニューからNew code module)。このようなスクリプトを実行できるようにするため、モジュールの*一つに*プロシージャSub Mainを定義し、実行を開始します。このモジュールは*Start module*と呼ばれます(下図を参照)。また、プロジェクトのコンテキストメニューからStart moduleを定義することができます。



スクリプトプロジェクトのStart module

⚠ プロシージャSub Mainは、*Start module*でのみ発生し、プロジェクト毎に*一つだけ*あります。

別のモジュールからプロシージャを実行したい各モジュールは、'#usesとモジュール名を宣言する必要があります:

```
'#uses "SETUP.BAS"
```

モジュールヘッダに宣言がないと、呼び出されるプロシージャが認識されません。スクリプトを開始すると、呼び出しの時点で"Unknown identifier"(不明な識別子)というエラー メッセージが表示されます。

⚠ 命令文は、上記のように正確に入力する必要があります。すなわち、アポストロフィおよび#を含みます。

💡 catmanEasy/APは新規モジュールの追加時に、自動的にすべての既存モジュールに`#uses`命令を挿入することができます: **File → Options, EasyScript**タブ、**Code generation**フレーム: **Automatically insert #uses directive when creating a new module.**

☞ `#uses`命令で指定したモジュールが見つからない場合、catmanEasy/APはスクリプトを開始しません。

Projectウィンドウ内のモジュールをダブルクリックするか、Codeウィンドウの左上の番号をクリックしてモジュールを変更します。Codeウィンドウの番号はProjectウィンドウの順番ではなく、Codeウィンドウでの読み込み順に対応しています。

💡 モジュールおよびプロジェクトには意味のある名前を付けます; 古いファイルは新規プロジェクトの作成により、上書きされる可能性があるため、デフォルト名のMODULE1.BASおよびPROJECT1.ESPは使用しないでください。

Codeモジュール

Codeモジュールには、Subおよび関数形式のプロシージャ(サブプログラム)が含まれています。これらプロシージャは、同一または他のモジュールから実行することができます。

Codeモジュールは、常に拡張子が".BAS"となります。Projectウィンドウまたは、**File → New code module**から新規のCodeモジュールを作成します。新規のCodeモジュールはProjectウィンドウで作成し、Codeウィンドウで表示します。

ℹ VBAリファレンスのCodeモジュール参照してください。

Classモジュール

Classモジュールは、後でオブジェクトを作成するためのテンプレートです。Classモジュールは、プロシージャ(ここではメソッドやプロパティと呼ばれています)が含まれていますが、これらは他のモジュールから直接呼び出すことはできません。テンプレートに基づいて、クラスのオブジェクトはコードモジュール内で作成(インスタンス化)することができます。内部メソッドおよびプロパティは、このオブジェクトにより対処します。クラスのテンプレートから任意のオブジェクトを作成でき、各オブジェクトはクラス独自の内部データセットを受け取ります。

Classモジュールは、常に拡張子が".CLS"となります。Projectウィンドウのコンテキストメニューまたは、**File → New class module**から新規のClassモジュールを作成します。Projectウィンドウの新規モジュールは、Class_InitializeおよびClass_Terminateプロシージャで作成され、Codeウィンドウに表示されます。

クラス使用の例

```
Module PERSON.CLS
  Public First_name As String
  Public Family_name As String
  Public Sub SetName(ByVal VN As String, ByVal NN As String)
    First_name = VN
    Family_name = NN
  End Sub

Module TEST.BAS
```

```
'#uses "PERSON.CLS"
  Dim p1 As New Person
  Dim p2 As New Person
  p1.SetName "John", "Smith"
  p2.SetName "Ann", "Jones"
  ...
  MsgBox p2.First_name
```

⚠ クラスからのプロシージャはAuto-script execution(実行時点)に使用できず、任意のボタンに割り当てられません。

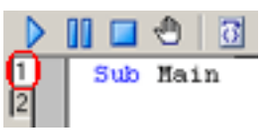
ℹ VBAリファレンスのClassモジュール参照してください。

Codeウィンドウ

Code Builderのエントリをダブルクリックすると、ウィンドウ内の対応するコードを生成します。ただし、通常のエディタと同様に命令を入力することができます。インテリセンス機能により、オブジェクトや命令が認識されたとき、可能なプロパティおよびメソッドの表示、キーワードのポップアップやツールチップをサポートしています。

オブジェクトや構文の説明を取得するには、用語をマークしてF1キーを押します。

読み込まれた各モジュールは、Code window左上にある番号に対応します。



現在、表示されているコードモジュールの番号が白色で強調表示されます。順序は読み込み順であり、Projectウィンドウ内の順序ではありません。エディタはCodeウィンドウで9個までのモジュールを管理でき、**Edit → Remove module from Code window**(または→ **Only display start module**により、現時点で必要ないモジュールをCodeウィンドウから削除することができます。モジュール自体は削除されませんが、再び編集できるようにするためには、Projectウィンドウでもう一度クリックする必要があります。

Codeウィンドウ右下の数字は、カードルが置かれている行を示しています。

Codeウィンドウの最下行はまた、EasyScriptのステータスメッセージを見ることができます。

スクリプトの記述後は、ツールバーやメニューから実行することができます。スクリプトの実行を参照してください。

EasyScript Code Builder

EasyScriptでプログラミングを簡単にするため、catmanEasy/APはEasyScriptエディタの左側にある*Code Builder*が利用できます。catmanEasy/APのオブジェクトモデルのメソッドおよびプロパティグループにまとめています。ここでは実際のスクリプトコードは表示されませんが、プレーンテキスト、例えば"Load test"が表示されます。目的の関数またはアクションを*ダブルクリック*し、Codeウィンドウの現在のカーソル位置に対応するコードを挿入します。

挿入されたコードは通常、メソッドの引数や戻り値の括弧内のプレースフォルダにポインタを持っています。例えば:

```
EA_Test.Load <Filename>
```

括弧を含む引数を変数名、数値、または文字列に置き換えます。

サンプル

```
Dim FileName As String
FileName = "C:\MYDATA\TEST1.TST"
EA_Test.Load FileName
```

Code BuilderおよびEasyScript Help

デフォルト設定でCode Builderは、EasyScript Helpと共に動作します: Code Builderで関数またはアクションをクリックすると関連する構文の説明がエディタ下部に表示されます(**File** → **Options**, **EasyScript**タブ:*Activate context help for Code Builder*)。

Code Builderの拡張

独自の関数でCode Builderを拡張することができます。Code Builderのコンテキストメニューまたは、EasyScriptエディタの**Code Builder**のメニュー内のエントリを使用してグループおよび関数の挿入、関数の編集(説明、コードの保存)と同様に切り取り、削除、コピーおよび貼り付けを行います。

注記

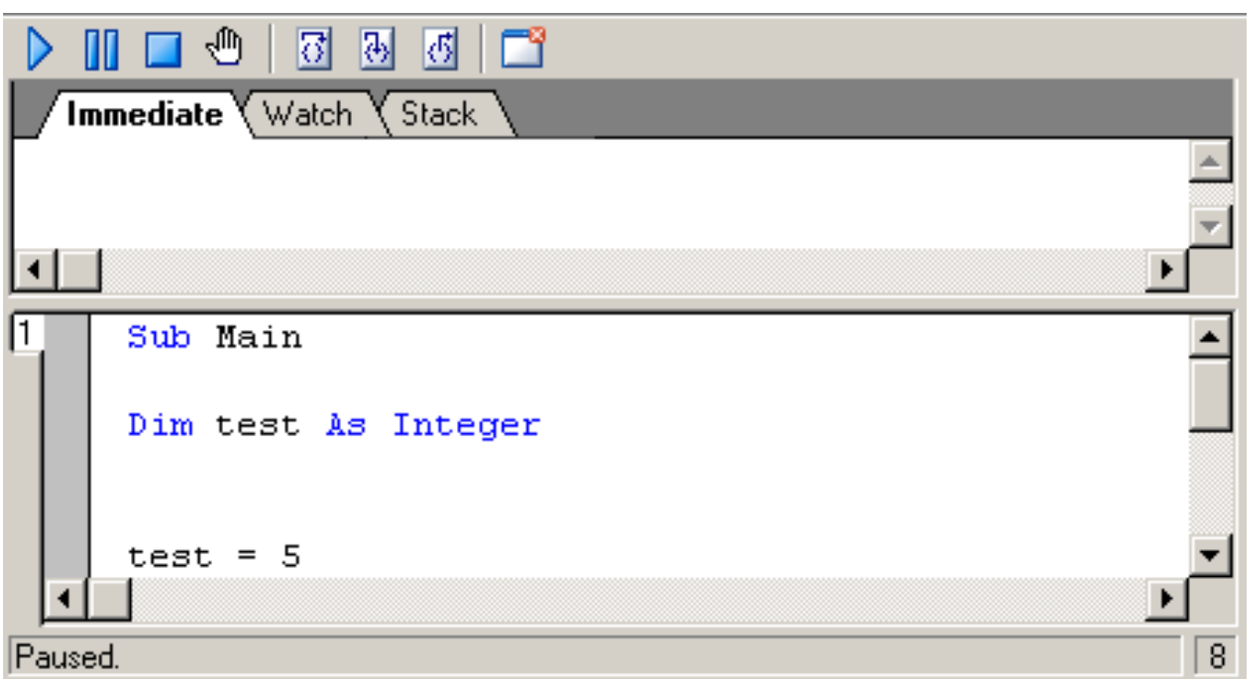
- Code Builderで挿入やエントリを削除する前に、VBSCRIPTBUILDER.XMLファイルのコピーを作成してください。
- 現在、複数グループ全体のコピー、切り取り、貼り付けは行えません。単一エントリのみ可能です。

EasyScript Helpウィンドウ

デフォルト設定でCode Builderは、EasyScript Helpと共に動作します: Code Builderで関数またはアクションをクリックすると関連する構文の説明がエディタ下部に表示されます(**File** → **Options**, **EasyScript**タブ:*Activate context help for Code Builder*)。

Codeウィンドウ内の命令をマークしてFiキーを押して構文の適切な説明を呼び出すか、- **Search** -ボックスのエントリを使用します: キーワードまたは検索フレーズを入力し、 Search をクリックします。用語を含むトピックは、左側に表示されます。再びコンテンツの一覧を表示するには、 をクリックします。

Debugウィンドウ



スクリプトの実行( )

スクリプトの実行を参照してください。

スクリプトの実行を中断( )

スクリプトの実行を手動(、 または)で継続、または終了( )させます。

スクリプトの実行を停止( )

現在のスクリプトを終了します。

ブレークポイントの設定/クリア( )

ブレークポイントを設定または削除します。スクリプトの実行は手動(、 または)で継続、または終了( )させるまで、ブレークポイントで保持されます。

飛び越える( )

シングルステップで、プロシージャを飛ばして次のプロシージャまで実行します。このボタンは、デバッグモードでのみ表示されます。

中に入る( )

シングルステップで、プロシージャに入って一時停止します。このボタンは、デバッグモードでのみ表示されます。

外に出る( )

シングルステップモードを終了し、次のプロシージャを実行します。このボタンは、デバッグモードでのみ表示されます。

モジュールの削除( )

Codeウィンドウから、表示されているモジュールを削除します。Start moduleは、Codeウィンドウから削除することはできません。Codeウィンドウでは、9個までのモジュールしか管理できないため、不要なモジュールはこのシンボルで削除します。モジュール自体はプロジェクトから削除されませんが、再び編集できるようにするためには、Projectウィンドウでもう一度クリックする必要があります。

イミディエイトウィンドウ(Immediate)

以下のアクションを実行することができます。

- コードの入力および実行(<ENTER>キーを押す)
- DirectウィンドウからCodeウィンドウにコードをコピー
- CodeウィンドウからDirectウィンドウにコードをコピーし、実行
- 変数の内容を表示: ?に続けて変数名(例えば`?Temperature`)を入力し、<ENTER>キーを押します。変数の内容は、次の行に表示されます。
- 変数の内容を変更: =演算子を使用して新しい内容を割り当てます。例えば`x = 25`。

ウォッチウィンドウ(Watch)

変数の内容を監視するには、このタブを使用します。単に変数名を入力し、<ENTER>キーを押します。変数の現在の内容は各プログラムステップ後に自動的に表示されます。

スタックウィンドウ(Stack)

呼び出しツリーを表示: 実行中のプロシージャにつながる呼び出しシーケンスです。呼び出しツリーのこのレベルでのコードの調査に、プロシージャをクリックします。

ステータスバー(Paused. 8)

ステータスメッセージを表示します。

EasyScriptツールバー

プロジェクトを開く( )
スクリプトプロジェクトを読み込むためのダイアログボックスを開きます。

プロジェクトの保存( )
現在のスクリプトプロジェクトを保存します。プロジェクトまたはモジュールの一つが、デフォルト名のPROJECT1.ESPやMODULEx.BASの場合、モジュール(*.BAS)およびスクリプトプロジェクト(*.ESP)を保存するためのダイアログを開きます。

切り取り( )
*Code*ウィンドウでマークしたテキストを、クリップボードに移動します。

コピー( )
*Code*ウィンドウでマークしたテキストをコピーします。

貼り付け( )
*Code*ウィンドウ内でコピーまたは切り取ったテキストを貼り付けます。

検索...( )
*Code*ウィンドウ内のテキストを検索するためのダイアログボックスを開きます。

次を検索( )
開始している検索を続行します。最初に を使用して*Code*ウィンドウ内のテキスト検索を定義します。

置き換え( )
*Code*ウィンドウ内のテキストを検索および置き換えするためのダイアログボックスを開きます。

ユーザダイアログエディタ( )
UserDialogエディタウィンドウを開きます。

スクリプトの実行( )
スクリプトの実行を参照してください。

スクリプトの実行を中断( )
スクリプトの実行を手動(、 または)で継続、または終了( )させます。

スクリプトの実行を停止( )
現在のスクリプトを終了します。

ブレークポイントの設定/クリア( )
ブレークポイントを設定または削除します。スクリプトの実行は手動(、 または)で継続、または終了( )させるまで、ブレークポイントで保持されます。

飛び越える( )
シングルステップで、プロシージャを飛ばして次のプロシージャまで実行します。このボタンは、デバッグモードでのみ表示されます。

中に入る( )
シングルステップで、プロシージャに入って一時停止します。このボタンは、デバッグモードでのみ表示されます。

外に出る( )
シングルステップモードを終了し、次のプロシージャを実行します。このボタンは、デバッグモードでのみ表示されます。

モジュールの削除( )
Codeウィンドウから、表示されているモジュールを削除します。Start moduleは、Codeウィンドウから削除することはできません。Codeウィンドウでは、9個までのモジュールしか管理できないため、不要なモジュールはこのシンボルで削除します。モジュール自体はプロジェクトから削除されませんが、再び編集できるようにするためには、Projectウィンドウでもう一度クリックする必要があります。

UserDialogエディタ

Codeウィンドウのカーソルを置いた位置で、**Edit** → **UserDialog editor**または よりUserDialogエディタを呼び出して生成します。

アイテムのプロパティの編集( )
選択したアイテムのプロパティを編集します。
ホットキー: Enter

削除( )
選択したアイテムを削除します。
ホットキー: Del

切り取り( )
選択したアイテムをクリップボードに移動します。
ホットキー: Ctrl + X

コピー( )
選択したアイテムをクリップボードにコピーします。
ホットキー: Ctrl + C

貼り付け( )
選択したアイテムの前面にクリップボードのアイテムを貼り付けます。
ホットキー: Ctrl + V

前面へ移動( )
選択したアイテムを前面に移動します(コードのEnd Dialogに対して)。
ホットキー: Ctrl + Up

背面へ移動( )
選択したアイテムを背面に移動します(コードのEnd Dialogに対して)。
ホットキー: Ctrl + Down

前面のアイテムの選択( )

選択したアイテムの前のアイテムを選択します(コードのEnd Dialogに対して)。

背面のアイテムの選択( )

選択したアイテムの後のアイテムを選択します(コードのEnd Dialogに対して)。

グリッドの設定( )

グリッド設定を変更します。

ホットキー: Ctrl + G

保存して終了( )

UserDialogを保存して終了します。

アイテムの選択(*)*
　アイテムの選択、移動または編集モードです。

グループボックスの追加(*)*
　GroupBoxアイテムを追加します。

テキストの追加(*)*
　Textアイテムを追加します。

テキストボックスの追加(*)*
　TextBoxアイテムを追加します。

チェックボックスの追加(*)*
　CheckBoxアイテムを追加します。

オプションボタンの追加(*)*
　OptionButtonアイテムを追加します。

コンボボックスの追加(*)*
　ComboBoxアイテムを追加します。

リストボックスの追加(*)*
　ListBoxアイテムを追加します。

ドロップダウンリストボックスの追加(*)*
　DropListBoxアイテムを追加します。

マルチリストボックスの追加(*)*
　MultiListBoxアイテムを追加します。

OKボタンの追加(*)*
　OKButtonアイテムを追加します。
　　ダイアログ毎にOKボタンを一つだけ使用できます。

キャンセルボタンの追加( )

CancelButtonアイテムを追加します。

ダイアログ毎にキャンセルボタンを一つだけ使用できます。

プッシュボタンの追加( )

PushButtonアイテムを追加します。

ピクチャの追加( )

画像アイテムを追加します。

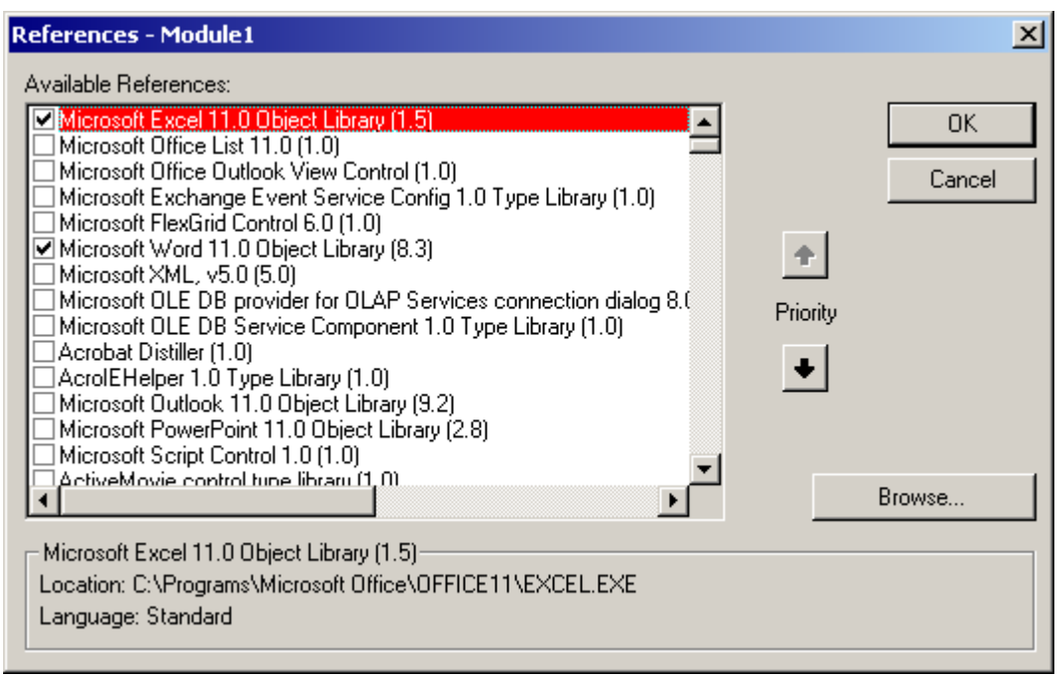
ユーザダイアログは、`Begin Dialog … End Dialog`をコードで識別します。コードセクション内にカーソルを置き、 をクリックしてグラフィカルなダイアログの編集を行います。

Object Referencesダイアログ

このダイアログを使用して"事前バインディング"ソフトウェアオブジェクトを指定します。スクリプトで使用する予定のすべてのActiveX/COMに対応するプログラムまたはコンポーネント(.EXE、.COM、.OCX)をチェック( )するだけです。OKで確定後、有効化したオブジェクトのリファレンス(GUID)が、スクリプトコードに挿入されます。これらリファレンスはコードエディタに表示されませんが、コードの入力時にオブジェクトが利用可能になります。

これらを適切に使用するために、一般的にソフトウェアオブジェクトへのアクセス権を持っている必要があります。

重要なテーマ: オブジェクトなど他のプログラムの作成と使用も参照してください。



スクリプトの実行

catmanEasy/APでは、スクリプトを実行する三つの方法を提供しています。

1. EasyScriptプロジェクトの実行(完全)
2. 単一プロシージャの実行
3. スクリプトでcatmanEasy/APを起動

Debugウィンドウ、EasyScriptツールバー、Quick Accessツールバーも参照してください。

EasyScriptプロジェクトの実行

完全なEasyScript projectの実行には、*Main*プロシージャが必要です。

```
Sub Main
...
End Sub
```

*最初の*プロジェクトモジュール。新規プロジェクトの作成時にプロシージャは、エディタより自動的に作成されます。プログラムの実行は常に、このプロシージャのコードで始まります。*Main*プロシージャにコードが含まれていなくてもスクリプトエンジンは開始し、単一プロシージャの呼び出しを待ちます。単一プロシージャの実行を参照してください。

EasyScriptプロジェクトの開始

EasyScriptのRunメニューから

エディタでスクリプトプロジェクトを開始するには、**Run** → **Run script**または、▶を使用します。
⏸で一時停止させたり、ブレークポイントに達すると、Debugウィンドウを表示します。

catmanEasy/APのEasyScriptメニューから

1. **EasyScript** → **Load script project**で、スクリプトを読み込み
2. **EasyScript** → **Run script project**で、読み込んだプロジェクトを開始

Quick Accessツールバーから

1. ツールバーが表示されていない場合、**Cockpit** → **Quick Access toolbar**から開きます。
2. ▶でスクリプトプロジェクトを実行します。

catmanEasy/APプロジェクト(拡張子.MEPまたは.OFP)にEasyScriptプロジェクトが含まれている場合、プロジェクト読み込み時にスクリプトを自動的することができます(**File** → **Options**から**EasyScript**タブ)。自動実行オプションを無効にしている場合、catmanEasy/APはスクリプトの実行を求めるメッセージを表示します。

実行中のEasyScriptは、ステートメントごとに実行中のDAQジョブに制御を渡します。これにより、catmanEasy/APとデバイス間でのデータ転送がスクリプトの実行でブロックされなくなります。しかし、データ転送がブロックされる例外はいくつかあります:

- スクリプトでメッセージボックスを表示する場合(MsgBoxステートメント)。
- スクリプトでモーダルなUser Dialogを表示する場合

単一プロシージャの実行

完全なスクリプトの実行に代わり、EasyScriptプロジェクトから個々のプロシージャを実行することができます。その場合、スクリプトプロジェクトはプロシージャのコンテナの役割を果たします。

☞ クラスモジュールのプロシージャ(メソッド)は、*関数*(プロシージャの戻り値)や、プロシージャの宣言を*Private*で実行できません。
プロジェクトおよびモジュールも参照してください。

☞ 単一プロシージャのみの実行を予定している場合であっても、スクリプトを開始する必要があります(EasyScriptプロジェクトの実行を参照してください)。しかしこの場合、*Main*プロシージャは必須ではありません - 省略するか空き(コードを書かない)にすることができます。*Main*プロシージャにコードが含まれている場合、そのコードが実行され、スクリプトを終了します。

☞ プロシージャの実行は、いつでも中断(例えばボタンをクリックして)することができます。このイベント自身で、別のプロシージャを呼び出すことができます。このようなプロシージャは入れ子にすることができるため、いわゆる"Reentrancy problems"(再入可能問題)につながる可能性があります。詳細については入れ子のプロシージャ呼び出しを参照してください。

単一のプロシージャの実行するには、以下の三つの方法があります: 自動(catman ProfessionalのAuto Command Listのような)、Quick Accessツールバー、またはボタンをクリック。

特定の時間に自動実行

現在のEasyScriptプロジェクトから単一プロシージャに特定の実行時点を割り当てます。catmanEasy/APは、この時点に達すると自動的にプロシージャ(複数可)を実行します。

実行時点

- DAQジョブの開始前
- DAQジョブの終了後
- DAQジョブのすべての繰り返し後
- すべてのDAQジョブ後
- チャネルの初期化前
- チャネルの初期化後
- 計測の開始前
- 計測データブロックの各転送後(読み取りブロック、**After transfer DAQ data block**)
- 計測の停止後(**DAQ stop**)
- テストの読み込み後(Analysisモードのみ)
- 他のパネルへ変更後

手順

1. スクリプトエディタでプロシージャを作成し、エディタを閉じます。別の方法としてcatmanEasy/APで、**EasyScript → Load script project**からプロシージャを読み込むことができます。
2. **EasyScript → Auto execution**より、定義のためのダイアログボックスを表示します。
3. 使用するプロシージャをリストから選択します。
4. ダイアログボックスで、希望する実行時点をマークします。
5. 実行時点をプロシージャに挿入するため、***Assign***をクリックします。

💡 4.および5.に代わり、希望する実行時点までドラッグすることができます。

6. 他のプロシージャに必要な場合、上記実行時点の手順を繰り返します。

Quick Accessツールバーより実行

ドロップダウンリストから目的のプロシージャを選択し、をクリックします。スクリプトが開始されている必要があり、そうでないと、このボタンは有効になりません。

パネル上のボタンをクリックして実行

プロシージャは、ボタンからインタラクティブに実行できます。自動実行とは対照的に、任意のEasyScriptプロジェクトを割り当てることができます。必要に応じて、プロジェクトは実行前に読み込まれます。

手順

1. ツールバー(右側のサイドバー)よりボタン( )を生成します。
2. ボタンのコンテキストメニューから**Configure: Button**を選択します。
3. 元となるプロシージャ( )または、現在のプロジェクト(デフォルト設定)を使用してEasyScriptプロジェクトを選択します。
4. 完全なスクリプトを実行したい場合、リストから利用可能なプロシージャを選択するか、*[Main]*プロシージャを選択します。
5. スクリプトを開始します(メニューの**EasyScript → Run script project**)。

注記

- 個々のプロシージャの実行前に、スクリプトプロジェクトを開始する必要があります(catmanEasy/APメインウィンドウの**EasyScript → Run script project**)。しかし、このモードでは新規ボタンの作成やインタラクティブオブジェクトの編集も行えません。
- 新規作成したボタンでは、*Procedure to execute*ボックスに*[Main]*がデフォルトとして設定されています。そのため、完全なスクリプトプロジェクトを実行します。
- catmanEasy/APプロジェクト(ファイル拡張子.MEPまたは.OFP)にEasyScriptプロジェクトが」含まれている場合、プロジェクト読み込み時にスクリプトを自動的に実行され(**File → Options**から**EasyScript**タブ)、ボタンや自動プロシージャ実行が機能するようになります。自動実行オプションを無効にしている場合、catmanEasy/APはスクリプトの実行を求めるメッセージを表示します。
- スクリプトプロジェクトに*Main*プロシージャが含まれいない場合、または、Mainプロシージャが空きの場合、スクリプトは自動的に終了しません。どのような場合でも手動またはEA.Terminateで、終了させる必要があります。

Automatic Procedure Execution(自動プロシージャ実行)

*Automatic procedure execution*はcatmanEasy/APのメカニズムであり、特定の時点(実行ステージ)で現在のスクリプトプロジェクトまたは、直接スクリプトコードからプロシージャを実行することが可能です。例えば、DAQジョブの開始後や、チャネルの初期化後に発生します。Configure automatic procedure executionは、EasyScriptエディタのメニューから**EasyScript → Auto execution**または、AutoSequenceエディタのキーボードショートカットCTRL + Aを使用します。

💡 いくつかの実行ステージでは、*EA_AutoExecParam*プロパティを介して取得できる追加データが提供されます。

Configure automatic procedure executionマネージャダイアログは現在、以下のエントリポイントを選択することができます。

Before DAQ job	DAQジョブの開始前に呼び出されます。チャネルの初期化といった準備は、この段階では行われません。
After DAQ job	DAQジョブが終了し、すべてのDAQ後の操作(データをファイルに保存する)が完了した直後に呼び出されます。
After all DAQ job repetitions	ジョブの繰り返し後、最後のジョブの繰り返し後に発生します。
After all DAQ jobs	一連のジョブ終了後、最後のジョブ終了後に発生します。
Before channel initialization	チャネルの初期化前に発生します。
After channel initialization	チャネルの初期化後に発生します。 直接コマンドインタフェースで特別チャネルの設定を行う場合、適切な場所です(EA_IO.Commandを参照してください)。
Before DAQ start	すべての準備(チャネルの初期化など)が完了した後、データ収集開始の直前に発生します。"Before DAQ job"の後に発生します。
After start trigger	開始トリガのイベント発生後に発生します。
After transfer DAQ data block	デバイスから新規のデータブロックを転送した直後に発生します。データは既にスケーリングされていますが、オンラインの演算チャネルはまだ計算していません。これはオンライン計算の入力前に、データを操作する機会を与えています。可視化の更新前に発生します。
After offline computation	デバイスから新規のデータブロックを転送し、すべてのオンラインの演算チャネルで処理された後、データが一時ストアに保存される前に発生します。可視化の更新前に発生します。
After storing DAQ data block	デバイスから新規のデータブロックを転送し、すべてのオンライン計算チャネルを処理し、データを一時ストアに保存された後に発生します。可視化の更新前に発生します。
After complete data transfer cycle	デバイスから新規のデータブロックを転送し、すべてのオンライン計算チャネルを処理し、データを一時ストアに保存され、可視化の更新後に発生します。 注意:*EA_Job.Terminate*で実行中のDAQジョブを終了させるために使用できる、唯一の実行ステージです。
During DAQ job	DAQジョブの実行中に発生します。
Before saving data	計測データが一時ストアからテストファイルに書き込まれる直前に発生します。
After saving data	計測データが一時ストアからテストファイルに書き込まれる直後に発生します。
After DAQ stop	データ収集の停止直後に発生しますが、すべてのDAQ後の操作(データをファイルに保存)が実行される前です。
After online test load	解析モード。テストの読み込み後に発生します。
After panel change	ユーザによる、視覚化パネルの切り替え時に発生します。
Before loading a project	プロジェクトの読み込み前に発生します。
After loading a project	プロジェクトの読み込み後に発生します。
Before saving a project	プロジェクトの保存前に発生します。
After saving a project	プロジェクトの保存後に発生します。
Before closing the file on project load	catmanのプロジェクトデータの読み込み後、プロジェクトの読み込み中にプロジェクトファイルを閉じる前に発生します。
Before closing the file on project save	catmanのプロジェクトデータの書き込み後、プロジェクトの保存中にプロジェクトファイルを閉じる前に発生します。
After start of catman	catmanのメインウィンドウ表示後に発生します。
Before terminating catman	catman終了前に発生しますが、メインウィンドウのアンロード前に恒久的なユーザ設定がレジストリに書き戻される前に発生します。
After synchronisation of cursors	この実行ステージは、グラフカーソルが移動された場合、またはビデオ再生で新しいフレームが表示された場合に呼び出されます。カーソルがマップまたは角度同期表示に配置されている場合にも呼び出されます。現在位置のタイムスタンプ(秒単位)は、*EA.AutoExecParam*プロパティを介して取得します。

スクリプトの実行を終了

catmanEasy/APは、Mainプロシージャのコードがプロセスを完了したとしても、自動的に実行中のスクリプトを終了しません。スクリプトは、ボタンや自動実行ステージの一つが発行するプロシージャ(可能な)の呼び出しを待ちます。

スクリプトは対話的(Quick-Accessツールバーのメニューから)、またはスクリプトコード自身のコマンド(EA.Terminate)で終了することができます。

スクリプトでcatmanEasy/APを起動

catmanEasy/APの起動時にEasyScriptを実行するよう、設定することができます。これを実現するには、次のコマンドラインを入力します:

```
/script: Project file
```

例えば以下のように入力します:

```
c:\programs\HBM\catmanEasy/AP\catmanEasy/AP.exe /script: c:\programs\hbm\
catmanEasy/AP\projects\myscript.esp
```

シーケンス全体を自身で制御したい場合、この方法を使用します。指定したスクリプトが実行される前に、次のアクションのみが実行します。

1. catmanのCore Servicesとの接続
2. 計測データの一時ストレージの作成
3. メニューシステムの初期化

デバイスのスキャンは実行しません。スクリプトにより、同様のアクションを実行する必要があります。例えばEA.OpenProjectで、DAQを読み込みます。

注記

- このモードでは多くの場合、ユーザインタフェースを自身で処理します。そして、EA.ConfigureGUIでcatmanEasy/APのメニューおよびツールバーをマスクします。
- スクリプトの最後には、catmanEasy/APを終了するEA.Terminateを使用します。
- 指定したファイルにパスが含まれていない場合、catmanEasy/APはスクリプトプロジェクトが最後に使用したフォルダ内のファイルを検索します。デフォルト設定は、C:\Users\<ユーザ名>\ドキュメント\HBM\catmanEasy\PROJECTSです。

QuickScript

*QuickScript*ウィンドウを使用すると、その場でコードを入力して実行することができます。成熟したのEasyScript開発環境とは対照的に、対話型のコマンドウィンドウのように動作します。それでもコードの編集、SUBや関数の作成、ファイルからコードを読みむことはできます。

メインメニューから"Advanced/QuickScript"を選択して*QuickScript*ツールを起動します。*QuickScript*ウィンドウは、常に画面の上部に常駐するため、簡単にアクセスすることができます。

QuickScriptツールは、以下が行えます:

- コードの編集
- SUBおよび関数の作成と呼び出し
- ファイルからコードを読み込み/書き込み
- MAINプロシージャの実行

ただし、以下の機能はQuickScriptに提供されていません。

- 追加のコードモジュールを作成
- クラスモジュールの作成

重要なテーマ

このセクションでは、以下のトピックについて説明します。

直接スクリプトコードの使用方法
ユーザとの相互作用
catmanのGUIを非表示にする方法
EasyScriptアプリケーションのユーザインタフェースを作成
チャネルの初期化およびトレーサビリティデータ
オブジェクトなど他のプログラムの作成と使用
Microsoft Officeの使用
FileSystemオブジェクトを使用
DLL関数の呼び出し
入れ子のプロシージャ呼び出し
AddIn
ActiveX ServerとしてcatmanEasy/APを使用
カスタムデータのグラフ表示
ローレベルのグラフプログラミング
ローレベルのテーブル(スプレッドシート)プログラミング
パネルに独自の可視化コントロール(OCX)を追加
SystemLogオブジェクトの使用
QuantumXモジュールと直接通信

直接スクリプトコードの使用方法

概要

EasyScriptプロジェクトの外部で実行されるEasyScriptコードのセクションは、"direct script code"と呼ばれます。以下の場所で直接スクリプトコードを使用することができます:

- 可視化パネル上のボタンオブジェクト、カスタムのリボンコントロール、コンテキストメニューに接続。
  コントロールをクリックすると、コードを実行します。
- イベントの"Script"アクションに接続。
  イベント発生時にコードを実行します。
- AutoSequenceの"Script"アクション。
- Automatic Procedure Executionの一つに直接コードを接続(version 3.2以降)。
  実行ステージが処理されると、コードを実行します。
- 補助チャネルに直接コードを接続(version 3.2以降)。
  コードは各転送ブロック後、自動的に各補助チャネルで実行します。
- QuickScriptエディタを起動(Mainメニュー/Advanced functions)し、対話的に*直接コード*を実行します。

*直接スクリプト*が"real"スクリプトコードのオーバーヘッドを回避し、取り扱いがいくらか容易です。成熟したEasyScriptエディタを起動(EasyScriptリボンタブ)する必要はなく、代わりにシンプルなQuickScriptエディタ(ボタンに直接コードを接続したり、catmanでエディタの起動を制御)を使用します。*直接スクリプト*はDAQまたは解析プロジェクトに保存されます - 個別のEasyScriptプロジェクトは必要ありません!

直接スクリプトコードは、エディタよりデフォルトで提示されている`Main()`関数に限定されるものではありません。コードの要求に応じて、いくつでも`Subs()`や`Functions()`を追加することができます。実行はしかし、常に`Main()`関数から開始します。最大限のスクリプトプロジェクトとは対照的に、複数のコードモジュールを持つことができず、クラスモジュールもサポートしていません。

注記: 直接スクリプトコードの一部が実行されると、スクリプトエンジンは再初期化され、コードは実行するエンジンにより一時ファイルにコピーされます。このプロセスにはいくつかのオーバーヘッドを伴い、数ミリ秒かかる場合があります。時間的余裕がない場合、EasyScript内のプロシージャのコレクションとしてコードえお維持する方がよいかもしれません。

直接コードブロック間で共通データを共有

catmanのバージョン5.3では、いわゆる"グローバルコード"を導入しています。これにより、共通データの共有がさらに簡単になり、プロセス変数とは対照的に、共通関数の共有も可能になります。詳細については、EA.SetGlobalCodeを参照してください。

直接スクリプトコードの一部が実行されると、スクリプトエンジンは再初期化され、コードは実行するエンジンにより一時ファイルにコピーされます。これは個々の直接スクリプトのスニペットはEasyScript共通の"屋根"の下で実行しないと、特に後続の呼び出し間でデータ(変数の内容)を保持することができないことを意味します(各直接スクリプトコードが複数のボタンと接続するイメージです)。すなわちコードスニペットは、データを共有する方法はありません。次の例では、この問題を示しています:

```
Dim Counter As Long
Sub Main()

  Counter = Counter +1
  DoSomething
  If Counter > 10 Then ......
  'Will never happen
  EndIf

End Sub

Sub DoSomething()
  MsgBox Counter
End Sub
```

コードの呼び出しすべてでゼロに再初期化されるため、*Counter*は動作しません。もちろん、Counterはコードプロシージャ全体で使用することができ、DoSomethingは正しく"1"を表示します。

この問題を解決するには、catmanの*プロセス変数*を利用する必要があります。プロセス変数はAutoSequencesで使用し、EasyScriptとAutoSequences間でデータ交換を行います(AutoSequenceにEasyScriptコードアクションが含まれている場合)。ただし、個々の直接コードブロック間でデータを交換するためにプロセス変数を使用できます(例えば、直接スクリプトコードに接続したボタンを使用している場合)。このようにプロセス変数は、プロジェクト全体の"グローバル"変数のように機能します。

コマンドEA.CreateProcessVariable、EA.SetProcessVariableおよびEA.GetProcessVariableを参照してください。

ファイルにデータを書き込む場合、特定の問題が発生する可能性があります。スクリプトエンジンの再初期化はファイルハンドルを無効にするため、ある段階でファイルを開き、別の直接スクリプトコードでそのファイル継続的に書き込むことはできません。たとえプロセス変数にファイルハンドルを保存した場合でも動作しません。したがって、ファイルにデータを書き込みようなスクリプトコードが呼び出される度に、ファイルを再度開いて閉じる必要があります。ファイルの開閉に必要なオーバーヘッド時間の余裕がない場合、EasyScript内のプロシージャのコレクションとしてコードを保持する方がよいかもしれません。

補助チャネルで直接スクリプトコードを使用

補助チャネルは、DAQモードで作成する仮想DAQチャネルです("Special"メニューの*channel configuration*タブ)。catmanを使用すると、これらのチャネルに独自のデータを提供でき、他のDAQチャネルや演算チャネルと同様に視覚化され、保存されます。バージョン3.3.2以前ではスクリプトプロジェクトを作成し、自動実行ステージ"After transfer DAQ data block"にスクリプトプロシージャをリンクしなければなりませんでした。プロシージャはその後、*EA_DAQ.CurrentBlockSize*、*EA_DAQ.SetCurrentBlock*および*EA_DAQ.GetCurrentBlock*メソッドを利用してカスタムサンプルを埋めます。

Version 3.3.2以降からEasyScriptを必要とせずに、補助チャネルに直接スクリプトコードを接続することができます。これに伴い"reference"チャネルを割り当てることができます - catmanは、どのチャネルから計算するサンプル数を決定する必要があるかを自動的に認識します。*EA_DAQ.CurrentBlockSize*、*EA_DAQ.SetCurrentBlock*および*EA_DAQ.GetCurrentBlock*メソッドは、基準チャネルを使用するため、channel parameter = 0を受け入れます。

補助チャネルに直接スクリプトコードを割り当てる場合、catmanは自動的に一般的な動作原理を示すスクリプト本体を提供します。

実際、補助チャネルは、チャネルにサンプル データを入力するだけでなく、あらゆる目的に使用できます。実際、内部でコード化したアクションはすべて、各データ転送ブロックの後に周期的に実行されます(例えばデータを監視し、必要に応じてデジタル出力を設定)。

ユーザとの相互作用

スクリプトプロジェクトでは通常、ユーザから情報を取得(例えばパスワードや選択内容)するか、ボタンをクリックしたときに反応する必要があります。EasyScriptには、これを実行するためのいくつかの方法があります:

1. 独自のタブ、グループおよびコントロールを使用してリボンバーを拡張
2. パネル上にボタン、テキストボックス、チェックボックスなどを直接作成
3. コンテキストメニューポップアップメニュー)の作成(
4. "ユーザ ダイアログ"を作成 - これはテキストボックス、チェックボックス、ラベル、画像などを含むポップアップウィンドウで、catmanEasy/APは、このためのUser dialog editorを提供します。

リボンバーを拡張

この方法は、"EasyScriptアプリケーションのユーザインタフェースを作成"に詳細に記載されています。これはユーザインタフェースを作成する最も強力で柔軟な方法です。

パネルオブジェクトの使用

EasyScriptのオプションが利用可能な場合、Visualizationタブのオブジェクトツールバーに複数の追加オブジェクトを提供します: コマンドボタン、テキストボックス、チェックボックス(ラジオ ボタンとしても機能)、ドロップダウンリストボックスおよびテーブル(スプレッドシート)。パネルにオブジェクトを配置し、EA_Panelオブジェクトの適切なメソッドで内容を読み書きするだけです(SetValue、GetValue)。

これらのオブジェクトはすべて、スクリプト内のいわゆる"イベントハンドラプロシージャ"を起動することができます。これはユーザがオブジェクトをクリックするか、ダブルクリックすると発生します。ただし、*MouseDown*、*MouseUp*やスライダ/ノブの値変更など、さらに多くのイベントタイプがあります。イベントハンドラプロシージャは例えば、"Button1_OnClick"のように、オブジェクト名とイベントのタイプで構成される標準名が付けられます。この名前またはプロシージャのシグネチャ(プロシージャ名以下のパラメータ)は変更しないでください。

イベントハンドラプロシージャの作成や編集は、オブジェクトのコンテキストメニューを使用します。catmanEasy/APは、プロシージャが存在しない場合は作成するか、既存のプロシージャを表示します(もちろん手動でイベントハンドラ関数を作成することも問題ありません)。

注記: バージョン3.0以前では、ボタンオブジェクトにのみプロシージャの割り当てが行えました。これは、スクリプトプロジェクトで利用可能な任意のプロシージャにすることができます。これは3.0以降でもサポートしていますが、_OnClickイベントプロシージャのみを使用することを推奨しています!

コンテキストメニュー(ポップアップメニュー)

非常に単純なポップアップメニューは、VBAのShowPopupMenuメソッドで実現することができます。以下のセクションでは、ポップアップメニューの視覚的スタイルやアイコンなどを使用してポップアップ メニューを作成および表示する方法について説明します。

メニューの作成および表示

EA_Ribbon.ShowTemporaryPopupMenuからポップアップメニューを作成する方法が最も簡単です。この関数は、ユーザが行ったメニューのクリックを処理や、メニュー項目のパブリックプロシージャの名前を処理します。このハンドラプロシージャは、"MenuHandler_"で始まる必要があり(例えば、"MenuHandler_SelectSampleRate")、以下のシグネチャを持っている必要があります:

```
Public Sub MenuHandler_xxxxx(ByVal Index As Integer, ByVal Caption As Integer)
```

catmanがハンドラを呼び出すと、キャプションとともに選択した項目のインデックス(最初のアイテムは1)を渡します。

ポップアップメニューを作成する二つめの方法は、リボンバーシステムを拡張することです。最初に対話式メニューを備えたGUIデザイナで、コントロールおよびサブコントロールを作成します。そして、EA_Ribbon.ShowPopupMenuからメニューを呼び出します。この場合、ユーザ選択はデザイナでサブコントロールに割り当てたプロシージャにより処理されます。この方法の詳細は、"EasyScriptアプリケーションのユーザインタフェースを作成"で詳しく説明しています。

メニューの表示場所
通常、パネル上の何も無い場所や、パネル上のオブジェクトをマウスで右クリックするとポップアップメニューが表示されます。これに使用できるイベントハンドラプロシージャは、*xxx_OnClick*、*xxx_OnMouseDown*および*xxx_OnMouseUp*です。

catman組み込みのコンテキストメニューと干渉
作成したポップアップメニューが、catman組み込みのコンテキストメニューの一つと競合する可能性があることに注意してください。例えば、スクリプト実行中のパネル領域でのMouseUpイベントは、コンテキストメニューの印刷やエクスポートを呼び出します。このイベント(またはMouseDownやClick)に独自のポップアップを表示させることを想像してください: その場合、両方のメニューが相次いで現れます。これを防ぐには、`EA_Ribbon.SetContextMenuMode`ですべてのオブジェクトクラスのビルトインコンテキストメニューを無効にします。

ビルトインコンテキストメニューの拡張
ビルトインコンテキストメニューを置き換える代わりに、独自のメニュー項目を追加することができます。これを行うには、*EA_Ribbon.ExtendContextMenu*を使用します。このメソッドは*EA_Ribbon.ShowTemporaryPopupMenu*で説明しているように、ハンドラプロシージャを指定する必要があります。

ユーザダイアログ

ユーザダイアログは、オンデマンド(スクリプトコードにより呼び出される)にポップアップ表示する別ウィンドウです。ダイアログウィンドウは、テキストボックス、チェックボックス、ラジオボタン、ラベル、画像など複数の異なるコントロールタイプが含まれる場合があります。このようなダイアログは、純粋にコードによって最初からを作成することもできますし、EasyScript開発環境の*User Dialog Editor* を使用することもできます。User Dialog Editorを使用して、視覚的にダイアログを作成することができます(ドラッグによるオブジェクトの位置決め、コントロールをダブルクリックしてプロパティを設定)。閉じると、対応するコードが自動的に生成されています。

ダイアログウィンドウが表示された後、ユーザはテキストボックスに書き込むなどのオプションを選択することができます。その後、ユーザはOK、CANCELまたは指定した任意のボタンを押します。ダイアログウィンドウを閉じた後、**dlg**オブジェクトのプロパティによってどのボタンが押されたかがを通知し、ダイアログのコントロールの内容にアクセスすることができます。

複数のダイアログをコード内に作成することができます。しかし、一度に表示できるダイアログは、一つだけです。

⚠ ダイアログエディタで既存のダイアログを編集するには、ダイアログエディタを開く前に、**Begin Dialog**および**End Dialog**間の行にカーソルを配置しておくことが必要です。

例

```
Sub StartTest()
  Dim Operator As String
  Begin Dialog UserDialog 400,203, "Log In"
    TextBox 20,28,330,21,.txtOperator
    Text 20,7,120,14,"Operator",.Text1
    Text 20,63,120,14,"Password",.Text2
    TextBox 20,84,330,21,.txtPassword,-1
    CheckBox 20,126,270,14,"AutoCAL ON",.chkAutoCal
    OKButton 150,175,100,21
    CancelButton 280,175,90,21
  End Dialog
  Dim dlg As UserDialog          'Must always occur AFTER the dialog creation code!
  dlg.txtOperator = "Smith       'Preset fields
  Dialog dlg                     'Show the dialog
  Operator = dlg.txtOperator     'Query user input
End Sub
```

名前(.txtOperatorのように先頭にドットが付いている)は非常に重要です。これらの名前から、ダイアログの個々の要素の内容や状態にアクセスすることができます。次の例では、チェックボックスの値を"checked"に設定します:

```
  dlg.chkAutoScale = 1
```

注記

- ダイアログ変数の宣言(`Dim dlg As UserDialog`)は、常にダイアログ定義コードの後に指定する必要があります。
- **Cancel**ボタンをクリックすると、常にランタイムエラー10031が発生します。次の例では、エラーメッセージを非表示にするために、On Errorを使用します。

```
Sub MyDialog
  Dim Canceled
  On Error GoTo CheckError
  ...
  Dialog dlg        'Show dialog
  If Canceled Then
    ...
  End If
  Exit Sub

  CheckError:
  If Err.Number = 10031 Then
    Canceled = True
    Resume Next
  Else
    ...
  End If
End Sub
```

catmanのGUIを非表示にする方法

多くの場合、ユーザがcatmanEasy/APに組み込まれた機能に完全にアクセスできないようにすると便利な場合があります。プロジェクトの視覚パネルがあり、ユーザがいくつかのボタンをクリックすることで動作するプロジェクトが用意されていると想像してください。

catmanのGUIの個々の可視化を制御する*EA_Ribbon*クラスにいくつかの関数がありますが、すべてのGUI機能を非表示にする最も簡単な方法は、/HideGUIコマンドラインオプションを使用することです。

```
/HideGUI
```

すべてのGUI要素を非表示(リボンバー、メニューおよびコンポーネントウィンドウ)にし、VISUALIZATIONタブに切り替えます。視覚パネル上のボタンからのみの機能をユーザに許可する場合に、このモードは便利です。このコマンドラインの引数は、*/project*と接続すると最も有用です - この場合、起動画面さえ表示されません。

このモードでは、チャネルをグラフや表示器に割り当てるためのすべての組み込み機能が利用できないことに注意してください。つまり、すべてのデータソースの割り当ては、EasyScriptから実行するか、またはプロジェクトで用意する必要があります。

EasyScriptアプリケーションのユーザインタフェースを作成

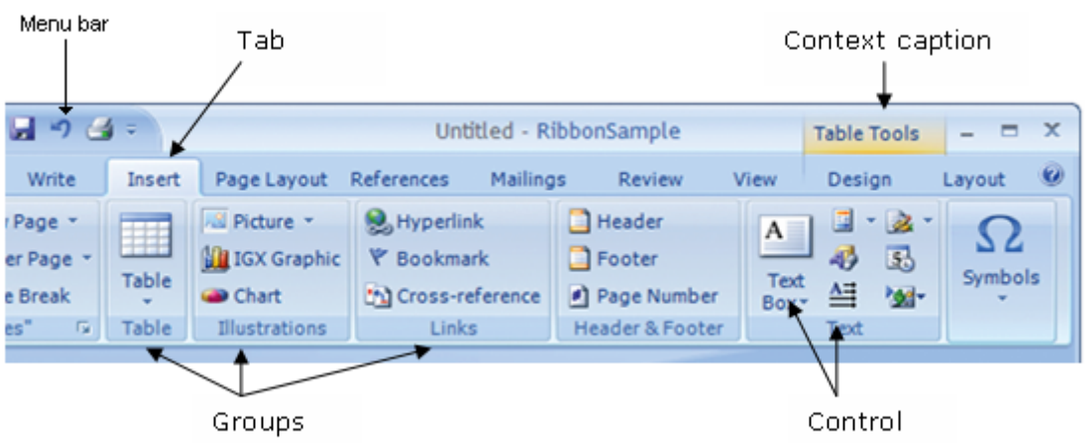
スクリプトにユーザインタフェースを提供する方法は二つあります:

1) パネル上にボタンやテキストボックスなどのオブジェクトを作成します。この仕組みはversion 2.0以降から利用可能であり、本書での説明は省略します。

2) catmanEasy/AP 2.5より導入されたリボンバーを独自のボタン、いわゆる"コントロール"で拡張します。すべてのコントロールはEasyScriptプロシージャを呼び出すことができます。テキストボックス、リストボックス(コンボボックス)やチェックボックスといった他のコントロールタイプもリボンに含めることができます。EA_Ribbonのいくつかのメソッドにより、これらコントロールのアクセスが可能になります。

この項では、リボンバーを拡張する方法について説明します。

リボンバーはタブ、グループ(タブ内)およびコントロール(グループ内)から構成されています。さらにコントロールはタブ外側のメニューバーに表示させることもできます。すべてのコントロールは、複数のサブコントロールを(ドロップダウンメニューなど)持つpopup-root-controlとして振る舞うことができます。



対話形式でコントロールを作成

リボンバーは対話形式で構成されます - スクリプト実行時でのコントロールの追加は、現在サポートしていません。

リボンのメインメインメニューから"Customize user interface"をクリックします。"Configure GUI"ダイアログが開きます。このダイアログでは、リボンに新規タブを追加、タブにグループを追加、最後にグループにコントロールを追加します。さらに、メニューバーに直接コントロールを追加することができます。

すべてのコントロールにアクションを定義することができます。しかし、EasyScriptプロジェクトでは主に"EasyScript procedure"が使用されます。

Simple action	"Start DAQ job"などの基本的なアクションを実行します。
Auto sequence	AutoSequenceを実行します。これは一つ以上のシーケンスを含むAutoSequenceプロジェクトが読み込まれていることが必要です。 EasyScriptプロジェクトでは推奨していません。
EasyScript procedure	EasyScriptプロシージャを実行します。これには一つ以上のプロシージャ(Subsまたは関数)を含むEasyScriptプロジェクトが読み込まれていることが必要です。単にプロシージャをクリックします - 緑色の実行マークが表示されます。 実行したいEasyScriptプロシージャが存在しない場合でも、問題ありません。プロシージャを割り当てていなくてもユーザインタフェースの設計を完了することができます。これは、後でEasyScript editorで実行できます
EasyScript code	コントロールに割り当てるEasyScriptコードを実行します。ダイアログの"Action"セクションの"Configure"をクリックしてコードエディタを起動します。コードはGUI拡張機能とともに保存され(この章で後述する拡張したGUIの読み込み/保存を参照)、EasyScriptプロジェクトは必要ありません。実行はSub "Main"を開始しますが、コードには Main から呼び出される他のSubや関数が含まれる場合があります。

コントロールにアクションを割り当てることは必須ではありません。チェックボックスコントロールを例にあげると、クリックした時に必ずしもアクションが実行されるわけではありません。スクリプトは代わりにチェックボックスの状態を読み取るだけです(EA_Ribbon.GetValueを参照)。

コントロールは、サブコントロールを持つことができます。サブコントロールは"DropDown"コントロールと呼ばれます。サブコントロールの制御を可能にするため、コントロールの構成セクションで該当するオプションをチェックします("Allow sub-controls")。最大2つのサブコントロールレベルをサポートします。

catmanEasy/APの組み込みタブ、またはシステムメニューを表示させるかどうかの設定も、GUI構成ダイアログで制御できます。これらのプロパティにアクセスするには、オブジェクトリスト内の"System"ノードを選択します。

すべてのコントロールに意味のある名前を割り当ててください。スクリプトコードからコントロールにアクセスするには、この名前が必要です。デフォルトでコントロールは、番号付けしたCONTROL(例えば"Control1")名前が付けられます。

"Accept"でGUI構成ダイアログを閉じると、すぐに変更が反映されます。

スクリプトでユーザインタフェースを利用可能にする

ユーザインタフェースの設計完了後、"Save"ボタンをクリックして保存します。ファイルはEA_Ribbon.LoadExtensionsよりEasyScriptに読み込むことができます。このメソッドは非常に重要で、スクリプトの先頭に呼び出さなければなりません。

コントロールにイベントプロシージャを割り当て

事前にEasyScriptプロシージャを作成し、GUI構成ダイアログを割り当てることができます。別の方法として最初にコントロールを作成し、コントロールのイベントプロシージャを作成します。このためには、EasyScript editorを開き、コントロールを右クリックします(他のタブでは右クリックしても効果はありません!)。コンテキストメニューから"Event procedure"を選択します。catmanEasy/APはその後、CONTROLNAME_ONCLICK名のプロシージャを検索します。このプロシージャが存在しない場合は作成して自動的にコントロールに割り当てます。

チャネルの初期化およびトレーサビリティデータ

計測の開始前に計測プロジェクトのすべてのチャネルを初期化する必要があります。初期化中に何が行われるのでしょうか?

1. catmanEasy/APは例えばチャネルの単位、割り当てたセンサ等、必要なデータを転送します。
2. 値のチェック、有効なアンプパラメータに変換し、デバイスに転送(ハードウェア)します。
3. トレーサビリティデータを決定します。例えばフィルタ、変換回路などと同様にアンプタイプやシリアル番号などの追加情報を、実装されているチャネルの設定です。このトレーサビリティデータは、計測データとともに後で保存するため、テストを追跡可能にします。

通常、チャネルのセンサ割り当て後や計測の開始前といった必要に応じてcatmanEasy/APにより実行します。catmanEasy/APはチャネルを初期化する必要があるかどうかを認識しています; したがって不要な初期化は発生しません。

ただし、スクリプトでEA_IO.Commandを使用してアンプの設定を実行すると、自動チャネルの初期化設定は元に戻すことができないことを確認する必要があります。実行時点としてcatmanEasy/AP: ***After channel initialization***を選択し、自動実行時間の一つにスクリプトプロシージャを割り当てることが最も簡単です。Auto-script executionせずに作業する場合、チャネルをDAQジョブ開始時の初期化や設定の上書きから防ぐためにEA_IO.NoAutoInitを使用する必要があります。

⚠ 各チャネルは計測を開始できるよう、少なくとも一度は正しく初期化する必要があります。

自動初期化から除外するチャネルはしたがって、最初に完全にEA_IO.InitializeChannel(Mode = 0)で初期化する必要があります。その後、EA_IO.Commandでアンプに独自の設定を書き込み、今度はMode = 1で再度EA_IO.InitializeChannelを実行します: この方法でトレーサビリティデータのみがcatmanEasy/APにより更新されます。

例

```
Sub SetSpecialFilter()
  'Sets a special filter frequency in an MGCplus channel
  Dim Answer
  'Execute once-only initialization!
  EA_IO.InitializeChannel "Valve pressure", 0
  'Set filter
  EA_IO.ChanSelect "Valve pressure"
  Answer = EA_IO.Command(1, "SFC 142,967")
  'Update traceability data
  EA_IO.InitializeChannel "Valve pressure", 1
  'Exclude channel from automatic initialization
  EA_IO.NoAutoInit("Valve pressure") = True
  'Start measurement
  EA_Job.Start 1
End Sub
```

他のプログラムをオブジェクトとして作成して使用

多くのプログラムは、関数をオブジェクトとして他のプログラム(したがってスクリプトも)に利用することができます。スクリプト内でこれらプログラムの関数にアクセスするため、すべてを最初に導入しておく必要があります。ここでは、二つの方法を紹介します: 遅延バインディングおよび事前バインディング.

☞ この章では一般的な手順について説明します。他のプログラムから提供される関数のドキュメントは、このヘルプの範囲外です。各プログラムの開発元より情報を入手してください。MicrosoftのOfficeについては、Microsoft Officeのヘルプおよびサポートのホームページを参照してください。

実行時にオブジェクトを生成(遅延バインディング)

外部プログラムのオブジェクトを導入するには、CreateObjectを使用します。このメソッドは、スクリプト実行時にのみプログラムオブジェクトを要求するため、*遅延バインディング*と呼ばれています。CreateObjectは、外部プログラムから提供される、いわゆる*クラス*を引数として想定しています。プログラムに応じて、複数のクラスを利用することができます。それぞれのプログラムマニュアルから情報を入手してください。いくつかのMicrosoft製品の一般的なクラス名は以下の通りです:

`Excel.Application` (MS Excel関数へのアクセス)
`Word.Application` (MS Word関数へのアクセス)
`FileSystemObject` (Windowsのファイルアクセスの関数)

CreateObjectを使用してプログラムオブジェクトを利用できるよう、最初にObject型の変数を宣言する必要があります:

```
Dim objXls As Object
Set objXls = CreateObject("Excel.Application")
objXls.Workbooks.Add
objXls.Worksheets(1).Cells(1,1).Value = "1"
```

プログラムオブジェクトが必要でなくなった場合は、閉じる必要があります:

```
Set objXls = Nothing
```

☞ 遅延バインディングの場合、CreateObjectで何が生成されるかをまだ分かっていないため、スクリプトエディタはコード入力時に外部プログラムのプロパティおよびメソッドを認識できません。そのため、命令を書き込むときのポップアップやツールチップで、有効なキーワードおよびパラメータを表示する*インテリセンス*機能は使用できません。

オブジェクトの参照(事前バインディング)

スクリプト作成時に外部のプログラムオブジェクトを導入します。これを行うには、**File**メニューから**Object references**ダイアログボックスを開き、利用可能なプログラムのリストから使用するプログラムを選択します。

☞ 初めてこのダイアログボックスを開いた場合、利用可能なプログラム数によっては数秒かかかる場合があります。

CreateObjectを呼び出す必要はありません。この方法は事前バインディングとして知られており、遅延バインディングと比較して長所と短所を併せ持っています:

- コード入力で、オブジェクトのプロパティとメソッドは、コードの開始時に利用可能(インテリセンス機能)です。
- スクリプト実行時に、メソッドおよびプロパティへのアクセスは、遅延バインディングよりも速く行われます。
- オブジェクトの参照にEasyScriptは、コードモジュールに外部プログラムオブジェクト(GUID)識別子を挿入します。その後、スクリプトを"事前"バインディングしたバージョンと互換性のない、別バージョンの外部プログラムが存在するコンピュータ上でスクリプトを実行すると、アクセスに失敗します。

事前バインディングはObjectの一般型ではなく、既知のクラスのオブジェクト参照を使用して、すでに宣言されている型で宣言します:

```
Dim objXls As Excel.Application
```

Asキーワードの入力後にインテリセンス機能は、Excelクラスの利用可能なオブジェクトのリストを表示します。変数の宣言だけでは、オブジェクトを生成しません。CreateObjectと同じ様に、Set命令をここで実行する必要があります:

```
Set xlsObj = New Excel.Application
```

簡単な方法として、宣言時に変数を生成することもできます:

```
Dim objXls As New Excel.Application
```

この場合、すぐに変数を使用することができ、後からSetで割り当てる必要はありません。

Microsoft Officeの使用

MicrosoftのOfficeプログラムは、スクリプトで簡単に使用できる多くの関数を提供しています。このセクションでは、ExcelやWordといったOfficeアプリケーションを制御する方法について説明します。これには、頻繁に使用する一例が含まれています。

☞ この章では一般的な手順について説明します。MS ExcelやMS Wordなどのプログラムから提供される関数の文書化は、このヘルプの範囲外です。Microsoft Officeのプログラミングインタフェースの説明は、Microsoft OfficeのWebページを参照してください。Officeオブジェクトを使用する場合、Office CDから"Office共有機能"の"Visual Basic for Applications"を使用するPCにインストールする必要があります。

最初に、各プログラムのオブジェクトを作成する必要があります。"他のプログラムをオブジェクトとして作成して使用"を参照してください。先に説明した通り、オブジェクトの参照を追加するか、`CreateObject`を呼び出すことにより発生します。

CreateObjectの例

```
Dim objXls As Object
Set objXls = CreateObject("Excel.Application")
Dim objWrd As Object
Set objWrd = CreateObject("Word.Application")
```

これにより、オブジェクトからOfficeプログラムのメソッドおよびプロパティに対応することができます:

```
objXls.Worksheets(1).Cells(1,1).Value = "1"
objXls.Workbooks(1).SaveAs "MyData.xls"
```

以降の例は、オブジェクトがすでに作成されており、名前が`objXls`(Excel)や`objWrd`(Word)であることを前提としています:

Microsoft Excel

新規ワークブックやワークシートを作成

新規ワークブックを作成して保存します:

```
Dim WBcount As Integer
objXls.Workbooks.Add
WBcount = objXls.Workbooks.count
objXls.Workbooks(WBcount).SaveAs "NewWorkbook.xls"
```

新規ワークシートを作成します:

```
objXls.Worksheets.Add(Before, After, Count)
```

引数	データ型	説明
Before	Variant	省略可。指定したオブジェクトのシートの直前に、新しいワークシートを追加します。
After	Variant	省略可。指定したオブジェクトのシートの直後に、新しいワークシートを追加します。
Count	Variant	省略可。追加するワークシートの番号を指定します。既定値は1です。

☞ 引数`Before`および`After`の両方を省略すると、アクティブシートの直前に新規ワークシートが追加されます。

```
objXls.Workbooks(1).Activate 1
objXls.Worksheets.Add
objXls.Worksheets(objXls.Worksheets.Count).Name = "MySheet"
```

ワークブック、ワークシートやセルの選択

Book4.xlsをアクティブにします。Book4.xlsが複数ウィンドウの場合、Book4.xls:1と、最初のウィンドウをアクティブにします:

```
objXls.Workbooks("BOOK4.XLS").Activate
```

最初のワークブックをアクティブにします:

```
objXls.Workbooks(1).Activate
```

ワークシートSheet1をアクティブにします:

```
objXls.Worksheets("Sheet1").Activate
```

ワークシートSheet1のセル範囲A1:C3を選択し、セルB2をアクティブにします:

```
objXls.Worksheets("Sheet1").Activate
objXls.Range("A1:C3").Select
objXls.Range("B2").Activate
```

現在のワークブックを表示(最前面に)します:

```
objXls.Visible = True
```

ワークブックの保存/読み込み

既存のワークブックを開きます:

```
objXls.Workbooks.Open "ANALYSIS.XLS"
```

ワークブックの変更を保存(最初の保存)します:

```
objXls.Workbooks(1).SaveAs "C:\MYDATA\TEST1.XLS"
```

次回以降の保存:

```
objXls.Workbooks(1).Save
```

保存するかどうかを確認するダイアログボックスを非表示にする場合、Savedプロパティで`True`に設定します。

セルの読み取り/書き込み

セルに値を書き込みます:

```
objXls.Worksheets(1).Cells(1,8).Value = 0.004
```

ワークシートSheet1のA1セルに3.14159と書き込みます:

```
objXls.Worksheets("Sheet1").Range("A1").Value = 3.14159
```

ワークシートSheet1のセル範囲A1:D10を確認。値が0.001未満の場合、0(ゼロ)に置き換えます:

```
For Each c In objXls.Worksheets("Sheet1").Range("A1:D10")
  If c.Value < 0.001 Then
    c.Value = 0
  End If
Next c
```

セルの書式設定

セル範囲の色やフォントを変更します:

```
With objXls.Worksheets("Sheet1")
  .Range("A1:B5").Font.Name = "Verdana"
  .Range("A1:B5").Font.Bold = True
  .Range("A1:B5").ForeColor = vbRed
  .Range("A1:B5").BackColor = vbYellow
End With
```

Microsoft Word

新規文書の作成

デフォルトのテンプレートに基づいて新規文書を作成します:

```
objWrd.Documents.Add
```

他の方法としてSetコマンドも使用できます。これはその後の文書の変更を容易にするため、オブジェクトとして変数に保存されます:

```
Set newDoc = objWrd.Documents.Add
With newDoc
  .Content.Font.Name = "Arial"
  .SaveAs FileName:="Sample.doc"
End With
```

文書を開く

文書MyDoc.docを読み取り専用で開きます:

```
objWrd.Documents.Open FileName:="C:\MyFiles\MyDoc.doc", ReadOnly:=True
```

文書の保存

文書Sales.docを保存します:

```
objWrd.Documents("Sales.doc").Save
```

開いているすべての文書を保存します:

```
objWrd.Documents.Save
```

アクティブなドキュメントを別名で保存します:

```
myDoc.SaveAs FileName:="Test.doc"
```

文書を閉じる

ドキュメントSales.docを閉じます:

```
objWrd.Documents("Sales.doc").Close
```

文書内のRangeオブジェクトを作成

Rangeオブジェクトを作成し、myRange変数に割り当てます:

```
Set newDoc = objWrd.Documents.Add
Set myRange = newDoc.Range(Start:=0, End:=10)
```

上記、myRangeはアクティブな文書の最初の10文字を参照します。StartおよびEndの両方を省略すると文書全体として設定されます:

```
Set myRange = newDoc.Range
```

クリップボードの内容を文書に貼り付け

クリップボードの内容をmyRangeに代入します:

```
myRange.Paste
```

範囲myRangeの末尾にクリップボードの内容を貼り付けます:

```
myRange.Collapse wdCollapseEnd
myRange.Paste
```

Microsoft Excelを使用せずにExcelの基本操作を実行する

この章では、catmanインストールに付属するOLE XLSFile ActiveXライブラリを使用して、ワークシートセルへのデータの読み書きなどの基本的なExcel操作を実行する方法について説明します。主な利点は、コンピュータにMicrosoft Excelがインストールされていなくても、97-2003形式(.xls)の Excelファイルを操作できることです。

Microsoft Excelがすでにコンピュータにインストールされている場合は、Microsoft Officeアプリケーションが Excelファイルを操作するための最も効率的かつ強力な方法を提供するため、OLE XLSFile ActiveXオブジェクトを使用する必要はありません。「Microsoft Officeの使用」も参照してください。

まず、Visual BasicのCreateObjectメソッドを使用して、XLSFileObject オブジェクトのインスタンスを作成する必要があります。他のプログラムをオブジェクトとして作成して使用も参照してください。

```
Dim objXFO As Object
Set objXFO = CreateObject("olexlsf.XLSFileObject")
```

OpenFileメソッドで Excel ファイルが正常に開かれると、ブック、シート、セルなどを操作できるようになります。

```
Dim objXFile As Object
Set objXFile = objXFO.XLSFile
objXFile.OpenFile("MyData.xls")
```

ファイルに対する操作が完了したら、データをファイルに保存する必要があります。

```
objXFile.SaveAs("NewWorkbook.xls")
```

最後にオブジェクトをNothingに設定して、ファイルに関連付けられているリソースを解放します。

```
Set objXFile = Nothing
Set objXFO = Nothing
```

次の例では、必要なオブジェクト(objXFO、objXFile)がすでに作成されていることを前提としています。

ワークシートオブジェクトのメソッドとプロパティ

現在のワークブック内のシート数を取得:

```
Dim SheetCount As Integer
SheetCount = objXFile.Workbook.Sheets.Count
```

インデックスまたは名前で指定された既存のシートを選択してアクティブ化:

```
Dim objXSheet As Object
Set objXSheet = objXFile.Workbook.Sheets.Item(Worksheet)
Set objXSheet = objXFile.Workbook.Sheets.ItemByName(Worksheet)
```

引数	データ型	説明
Worksheet	Integer [in]	シートのインデックス。インデックスは1から始まります。
	String [in]	シート名。

指定した名前の新しいシートをワークブックに追加:

```
objXFile.Workbook.Sheets.Add(SheetName)
```

引数	データ型	説明
SheetName	String [in]	新規シート名。

インデックスまたは名前で指定されたシートを削除:

```
objXFile.Workbook.Sheets.Delete(Worksheet)
objXFile.Workbook.Sheets.DeleteByName(Worksheet)
```

引数	データ型	説明
Worksheet	Integer [in]	シートのインデックス。インデックスは1から始まります。
	String [in]	シート名。

既存のシートのコピーを作成:

```
objXFile.Workbook.Sheets.Copy(SourceSheetIndex, DestSheetIndex, NewSheetName)
```

引数	データ型	説明
SourceSheetIndex	Integer [in]	シートのインデックス。インデックスは1から始まります。
DestSheetIndex	Integer [in]	新規シートが配置される位置。インデックスは1から始まります。
NewSheetName	String [in]	新しいシート名。

次の例では、最初のワークシート (シートのインデックス = 1) がでも操作のために選択されていることを前提としています。

```
Dim objXSheet As Object
Set objXSheet = objXFile.Workbook.Sheets.Item(1)
```

セルの読み取り/書き込み

Sheet1のセルA1に3.14159を書き込み:

```
objXSheet.Cells.Cell(1, 1).Value = 3.14159 '行と列で指定したセル。
objXSheet.Cells.CellByA1Ref("A2").Value = "A1スタイルの参照文字列で指定したセル。
```

Sheet1のセルB1の値を読み取り:

```
Dim dCellValue As Double
Dim sCellValue As String
dCellValue = objXSheet.Cells.Cell(1, 2).Value
sCellValue = objXSheet.Cells.CellByA1Ref("B1").Value
```

セルの書式設定

例:

```
Sub ExcelNoAutomation
  Dim objXFO As Object
  Dim objXFile As Object
  Dim objXSheet As Object
  Dim dCellValue As Double
  Dim sCellValue As String
  Set objXFO = CreateObject("olexlsf.XLSFileObject")
  Set objXFile = objXFO.XLSFile
  objXFile.OpenFile("C:\Users\catman\Documents\HBM\catmanEasy\DATA\MyData.xls")
  Set objXSheet = objXFile.Workbook.Sheets.Item(1)
  objXSheet.Rows.Item(1).Height = 40.5
  objXSheet.Columns.Item(1).Width = 22.75
  objXSheet.Cells.Cell(1, 1).Value = 3.14159
  objXSheet.Cells.CellByA1Ref("A2").Value = "some text"
  dCellValue = objXSheet.Cells.Cell(1, 1).Value
  sCellValue = objXSheet.Cells.CellByA1Ref("A2").Value
  With objXSheet.Cells.Cell(1, 1)
    .FontName = "Verdana
    .FontHeight = 16
    .FontBold = True
    .FontItalic = True
    .FontColorRGB = vbBlue
    .FillPattern = 1
    .FillPatternFGColorRGB = vbRed
    .FillPatternBGColorRGB = vbRed
```

```
  End With
  objXFile.SaveAs("C:\Users\catman\Documents\HBM\catmanEasy\DATA\NewWorkbook.xls")
   Set objXSheet = Nothing
   Set objXFile = Nothing
   Set objXFO = Nothing
End Sub
```

FileSystemオブジェクトを使用

このオブジェクトは、PC上のファイルシステムにアクセスする多くの関数を公開しています。FileSystemオブジェクトを使用するとファイル、フォルダ(ディレクトリ)およびドライブにアクセスすることができます。

> FileSystemオブジェクトは高度な操作が必要です。ファイルおよびディレクトリの基本的な操作は、VBAリファレンスのFileおよびUser Inputの関数で十分です。

"他のプログラムをオブジェクトとして作成して使用"を参照し、FileSystemを作成します。以下のようにCreateObjectを使用します。

```
Dim objFileSys As Object
Set objFileSys = CreateObject("Scripting.FileSystemObject")
```

これで、オブジェクトからファイルシステムのメソッドやプロパティにアクセスすることができます:

```
CheckFile = objFileSys.FileExists "C:\catmanEasy\DATA\Test1.bin"
If CheckFile Then
  objFileSys.CopyFile "C:\catmanEasy\DATA\Test1.bin", "\\ServerPC\TestData"
End If
```

ファイルシステムの一部(ファイルやフォルダなど)に特定のタスクを実行するため、FileSystemの*サブオブジェクト*を作成します。これらのサブオブジェクトは、VBA関数のSetを使用して作成します:

```
Set f = objFileSys.GetFile("C:\catmanEasy\DATA\Test1.bin") 'Creates a File object
FileDate = f.DateCreated
```

以降の例は、objFileSys(ファイルオブジェクト)、objFolder(フォルダオブジェクト)およびobjDrive(ドライブオブジェクト)がすでに作成されていることを前提としています。

ファイルシステム

ファイルが存在するかどうかの確認

指定したファイルが存在する場合、Trueを返します:

```
objFileSys.FileExists(filespec)
```

引数	説明
objFileSys	FileSystemObjectの名前。
filespec	存在するかどうかを調べるファイルの名前。ファイルが現在のアクティブフォルダにない場合、完全なパスを指定(絶対または相対)する必要があります。

ファイルのコピー

一つ以上のファイルを、別の場所にコピーします:

```
objFileSys.CopyFile(source, destination[, overwrite])
```

引数	説明
objFileSys	FileSystemObjectの名前。
source	コピーするファイルのパスおよびファイル名(文字列)を指定します。ワイルドカードが使用できます。
destination	コピー先のパス(およびファイル名)を指定します。ワイルドカードは使用できません。パスのみを指定した場合(末尾のバックスラッシュを付けずに指定した場合)、元のファイル名を使用します。
overwrite	省略可能。True(デフォルト値)を指定するとファイルは上書きされます。 コピー先が読み取り専用の場合、CopyFileは失敗します。

フォルダが存在するかどうかの確認

指定したフォルダが存在する場合、True を返します:

```
objFileSys.FolderExists(folderspec)
```

引数	説明
objFileSys	FileSystemObjectの名前。
folderspec	存在するかどうかを調べるフォルダ名。カレントフォルダに無いフォルダを調べる場合、完全なパスを指定(絶対または相対)する必要があります。

フォルダの作成

```
objFileSys.CreateFolder(folderspec)
```

引数	説明
objFileSys	FileSystemObjectの名前。
folderspec	作成するフォルダ名を指定します。フォルダが現在のアクティブフォルダにない場合、完全なパスを指定(絶対または相対)する必要があります。

☞ 指定したフォルダが既に存在する場合、エラーが発生します。

ファイルオブジェクトの指定

ファイルオブジェクトを返します。

```
objFile = objFileSys.GetFile(filespec)
```

引数	説明
objFileSys	FileSystemObjectの名前。
filespec	ファイルのパスを指定(絶対または相対)します。

☞ 指定したファイルが存在しない場合、エラーが発生します。

フォルダオブジェクトの指定

ファイルオブジェクトを返します。

```
objFolder = objFileSys.GetFolder(folderspec)
```

引数	説明
objFileSys	FileSystemObjectの名前。
folderspec	ファイルのパスを指定(絶対または相対)します。フォルダが現在のアクティブフォルダにない場合、完全なパスを指定(絶対または相対)する必要があります

☞ 指定したフォルダが存在しない場合、エラーが発生します。

ドライブオブジェクトの指定

ドライブオブジェクトを返します。

```
objDrive = objFileSys.GetDrive(drivespec)
```

引数	説明
objFileSys	FileSystemObjectの名前。
drivespec	ドライブ名を指定します。ドライブ名(C)、コロン付きのドライブ名(C:)、コロンとパスの区切り文字の付いたドライブ名(C:\)、または任意のネットワーク共有名(\\computer2\share1)のいずれかで指定します。

☞ 指定したドライブが存在しない場合、エラーが発生します。

ファイル

ファイルの作成日を取得

指定したファイルまたはフォルダの作成日時を返します(読み取り専用):

```
Date = objFile.DateCreated
```

objFileはファイルオブジェクトでなければなりません。GetFileを参照してください。

最終更新日時を取得

指定したファイルまたはフォルダが最後に更新された日時を返します(読み取り専用):

```
LastChange = objFile.DateLastModified
```

objFileはファイルオブジェクトでなければなりません。GetFileを参照してください。

パスの取得

```
PathName = objFile.Path
```

objFileはファイルオブジェクトでなければなりません。GetFileを参照してください。

ドライブ

空き領域を取得

指定したドライブまたはネットワーク共有で利用可能なディスクの空き容量を返します(読み取り専用):

```
FreeSpace = objDrive.FreeSpace
```


ディスククオータを設定している場合、AvailableSpaceメソッドを使用します。

objDriveはドライブオブジェクトでなければなりません。GetFileを参照してください。

ドライブ名を取得

指定したローカル ドライブまたはネットワーク共有のドライブ名を返します(読み取り専用):

```
DriveLetter = objDrive.DriveLetter
```

objDriveはドライブオブジェクトでなければなりません。GetFileを参照してください。

DLL関数の呼び出し

このセクションでは、ダイナミックリンクライブラリ(DLL)に実装された関数を呼び出す方法について説明します。関数ライブラリの最も身近な例として、数千もの関数を実装しているWindows API(Application Programming Interface)が挙げられます。DLL関数の他の例として、データ収集カードや数学ライブラリなどのドライバライブラリがあります。

☞ この章では一般的な手順について説明します。Windows APIによって提供される関数のドキュメントは、このヘルプの範囲外です。各DLLの開発元より情報を入手してください。

DLL関数を呼び出すには、Declareで宣言する必要があります。次の例は、メッセージを出力するWindows APIのMessageBox関数を宣言しています。

```
  Declare Function MessageBox Lib "user32" Alias "MessageBoxA"(ByVal hwnd As Long,
ByVal lpText As String, ByVal lpCaption As String, ByVal wType As Long) As Long
```

最初に想定する関数名を宣言し、DLL(例では"user32.dll")と実装している関数が続きます。別名で関数を利用したい場合、"エイリアス"を使用します: 例では、DLL内の実際の関数名は"MessageBoxA"ですが、エイリアスとして"MessageBox"を使用します。

これ以降、自分で作成した他のプロシージャと同様に、スクリプト内でこの関数を利用することができます:

```
  Const MB_OK = 0
  MessageBox 0&, "Hello world", "My message box", MB_OK
  REM As the first parameter has to be provided as Long, 0& is used instead of a
  REM simple 0 (abbreviation for conversion into a Long value)
```

DLLを宣言する上での重要な点は、想定する関数パラメータを正しく指定することです。DLLは通常C言語で書かれており、CとVBA間での変数の型の違いが生じる場合があります。

⚠ パラメータリストの正しい宣言は、DLLの呼び出しが正しく機能するために極めて重要です。一般的に、間違った宣言はプログラムのクラッシュにつながります。つまり、スクリプトだけでなく、catmanEasy/APもクラッシュします。

正しい実装の例:

C	VBA
値渡し(Cのデフォルト)	プレフィックスキーワードByValキーワード。VBAのデフォルト設定で参照(ポインタ)として渡されるため、*必ず*指定する必要があります。
参照渡し(ポインタ、例えばchar *)	メソッドに渡すVBAのデフォルト設定で特別なキーワードを必要としません。オプションでByRefを使用することができます。
NULLで終端させる文字列渡し(char *)	*常に*VBAで"ByVal VarName As String"と宣言されます。 これらは常に VBA で「ByVal VarName As String」で宣言されます。
char型、BYTE	VBAのデータ型はByte
short型、ushort、WORD	VBAのデータ型はInteger
long型、ulong、DWORD	VBAのデータ型はLong
float型	VBAのデータ型はSingle
double型	VBAのデータ型はDouble
large integer型	VBAでの対応はありません。Doubleで実現できますが、バイト操作によって正しい表現に変換する必要があります。

構造体(struct)	原則として可能です。Typeを使用して、VBA側のユーザ定義型に定義する必要があります。パラメータリストのパラメータはその後、"As Typname"として表します。 このような型を予期する関数は、例外的な場合にのみ、型を正確に知った上で使用する必要があります。たとえば、CとVBAの"byte alignment"の違いによって問題が発生することがよくあります。
任意の型へのポインタ(void *)	"As Any"パラメータを宣言します。
共用体、ビットフィールド	VBAでの対応はありません。
フィールド(配列)渡し	C言語では、このようなパラメータはポインタ(例えばdouble *)として宣言し、呼び出されたときフィールドの先頭(フィールド内の最初の要素)が指定されることを想定します。VBAのフィールドはCとは異なる構造のため、単にフィールド名を指定しただけでは呼び出せず、例えば"MyArray(0)"など、パラメータとして*最初の*要素を指定する必要があります。通常通り、パラメータをポインタとして指定します: "..., MyArr As Double, ..."。

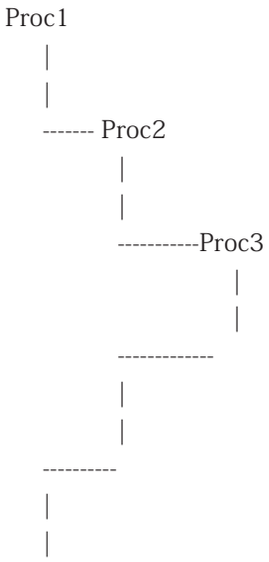
注記

- VBAでは符号なしの型を認識しません。このようなパラメータを期待する関数は、期待される値の範囲が正確にわかっている場合にのみ使用してください。
- *戻り値*にも正しい型が使用されていることを確認してください。。
- DLL関数の文字列(char *)を返す呼び出し元、すなわちスクリプトのが必要なメモリを提供する必要があります。十分なメモリ空間を確保していない場合、一般的にスクリプトだけでなくcatmanEasy/APもクラッシュします。以下の例は、Windowsのシステムディレクトリ名を検索します:

```
    Public Declare Function GetSystemDirectory Lib "kernel32" Alias
"GetSystemDirectoryA"(ByVal lpBuffer As String, ByVal nSize As Long) As Long
    Dim lpBuffer As String
    lpBuffer = Space$(256)  'Provide memory space
    GetSystemDirectory lpBuffer, 256
```

入れ子のプロシージャ呼び出し

EasyScriptは、プロシージャを実行するとイベントに反応します。このようなイベントは、ユーザがパネルのボタンをクリックしたり、自動プロシージャの実行時点で行われます。通常、このイベントは別のプロシージャを呼び出します。元のプロシージャを停止し、その後、新しいプロシージャに進みます。このプロシージャが終了すると、最初のプロシージャを再開します。二番目のプロシージャの実行中に三番目が呼び出されると、入れ子になることが想像できます。



深い入れ子は、スクリプトに予測不可能な振る舞いや反応を引き起こす可能性があります。ボタンや自動プロシージャによる中断で、このようなプロシージャが起動することを避けなければなりません。

- 長いプロシージャの実行中にボタンを無効にする(EA_Panel.SetAccessMode)。
- 最初のプロシージャを実行すると設定され、別のプロシージャが呼びされると終了するグローバルフラグを使用します。

例えば、長いループを中断するような単純な割り込みは、何ら問題ありません。

AddIn

"AddIn"とは、catmanEasy/APの機能を拡張するActiveX(COM)コンポーネントを表します。これを行うには、"Object Referencesダイアログ"で説明したように、コンポーネントはcatmanEasy/APを使用します。C#、VB.Net、Visual Basic 6、Visual C++やDelphiなどを使用してCOMコンポーネントを作成するスキルを持った方であれば、catmanEasy/APに簡単に強力な新機能を追加することができます。この方法は、catmanEasy/APの統合したEasyScript環境を使用する場合と比較して、いくつかの利点があります:

- C#、VB.Net、VB、DelphiやC++などの成熟した言語で、EasyScript VBAよりもさらに強力です。
- AddInは、任意の複雑な可視化コンポーネント(ダイアログ)含めることができます。
- AddInは、ネイティブコードにコンパイルすることができるため、EasyScriptよりも高速に実行できます。

- C#、VB.Net、VB、Delphi、C++ などの完全に成長した言語で、EasyScript VBA よりもさらに強力です
- アドインには、任意の複雑さのビジュアル コンポーネント (ダイアログ) を含めることができます。
- アドインはネイティブ コードにコンパイルできるため、EasyScript よりもはるかに高速に実行できます。

AddInを使用する典型的な例として、独自の表示器やグラフオブジェクトの可視化オブジェクトギャラリーの拡張や、特殊なファイル形式へのデータのエクスポートなどがあります。catmanのデフォルト作業のディレクトリ(通常はドキュメント/HBM/catmanEasy/)のEXAMPLES/AddInsフォルダに、AddInのテンプレートがあります。

以降で説明する手順に沿って、簡単にActiveXコンポーネントをcatmanEasy/APに拡張することができます:

- 選択した開発システムでActiveX/COMプロジェクト(インプロセスDLLまたはアウトオブプロセスEXEの何れか)を作成します。Microsoft .NETのいずれかを使用している場合、いわゆるCOMラッパーを作成する必要があります。Microsoft VisualStudioであれば、これを支援しています。
- 三つのパブリックメソッドを追加します: AddcatInterface、AddEasyInterfacesおよびRun。これらはcatmanがコンポーネントを初期化すると呼び出されます。
- AddInが可視化コンポーネントまたはファイルエクスポートフィルタの場合、オプションのメソッドを追加します。
- 実装したい関数のコードを、Runメソッドから追加します。
- コンポーネントをコンパイルし終えたら、catmanディレクトリのADD_INS.CATファイルを追加します。クラス名とAddInの説明とともにFLAGSパラメータで、アドインの外観とアクセスを制御します。

以降のセクションでは、より詳細に手順を説明します。すべてのコード例は、MS Visual Basic 6を前提としていますが、これは容易に他の言語に変換することができます。catmanのデフォルト作業のディレクトリ(通常はドキュメント/HBM/catmanEasy/)のEXAMPLES/AddInsフォルダに、アドインのテンプレートがあります。

mandatory属性のパブリックメソッド

catmanでコンポーネントを実行するには、コンポーネントのメインクラスに三つのメソッドをエクスポートする必要があります("main class"はADD_INS.CATファイルで指定されていることを意味します - catmanは内部目的に使用する他のクラスに関係なく、このクラスのみインスタンスを作成します)。

AddcatInterfaceおよびAddEasyInterfacesの二つのメソッドは、catmaEeasyよりオブジェクトが渡されます。データ型"As object"を指定(この場合、インテリセンスは提供されません)するか、正確なオブジェクト型を指定(この場合、インテリセンスがサポートされます)します。二番目の方法(事前バインディング)では、プロジェクトに以下の参照を追加する必要があります:

- catmanEasy/AP (catmanEasy.exe)
- HBM catman application (catman_AP.dll

参照を準備しておくと、VBのインテリセンス機能によりcatmanEasy/APのデータ型のリストを表示し、メソッドの宣言に使用することができます。

2 つのメソッド AddcatInterface および AddEasyInterfaces には、catmanEasy によってオブジェクトが渡されます。データ型「オブジェクトとして」を指定するか (この場合、インテリセンスは提供されません)、または正確なオブジェクト タイプを指定する (この場合、インテリセンスがサポートされます) ことができます。2 番目のアプローチ (「早期バインディング」と呼ばれる) では、次の参照をプロジェクトに追加する必要があります。

- catmanEasy/AP (catmanEasy.exe)
- HBM catman アプリケーション (catman_AP.dll)

AddEasyInterfaces

この方法は、AddInの読み込み時にcatmanEasy/APによって呼び出されます。catmanEasy/APよりエクスポートした主要な七つのクラス(例えばEA_DAQやEA_Job)へcatmanEasyは参照オブジェクトのポインタを渡します。これらのオブジェクトポインタをグローバルオブジェクト変数に割り当てます。

```
  Public Sub AddEasyInterfaces(o1 As object, o2 As object, o3 As Object, o4 As
Object, o5 As object, o6 As object, o7 As object, o8 As object, o9 As Object)
```

または事前バインディング

```
  Public Sub AddEasyInterfaces(o1 As clsEasy, o2 As clsIO, o3 As clsDAQ, o4 As
clsJob, o5 As clsTest, o6 As clsPanel, o7 As clsGraph, o8 As clsRibbon, o9 As
clsComm)
```

例(事前バインディングと仮定した場合):
コードモジュール(.BAS)は以下のパブリックオブジェクト変数を追加します:

```
  Public EA As clsEasy
  Public EA_IO As clsIO
  Public EA_DAQ As clsDAQ
  Public EA_Job As clsJob
  Public EA_Test As clsTest
  Public EA_Panel As clsPanel
  Public EA_Graph As clsGraph
  Public EA_Ribbon As clsRibbon
  Public EA_Comm As clsComm
```

オブジェクト名は、EasyScriptと本ヘルプファイル参照セクションで使用するオブジェクト名が一致するよう選択します。メインクラスでは、以下のコードを追加します:

```
PublicSub AddEasyInterfaces(o1 As clsEasy, o2 As clsIO, o3 As clsDAQ, o4 As clsJob,
o5 As clsTest, o6 As clsPanel, o7 As clsGraph, o8 As clsRibbon, o9 As clsComm)
  Set EA = o1
  Set EA_IO = o2
  Set EA_DAQ = o3
  Set EA_Job = o4
  Set EA_Test = o5
  Set EA_Panel = o6
  Set EA_Graph = o7
  Set EA_Ribbon = o8
  Set EA_Comm = o9
End Sub
```

次の例で示すように、コンポーネント内のどこからでもcatmanEasy/APインタフェースに、自由にアクセスすることができます。

```
Public Sub MeasureChannel(ByVal ChannelName As String)
Dim MV As Double
Dim Status As Long
EA_IO.Measure ChannelName, MV, Status
End Sub
```

catmanインタフェースの詳細な参照については、"EasyScriptリファレンス"を参照してください。

AddcatInterface

catmanEasy/APはAddInの読み込み時に、下層のcatmanカーネルオブジェクトへオブジェクト参照を渡し、メソッドを呼び出さします。このオブジェクトはしかし、非常に特殊なタスクにのみ有用で、必要になることはほとんどありません。

⚠ catmanカーネルおよびcatmanEasy/APがどのように連携するかといった十分な知識を必要とするため、このオブジェクトの使用を推奨していません。

```
  Public Sub AddcatInterface(catman As Object)
```

このオブジェクトとメソッドについては、catInterface.chmファイルを参照してください。

Run

このメソッドは、一度すべてのオブジェクト参照が渡されてcatmanEasy/APにより呼び出されます。これはコンポーネントのコードの出発点となります。ここは任意の初期化を行う場所です。AddInが自動実行にのみ依存していたり(catmanEasyの制御フローにAddInをフックを参照)、リボンバーのコントロールにリンクされたメソッドを提供する場合、Runメソッドに任意のコードを含める必要はありません。

次の例は、メソッドがフォームを呼び出す例を示しています:

```
Public Sub Run()
  frmMySpecialDialog.Show
End Sub
```

コンポーネント内のさらなるコードはその後、このフォームのイベントルーチンで実行されます(例えばユーザがボタンをクリック)。

⚠ Runメソッドが終了するまで制御がcatmanEasy/APに戻らないことに注目してください! 例えば、メソッドがモーダルダイアログを表示する場合、ユーザはcatmanEasy/APに制御が戻る前にダイアログを閉じる必要があります。

catmanEasyの制御フローにAddInをフック

多くの場合、catmanEasy/AP DAQフローの特定の段階でcatmanEasy/APのAddInに含まれているメソッドを呼び出したいことがあります(例えばDAQジョブの開始前やすべてのデータ転送ブロック後)。これはAutomatic Procedure Executionに対応しています。これを行うには、AddIn内に一つ以上のパブリックメソッドが必要になります:

```
Public Sub AutoExecute (ByVal ExecStage As Integer, Param As Variant)
```

このAddInが読み込まれると、次の実行ステージでcatmanEasy/APにより、自動的に呼び出されます。

実行ステージ	説明
0	DAQジョブの開始前。EA_Job.CurrentJobを使用して、現在開始しようとしているジョブ番号を確認します。
1	ジョブの終了後。
2	DAQの初期化前(チャネルの初期化)。
3	DAQの初期化後(チャネルの初期化)。
4	DAQの開始前。
5	DAQの停止後。
6	テストの読み込み後(解析モード)。
7	可視化パネルの変更後。
20	DAQデータ転送ブロック後(すなわちデバイスからの転送後)。これは通常、リアルタイム操作(監視など)を実行する際に、AddInが接続される場所です。
22	データ転送ブロックの保管(一時的)後。
23	すべてのDAQジョブの繰り返し後。
24	すべてのDAQジョブの終了後。
25	catmanEasy/APのシャットダウン前
26	catmanのDAQまたはAnalysisプロジェクトの読み込み前。*Param*がプロジェクト名に含まれています。
27	catmanのDAQまたはAnalysisプロジェクトの読み込み後。*Param*がプロジェクト名に含まれています。
28	catmanのDAQまたはAnalysisプロジェクトの保存前。*Param*がプロジェクト名に含まれています。
29	catmanのDAQまたはAnalysisプロジェクトの保存後。*Param*がプロジェクト名に含まれています。
30	catmanのプロジェクトデータ**書き込み**後、プロジェクトを保存時にプロジェクトファイルを閉じる前。これはファイル内に独自のバイナリデータを含める機会を与えています。*Param*には開いているプロジェクトファイルのファイルハンドルが含まれています。 注記: • AddInのバイナリデータは、実行ステージ31(下記を参照)で正確に取得できるように、一意の識別子で始まらなければなりません。 • ファイルのハンドルがAddInがDLL("InProcess Server")である場合にのみ使用できます。ActiveX EXE("out of process server")のハンドルは通常、AddInのプロセスでは有効になりません。
31	すべてのcatmanのプロジェクトデータ**読み込み**後、プロジェクトを読み込み時にプロジェクトファイルを閉じる前。これはファイル内に独自のバイナリデータを含める機会を与えています。*Param*には開いているプロジェクトファイルのファイルハンドルが含まれています。 注記: ファイルのハンドルがAddInがDLL("InProcess Server")である場合にのみ使用できます。ActiveX EXE("out of process server")のハンドルは通常、AddInのプロセスでは有効になりません。

32-38	予約。
39	この実行ステージは、グラフカーソルを移動した場合、またはビデオ再生で新しいフレームを表示した場合に呼び出されます。カーソルがマップまたは角度同期表示内に配置されている場合にも呼び出されます。現在位置のタイムスタンプ(秒単位)が *Param* に渡されます。
40	制限値/イベント監視からのイベントが発生した場合に発生します。イベントの名前は*Param*で渡されます。
Param	実行ステージに応じてcatmanにより渡される追加データ。これは個々の実行ステージに記載されています。

この方法により、アクションを自由に実行することができます。実行中のDAQジョブ(すなわち実行ステージ20(転送ブロック後)および22(転送ブロックの保管後))のコードによって、caamanからの新しいデータ転送をブロックされることに注意してください。つまり、コードはできるだけ速く返す必要があります。一般的にコードを完了するために利用できる時間を50 msかかると考えてください。コードに時間がかかりすぎると、catmanドライバの内部バッファかハードウェアでオーバーフローする可能性があります。ただし、通常これらバッファは最大数秒のデータを保持することができます。

catman バージョン5.3では、catmanプロジェクトチのャネル内容が変更された場合にもアドインに通知されます(つまり、計算チャネルの作成/削除、テストの読み込み/削除、チャネル名の変更)。これを機能させるには、アドインにメソッド *RefeshSignalList* を実装する必要があります。

```
Public Sub RefreshSignalList()
```

この関数では、次のcatmanメソッドとプロパティのいずれかを使用して、利用可能なチャネルを自由に調べることができます。

DAQモード

- *EA_IO.ChanCount*
- *EA_IO.ChannelAttribute*

解析モード

- *EA_Test.TestCount*
- *EA_Test.TestAttribute*
- *EA_Test.ChannelAttribute*
- *EA_Test.MathChanCount*
- *EA_Test.MathChannelAttribute*

catmanEasy/APのユーザインタフェースを拡張

注意: ユーザがAddInを起動するコントロール(アイコン付き)を使用するだけの場合場合、このセクションを読む必要はありません。これはADD_INS.CATファイルののFLAGSおよびICONパラメータにより、容易に実現することができます。詳細は"AddInの登録および起動"を参照してください。

このセクションでは、AddInをリボンバーコントロール(AddInによって追加)から呼び出す方法について説明します。独自のリボンタブ、タブ内のグループおよび、グループ内のコントロールをcatmanのリボンバーに追加するのは非常に簡単です。

基本的な要素はコントロール(通常は"ボタン")です。とりわけボタンを作成する場合、ユーザがコントロールをクリックした時に呼び出されるメソッド名をAddInに指定します。このメソッドの名前は例えば、"SpecialMathPack.clsMain.ComputeFFT"のように、クラス名を先行させなければなりません。詳細は"AddInの登録および起動"を参照してください。

コントロールを追加する前に、コントロールをホストするグループを作成する必要がありますが、その前にグループをホストするリボンタブを作成する必要があります。他の方法として、内蔵されている"System tabs"(インスタンスのようなチャネル構成タブ)を選択して追加することもできます。

カスタムタブの追加: EA_Ribbon.AddTab
カスタムタブにグループを追加: EA_Ribbon.AddGroupToCustomTab
Systemタブにグループを追加: EA_Ribbon.AddGroupToSystemTab
グループにコントロールを追加: EA_Ribbon.AddControl

オブジェクトの追加後は、新しいタブのためにEA_Ribbon.Refreshを呼び出して、グループやコントロールを表示させます。最初にすべてのオブジェクトを追加してから、一度*Refresh*を呼び出すことを推奨しています。

ユーザインタフェース項目は作成後に削除できないことに注意してください。これらはcatmanEasy/APが終了するまで存在します!

AddInの開放

AddInが作業を完了し、開放する必要があるとき、すなわちメモリから削除するには、catmanEasy/APが明確に決定しなければなりません。AddInのタスクを完了したことをcatmanEasy/APは知ることができないため、AddInはジョブが終了して安全に開放できることをcatmanに伝えなければなりません。この目的のためにAddInのメインクラスの*CanUnload*プロパティを提供します。プロパティはAddInのメインクラスに存在する必要があります(ADD_INS.CATファイルで指定した"main class"を意味します)。AddInがcatmanEasy/APから解放できる場合、戻り値はTRUEを返します。

注意: AddInが複数のインスタンスを処理する場合(AddInの複数インスタンスを参照してください)、AddInの最後のインスタンスが閉じられない限り、*CanUnload*を通知してはなりません。

このために通常はコードモジュール(.BAS)内で、AddInが終了するとTRUEに設定されるグローバル変数を宣言します。

```
Public mCanUnload As Integer
```

メインクラスのプロパティ関数は、次のように記述できます。

```
Public Property Get CanUnload() As Variant
  CanUnload = mCanUnload
End Sub
```

次の例は、ユーザがフォームを閉じるときにAddInも閉じることを前提としています。

```
Private Sub Form_Unload
  mCanUnload = TRUE
End Sub
```

AddInを開始すると、catmanは周期的(数秒ごと)にこのプロパティをチェックします。プロパティがTRUEを返すと、catmanはAddInコンポーネントを開放します。このようなプロパティを公開しないアドインは、AddInはcatmanEasy/APが終了するまで、メモリに残っています。catmanEasy/APのシャットダウン時、すべてのAddInsはこの時点で自動的に開放を開始します - *CanUnload*プロパティに関係なく。

⚠ catmanはコンポーネントのオブジェクト参照を保持しているため、AddInは自身を開放することができません。COMコンポーネントは、オブジェクト参照カウンタがゼロになるまでは開放することができません。

AddInの登録および起動

catmanEasy/APは起動時に、catmanEasy/APのデフォルトの開始ディレクトリにあるADD_INS.CATファイルを検索します(通常はUsers\Username\HBM\catmanEasy)。このファイルの各行は、説明文と、コンポーネントのクラス名を"," で区切ったプレーンテキストです。説明文には"NAME="を、クラス名には"CLASS="を必ず付加しなければなりません。catmanは、このファイルで見つかった行ごとにcatmanEasy/AP AddIn Managerにエントリを作成します("File/AddIn Manager...")。マネージャより提示されたエントリのリストをクリックしてAddInを起動します。

ファイルの任意の行に追加の識別子(","で区切る)含めて、catmanEasy/AP AddIn Managerに追加情報を渡します。次の表は、ADD_INS.CATファイルのエントリのコンポーネントについて説明します。パラメータ名とその値は、常に等号(=)で区切られます。値として文字列(テキスト)を持つパラメータは、二重引用符で囲む必要はありません。

コンポーネント	説明
NAME	AddIn名。AddInをcatmanのリボンバーや可視化オブジェクトギャラリーに表示する必要がある場合に使用します。
CLASS	AddInのクラス名。Windows Registry内のCOMオブジェクトを識別するために使用します。予想されるクラス名はコンポーネント名で構成されます(例えば、このVBのActiveX DLLはプロジェクト名です)。パブリックメソッド(例えば*Run*)と同様に、*CanUnload*プロパティを含むドットおよびクラス名が続きます。 例: COMプロジェクト名が"SpecialMathPack"で、コンポーネント内のメインクラスが"clsMain"であると仮定します。ファイルのエントリは次のようになります。 `NAME=Advanced analysis routines,CLASS=SpecialMathPack.clsMain`
DESCRIPTION	このパラメーターが指定されている場合、AddInがリボンバーや可視化オブジェクトギャラリーに表示される場合の、ツールチップに使用されます。
ICON	アイコンファイルを指定します。リボンバーや可視化オブジェクトギャラリーで、AddInを示すために使用します。ファイル名にパスが含まれていない場合、またはファイルが見つからない場合、まずcatmanのインストールフォルダ内の/RES/ICONSフォルダを検索します。見つからない場合catmanは、デフォルトの画像フォルダ内(OPTIONS/FOLDERSを参照)を検索します。
AUTORUN	このパラメータが存在し、値が1(すなわちAUTORUN=1)の場合、catmanは起動時にAddInを起動します。
MULTIINSTANCE	AddInが同じオブジェクト(例えばフォーム)の複数インスタンスを作成する場合、このパラメータを1に設定(MULTIINSTANCE=1)します。AddInの複数インスタンスを参照してください。

<table>
<tr><td>FLAGS</td><td>AddInの動作を制御するビットマスク。ビットマスクに対応する整数(例えばFLAGS=129: 可視化ギャラリーでAddInを表示し、catmanの可視化パネルのウィンドウサイズに常に合わせる)、または+記号で区切った個々の値をリストします(FLAGS = 32 + 64: AddInのチャネル構成リボンタブを表示し、大きいアイコンにキャプションを下に表示)。
<table>
<tr><td>ビット/マスク</td><td>効果</td></tr>
<tr><td>0 / 1</td><td>SHOW_IN_VISUAL_GALLERY
このフラグを設定した場合、catmanは可視化オブジェクトギャラリーでアイコン(およびキャプション)を表示します。このタイプのAddInは、catmanの可視化システムの機能拡張として特殊な表示オブジェクトを提供します。これには通常、MULTIINSTANCEフラグが設定され、各インスタンスのウィンドウを提供します。</td></tr>
<tr><td>1 / 2</td><td>SHOW_IN_EXPORT_FORMATS
AddInがSHOW_IN_EXPORT_FORMATSフラグを設定している場合、解析モードのExportリボンタブのexport formatリストにAddIn(その名前)のエントリを追加します。</td></tr>
<tr><td>2 / 4</td><td>SHOW_IN_FILE_CONVERTERS
AddInがSHOW_IN_FILE_CONVERTERSフラグを設定している場合、解析モードのTest Explorerリボンタブの File converterリストにAddInを追加します。ファイルコンバータは、あらゆる形式をcatmanの標準形式に変換する必要があります。</td></tr>
<tr><td>3 / 8</td><td>SHOW_IN_ANALYSIS_RIBBON_TAB
解析モードのComputationsリボンタブにAddIn(ICONパラメータが設定されている場合はアイコン)のコントロールを追加します。ユーザがコントロールをクリックした場合、AddIn のRunメソッドが呼び出されます。</td></tr>
<tr><td>4 / 16</td><td>SHOW_IN_NEW_PANEL_DROPDOWN_MENU
VisualizationリボンタブのCreate new panel dropdownメニューにAddIn(ICONパラメータが設定されている場合はアイコン)のコントロールを追加します。ユーザがコントロールをクリックした場合、AddInのRunメソッドが呼び出されます。</td></tr>
<tr><td>5 / 32</td><td>DAQモードのChannel ConfigurationリボンタブにAddIn(ICONパラメータが設定されている場合はアイコン)のコントロールを追加します。ユーザがコントロールをクリックした場合、AddInのRunメソッドが呼び出されます。</td></tr>
<tr><td>6 / 64</td><td>LARGE_ICON_AND_CAPTION_BELOW</td></tr>
<tr><td>7 / 128</td><td>FIT_WINDOW_TO_PANEL</td></tr>
</table>
</td></tr>
</table>

AddInの複数インスタンス

ADD_INS.CATファイル内のMULTIINSTANCEプロパティを設定する場合(アドインの登録および起動を参照)、catmanはAddInが複数回の起動が可能であることを想定します。複数のインスタンスをサポートするAddInは、catmanユーザが何度も使用することのできる、典型的な可視化オブジェクトとして使用されます。ほとんどの場合、SHOW_IN_VISUAL_GALLERY、またはSHOW_IN_NEW_PANEL_DROPDOWN_MENUフラグを設定します。

catmanは、実際にAddInクラスの複数インスタンスを作成しないことに注目してください - このクラスは1回だけインスタンス化され、Runメソッドも一度だけ呼び出されます。catmanは新しいインスタンスを要求場合、AddInクラスに三つの追加メソッドをサポートする必要があります: *NextInstance*、*RemoveInstance*、*Resize*および*SetProperty*。

複数インスタンスAddInの場合、*Run*メソッドを使用して最初のインスタンスの作成しないでください - catmanが*NextInstance*を呼び出すまで待つことをお勧めします。

```
Public Function NextInstance() As Integer
```

catmanは、ユーザがAddInからオブジェクトを作成したいときは常にこの関数を呼び出します。例えば、コントロールをクリックしてcatmanの可視化オブジェクトギャラリーにAddInを追加します。その後、インスタンス化可能なオブジェクトとみなされるものを作成するかどうかはアドイン次第です。ほぼすべてのケースで、これは何かを表示する(すなわちウィンドウ)フォームになります。catmanのデフォルト(作業)ディレクトリ(通常はドキュメント/HBM/catmanEasy)のフォルダ/EXAMPLES/AddInsで新規フォームを作成し、NextInstanceが呼び出される度にコレクションに追加する方法を示すCustomVisualObjectテンプレートがあります。

メソッドは、新規インスタンスを識別する一意のID(16bit整数)を返す必要があります。catmanは以降の*RemoveInstance*および*SetProperty*の呼び出しに、このIDを使用します。

```
Public Sub RemoveInstance(ByVal InstanceID As Integer)
```

catmanがこのメソッドを呼び出すと、*InstanceID*で識別したインスタンスを削除することができます(例えば、フォームコレクションからフォームを削除)。catmanは今のところ、あるケースでのみこのメソッドを呼び出します: パネルの削除および、ユーザがパネル上に視覚化のAddInオブジェクトを作成した場合。

```
Public Sub Resize(ByVal InstanceID As Integer, ByVal Left As Long, ByVal Top As
Long, ByVal Width As Long, ByVal Height As Long)
```

catmanが視覚化AddInインスタンスのサイズ変更を行いた場合に呼び出します。FIT_WINDOW_TO_PANELフラグが設定された場合および、ユーザがcatmanのパネルサイズを変更した場合に発生します。すべての座標は*Twips*単位です。

```
Public Sub SetProperty(ByVal InstanceID As Integer, ByVal PropertyID As Integer,
Value As Variant)
```

catmanはこのメソッドにより、インスタンスのいくつかのプロパティを制御することができます。現在はインスタンスの可視性を制御するPropertyID(ID = 0)のみをサポートしています。

ID	説明
0	インスタンスの可視(Value = 1)または不可視(Value = 0)を示します。ユーザの可視化パネル切り替え時に、catmanはこのプロパティを設定します。

ファイルエクスポートおよびファイル変換AddIn

AddInがSHOW_IN_EXPORT_FORMATSフラグを設定している場合、catmanは解析モードのExportリボンタブの利用可能なエクスポート形式のリストに、AddIn(その名前)のエントリを追加します。ユーザが追加したエクスポート形式を選択した場合、catmanはAddInの*RunExportFilter*メソッドを呼び出します。

```
Public Function RunExportFilter(ByVal FileName As String, ByVal ExportComment As
String) As Long
```

この関数は*EA_Test.GetExportListCount*および*EA_Test.GetExportListItem*より、現在選択しているエクスポートするチャネルにアクセスします。

```
Public Sub EA_Test.GetExportListCount(Count As Integer)
```

Count (16 bit整数)は、ユーザが追加したエクスポートリストに追加しているチャネルと演算の数を提供します。

```
Public Sub GetExportListItem(ByVal Index As Integer, CName As String, TestIndex As
Integer)
```

エクスポートリストに含まれているチャネルと演算のリストを反復処理するために、このメソッドを使用します。*TestIndex=0*の場合、項目は演算です。項目名を取得した場合、項目のデータを取得するためEA_Test.GetChannelDataを呼び出します。

AddInはSHOW_IN_FILE_CONVERTERSフラグを設定している場合、catmanはTestExplorerファイルリストのコンテキストメニューにAddIn(その名前)のエントリを追加します。ユーザが追加したコンバータを選択した場合、catmanはAddInの*RunFileConverter*メソッドを呼び出します。

```
Public Function RunFileConverter(ByVal FileList As String) As Long
```

*FileList*にはユーザがTest Explorerのリストで現在選択しているファイルの";"で区切られたリストが含まれます。ファイル名にはフルパスが含まれます。

C/C++プログラマのための注意事項

このセクションでは、C/C++から配列を返すインタフェース関数について説明します。

catmanEasy/APのActiveXインタフェースのいくつかのメソッドは、VARIANTまたはDOUBLE配列の形式でデータを返します。その中でも、読み込んだテストからチャネルデータを取得するインスタンス(*EA_Test.GetChannelData*)は重要なメソッドです。配列形式で渡すメソッドは、Visual BasicやBorland Delphiでは容易に処理できますが、C/C++、Matlab、HPVeeのようなCベース言語での処理は、やや困難なことがわかります。

これらの言語ではまずメソッドを呼び出して渡すことができるよう、"SAFEARRAY"を構築します。直接DOUBLE配列へのポインタを渡すことはできません! 以下のC/C++コードのスニペットは、読み込んだテストのチャネルからデータを取得するメカニズムを示しています:

```
long FetchData(short TestNo, short ChanNo, double *MVal)
{
VARIANT Count;
// How many samples in channel
EA_Test.GetChannelAttribute(TestNo, ChanNo, 1, Count);
// Creates a one-dimensional DOUBLE array whose lower bound is always zero.
SAFEARRAY* tMVBuf = SafeArrayCreateVector( VT_R8, 0, (int)Count );
// Prevent an error if no data is allocated
if ( tMVBuf == NULL )
  {
  AfxMessageBox( "No safe array today", MB_OK );
  return -1;
  }

// Fetches the datal.
EA_Test->GetChannelData(TestNo, ChanNo, StartPtr, &tMVBuf);
// Copy it into your pointer of doubles
memcpy( MVal, tMVBuf->pvData, Count*sizeof(double) );

// Destroys an existing array descriptor and all of the data in the array.
SafeArrayDestroy( tMVBuf );
```

ActiveX ServerとしてcatmanEasy/APを使用

通常、EasyScriptプロジェクトはcatmanEasy/AP内で実行され、表示されるユーザインタフェースはcatmanEasy/APのものです。

ここでは、VB、Excel VBA、C/C++ Delphiまたは他のCOM対応言語で作成した独自のアプリケーションでEasyScriptオブジェクトモデルを利用する方法について説明します。このモードではcatmanEasy/APはバックグラウンドで実行しますが、表示するユーザインタフェースはアプリケーションのものです。これを行うにはcatmanEasy/APは、"Application"を提供します。

ℹ 次の例は、Microsoft Visual Basicを前提としていますが、これは容易に他の言語に変換することができます。

Step 1: *Application*オブジェクトの作成:

```
Dim catApp As Object
Set catApp = CreateObject("catmanEasy.Application")
```

このようなオブジェクトの作成は"遅延バインディング"と呼ばれています。このようなオブジェクトを作成する場合、多くのIDEが提供するインテリセンス機能は使用できません。

その代わりとして、catmanEasy/APのApplicationオブジェクトへの参照を追加することができます(メニューの"Project/References..."からVB IDE)。これは"事前バインディング"と呼ばれています。この場合、インテリセンスが利用可能です。宣言にCreateObjectを使用する必要はありません:

```
Dim catApp As New catmanEasy.Application
```

Step 2: これ以降、EasyScriptオブジェクトモデルによって公開された任意のメソッドが使用できます。ただし、ほとんどの場合、最初に行うのはDAQプロジェクト(catmanEasy/APのインタラクティブセッション中に以前作成した)の読み込みになります。その後のすべてのアクション(例えばDAQジョブの開始)は、このDAQプロジェクトで動作します。

```
RetVal = catApp.OpenProject("MyProject.MEP")
RetVal = catApp.Job.Start 1
```

ℹ Applicationオブジェクトによって公開されたサブオブジェクトは、EasyScriptサブオブジェクトに対応しますが、接頭辞EA_は付きません。。

EasyScriptでの記述	アプリケーションでの記述
EA_DAQ (例えばEA_DAQ.CurrentReading)	DAQ (例えばcatApp.DAQ.CurrentReading)
EA_IO (例えばEA_IO.Measure)	IO (例えばcatApp.IO.Measure)
EA_Job (例えばEA_Job.Start)	Job (例えばcatApp.Job.Start)
EA_Test 例えばEA_Test.Load)	Test (例えばcatApp.Test.Load)

以下の例は、テストを読み込んでデータを解析する方法を示しています:

```
Dim ChanSize As Variant
Dim i As Long, RetVal As Long
catApp.OperationMode = 1 'Switch to analysis mode.
RetVal = catApp.Test.Load("C:\TESTDATA\TEST_3.TST")
RetVal = catApp.Test.ChannelAttribute(1, "Pressure_A2", CHAN_SRCLEN, ChanSize)
ReDim DataBuf(ChanSize) As Double
RetVal = cat.Test.GetChannelData(1, "Pressure_A2", DataBuf)
For i=0 To ChanSize-1
  If DataBuf(i) > 200 Then
  ..... 'Computations, etc.
  End If
Next
```

注意: オフライン解析プロジェクトを読み込むことはできません。

カスタムデータのグラフ表示

グラフ(Post-processやReal-timeにかかわらず)のプロット(曲線)データは、常にチャネルから取得します。catmanEasy/APは、自動的にプロット(曲線)に関連付けられているチャネル名に応じてデータを自動的に取得します。対話的にチャネル選択リストやData Explorer(解析モード)からグラフにドラッグしてチャネルを割り当てます。EasyScriptでは、*Panel.SetRTDisplayChannel*または*EA_Test.PlotYT resp.EA_Test.PlotYX*メソッドを使用します。チャネル内のデータは常に計測データまたは演算データです。

ただし、スクリプトで実行された複雑な計算の結果やファイルから読み込まれたデータなど、独自のデータを表示したい場合もあります。次のセクションでは、さまざまなシナリオでの処理方法について説明します。

バージョン3.0.1以降、基礎となるグラフオブジェクトを完全制御し、すべてのcatmanの関数をバイパスすることができます。使用方法は"ローレベルのグラフプログラミング"の章で説明しています。

配列ベースのメソッドを使用してプロットデータを追加する場合、組み込みのcatmanグラフルーチンが干渉して、変更が破棄されないことを確認してください。グラフ制御の取得に*EA_Panel.GetDirectControl*メソッドを使用します。*EA_Panel.ReleaseDirectControl*を呼び出して制御をcatmanに戻します。

解析モードでカスタムデータを表示(概要グラフ)

方法1: データチャネル作成してプロット

EA_Test.CreateChannelメソッドにより、データを含む新規チャネルを作成します。この新規チャネルは任意の計測データチャネルとして、EA_Test.PlotYTやEA_Test.PlotYXよりプロットすることができます。新規チャネルはTest Explorerに表示され、グラフにドラッグすることができます。y(t)グラフでの等間隔の増分は、*EA_Test.CreateChannel*でdtパラメータを設定する必要があります。

方法2: データを含む配列のプロット

これは方法1のチャネル作成よりも柔軟ですが、catman組み込みのグラフの再描画でプロットが破棄されないよう、注意する必要があります。配列データをプロットするEA_Graph.PlotArrayXYメソッドを使用して、配列データをプロットします。

配列データをプロットする場合、以下を遵守することを推奨します。

- 同一グラフにデータチャネルと配列データを混在させないでください(すなわち同一グラフに*EA_Test.Plotxxx*ルーチンと*EA_Graph.PlotArrayYX*を一緒に使用しないでください)。これはまた、配列のプロットを予定しているグラフに、チャネルをドラッグしてはならないことを意味します。
- 配列のプロットは常に非圧縮で行われます。したがって配列あたりの最大点数は、非圧縮データの最大数に依存します(グラフ構成の"Advanced"タブ、または*EA_Graph.SetProperty*で設定)。

DAQモードでカスタムデータの表示(概要グラフ)

方法1: 補助チャネルを使用

DAQモードの概要グラフは、チャネルに一時格納されたデータに基づいています。カスタムデータを表示するは、このような補助チャネルが必要です。そして、このチャネルを一時格納したデータで埋めます。補助チャネルはChannel Configurationウィンドウの"Special"リボングループ、"AUX"ボタンから作成します。そして、EA_DAQ.SetTempStoreメソッドを使用してデータを一時格納データに転送します。グラフを再描画するとcatmanは、この格納したAUXチャネルからデータをフェッチします。

一時ストアが転送したいデータ数に対して十分な大きさかを確認してください。一時ストアがどのように構成されているかを確認するため、OPTIONS/DATA STORAGEを確認します。"Fixed size"(固定サイズ)に構成されている場合、"Max. number of samples per channel"を十分に確保する必要があります。固定サイズに設定していない場合、データを保持するため自動的に拡張されます。

方法2: データを含む配列のプロット

これは補助チャネル作成よりも柔軟ですが、catman組み込みのグラフの再描画でプロットが破棄されないよう、注意する必要があります。配列データをプロットするEA_Graph.PlotArrayXYメソッドを使用します。

配列データをプロットする場合、以下を遵守することを推奨します。

- 同一グラフにデータチャネルと配列データを混在させないでください(すなわち同一グラフにEA_Panel.*SetRTDisplayChannel*ルーチンと*EA_Graph.PlotArrayYX*を一緒に使用しないでください)。これはまた、配列のプロットを予定しているグラフに、チャネルをドラッグしてはならないことを意味します。
- 配列のプロットは常に非圧縮で行われます。したがって配列あたりの最大点数は、非圧縮データの最大数に依存します(グラフ構成の"Advanced"タブ、または*EA_Graph.SetProperty*で設定)。

DAQモードでカスタムデータの表示(Real-timeグラフ)

Real-timeグラフでカスタムデータを表示させるには、補助チャネルが必要です。補助チャネルはChannel Configurationウィンドウ、コンテキストメニューの**Computations/Auxiliary channel**で作成します。このチャネルは、他の演算チャネルのように表示され、グラフにドラッグすることができます。

Real-timeグラフは、各転送後に更新します。したがって、転送完了後は直ちにデータブロックにカスタムデータを書き込む必要があります。これを行うには、タスクを実行するプロシージャにAutomatic Procedure Execution(ステージ"After transfer DAQ data block")を追加する必要があります。EA_DAQ.SetCurrentBlockメソッドを使用してカスタムデータを入力します。

リアルタイムのカスタムデータの表示

この方法はDAQおよび解析の両方のモードで可能です。EA_Graph.ChartArrayXYを使用すると、グラフのリアルタイム描画を制御することができます。

配列データをプロットする場合、以下を遵守することを推奨します。

- 同一グラフにデータチャネルと配列データを混在させないでください(すなわち同一グラフにEA_Panel.*SetRTDisplayChannel*ルーチンと*EA_Graph.PlotArrayYX*を一緒に使用しないでください)。これはまた、配列のプロットを予定しているグラフに、チャネルをドラッグしてはならないことを意味します。
- 配列のプロットは常に非圧縮で行われます。したがって配列あたりの最大点数は、非圧縮データの最大数に依存します(グラフ構成の"Advanced"タブ、または*EA_Graph.SetProperty*で設定)。

ローレベルのグラフプログラミング

この章ではcatmanグラフオブジェクトのすべての内部関数にアクセスする方法について説明します。ローレベルのグラフプログラミングでは、*EasyScript*の*EA_Graph*メソッドやプロパティで、公開されていないグラフの、すべての組み込み関数にアクセスすることができます。スクリプトで作成したデータのプロットや、catmanからアクセスできないプロパティの変更(グラフの背景画像など)を行う場合、この関数は非常に役立ちます。

⚠ これらのオブジェクトを使用した作業は、VBおよびcatmanの高度な知識を持つ上級プログラマおよび、システムインテグレータが行うべきです。HBM catmanプログラマのように、実際にオブジェクトを操作することができます。間違ったプロパティの変更やメソッドの呼び出しは、catmanがクラッシュする恐れがあります。本書では、公開されているグラフオブジェクトの、すべてのプロパティおよびメソッドについて説明していません。ただし、多くのプロパティおよびメソッドは説明不要(インテリセンスにより公開)です。

詳細はHBMにお問い合わせください。

⚠ 自身でグラフプログラムを作成する場合、組み込みのcatmanグラフルーチンが干渉して、変更が破棄されないことを確認してください。グラフ制御の取得に*EA_Panel.GetDirectControl*メソッドを使用します。*EA_Panel.ReleaseDirectControl*を呼び出して制御をcatmanに戻します。

最初のステップでは、以降のグラフオブジェクトの参照を保持する変数を宣言する必要があります。

```
Dim oGraph As Pesgo
```

二番目ステップでは、アクセスするグラフをこの変数に割り当てます。これには、*EA.GetObject*メソッドを使用します:

```
Set oGraph = EA.GetObject(0, "GRAPH_1")
```

"GRAPH_1"は、グラフに付けた名前の例です。インテリセンスシステムは、グラフオブジェクトによって公開されている、すべての関数がリスト表示されます。*oGraph*の後にドットを入力すると、すべてのメソッド/プロパティのリストがポップアップ表示されます。それらの多くは一目瞭然で、プロパティの後に"="を入力すると、インテリセンスによって可能な設定オプションを表示します。

次のステップで、catmanの組み込み描画ルーチンがプログラムに干渉するのを防ぎます:

```
EA_Panel.GetDirectControl("GRAPH_1")
```

最後に、グラフに適用されたかもしれないcatmanのすべての設定を元に戻します。

```
oGraph.PEActions=REVERT_TO_DEFAULTS
```

デフォルトにリセットすることは必須ではありませんが、これを行わないと、デフォルトの単精度データではなく倍精度データを取得しようとします。"倍精度データの追加"の項を参照してください:

以下は、データをプロットする方法の簡単な例を示しています:

```
  Dim oGraph As pesgo
  Dim i As Integer

  EA_Panel.GetDirectControl("GRAPH_1")
  Set oGraph = EA.GetObject(0, "GRAPH_1")
  oGraph.PEActions = REVERT_TO_DEFAULTS 'Reset to defaults - important, so anything
already configured by catman is reverted and you start with a clean graph

  oGraph.MainTitle = "Test"
  oGraph.Subets = 1              'The number of curves
  oGraph.Points = 100            'Number of points in every curve
  For i=0 To 99
    oGraph.XData(0, i) = i'Populate data, first index is curve index, second is point
index
    oGraph.YData(0, i) = Sqr(i)
  Next

  oGraph.PEActions = REINITIALIZE_RESETIMAGE 'Redraw the graph
```

次のセクションでは、最も一般的なプロパティおよびメソッドのいくつかを紹介します:

データの追加(シンプル)

最初に、*.Subsets*プロパティでカーブ数を指定します。次に*.Points*で各カーブの点数(すべてのカーブが同じである必要があります!)を指定します。カーブが異なる長さの場合、どちらかを*.NullDataValueX*または*.NullDataValueY*により、空値で埋めます。そして、後半のデータ配列を*.XData(CurveIndex,PointIndex)*および*.YData(CurveIndex,PointIndex)*でポップアップします。

```
  g.Subsets = 5
  g.Points = 100 'Per subset!

  g.XData(CurveIndex, PointIndex) = ...
  g.YData(CurveIndex, PointIndex) = ...
```

グラフのxデータとyデータは常に同数であることを確認します - 複数のカーブは同じxのデータセットを必ず共有するわけではありません!NullValuesのデフォルトはゼロですが、これは次のように変更できます。

```
  g.NullDataValueX = ...
  g.NullDataValueY = ...
```

座標の何れかがNull値のデータポイントは描画しません。

カーブの色、ラベルおよび線の種類を設定

```
g.SubsetColors(CurveIndex) = ....
g.SubsetLabels(CurveIndex) = ....
g.SubsetLineTypes(CurveIndex) = ...
```

線の種類:

0 = 細線
1 = 破線
2 = 点線
3 = 鎖線
4 = 二点鎖線
5 = 実線
6 = 太線
6以上も可能です。

データの追加(高速)

2次元のXDataおよびYData配列を使用する代わりに、グラフのDLLにあるAPI関数を使用することができます。この方法は、数千のデータをプロットする場合、高速に動作します。

```
  Declare Function PEvset Lib "PEGRP32E.DLL" Alias "PEvsetA"(ByVal hObject&, ByVal
nProperty&, lpvData As Any, ByVal nItems&) As Long

  Const PEP_faXDATA = 2135
  Const PEP_faYDATA = 2140

  Dim g As Pesgo
  Dim i As Integer
  Dim ret As Long
  Dim x(100) As Single
  Dim y(100) As Single

  Set g = EA.GetObject(0, "GRAPH_1")
  For i=0 To 99
    x(i) = i*0.1
    y(i) = Rnd(i)
  Next

  ret = PEvset(g, PEP_faXDATA, x(0), 100)
  ret = PEvset(g, PEP_faYDATA, y(0), 100)
  g.PEactions = RESET_IMAGE
```

グラフに複数のカーブが含まれている場合、二番目のカーブデータは、最初のカーブデータ後に追加されます(すなわちy(100)..y(199), x(100)...x(199))。グラフのxデータとyデータは常に同数であることを確認します - 複数のカーブは同じxのデータセットを必ず共有するわけではありません。!

倍精度データの追加

グラフに倍精度データセットを使用する場合:

```
  g.UsingXDataII = True
  g.UsingYDataII = True
```

データの取り込み:

```
  g.XDataII(CurveIndex, PointIndex)
  g.YDataII(CurveIndex, PointIndex)
```

またはPEvsetから取り込み:

```
  Const PEP_faXDATAII = 3013
  Const PEP_faYDATAII = 3014

  ret = PEvset(g, PEP_faXDATAII, x(0), 100)
  ret = PEvset(g, PEP_faYDATAII, y(0), 100)
```

リアルタイムストリップチャートを実現

```
  Declare Function PEvset Lib "PEGRP32E.DLL" Alias "PEvsetA"(ByVal hObject&, ByVal
nProperty&, lpvData As Any, ByVal nItems&) As Long

  Const PEP_faXDATA = 2135
  Const PEP_faYDATA = 2140
  Const PEP_faAPPENDXDATA = 3658
  Const PEP_faAPPENDYDATA = 3276

  Dim g As Pesgo
  Dim ret As Long
  Dim x(100) As Single
  Dim y(100) As Single

  Set g = EA.GetObject(0, "GRAPH_1")

  'First preset data buffer with zeroes - this is important! You may not append to
"empty" data buffer!
  ret = PEvset(g, PEP_faXDATA, x(0), 100)
  ret = PEvset(g, PEP_faYDATA, y(0), 100)

  'Append new data point to the right and discard old data point at left
  ....
  ....
  x = ...
  y = ...
  ret = PEvset(g, PEP_faAPPENDXDATA, x, 1)
  ret = PEvset(g, PEP_faAPPENDYDATA, y, 1)

  g.PEactions = RESET_IMAGE
  …
```

y軸のスタック表示

.MultiAxesSubsets()コントロールは、軸レイヤの数を示します。グラフのカーブの合計数(サブセット)は、*MultiAxesSubsets*のインデックス(=レイヤ)で分散されます。次の例では、最初のレイヤ1で二つのカーブ、次のレイヤ2で二つのカーブ、およびレイヤ3で一つのカーブをプロットします。

```
g.Subsets = 5
g.MultiAxesSubsets(0) = 2
g.MultiAxesSubsets(1) = 2
g.MultiAxesSubsets(2) = 1
```

レイヤの最大枚数は16で、すなわちMultiAxesSubsetsで許可された指標は0...15です。

軸レイヤのプロパティ設定

以降のメソッドの構成(例えばYAxisLabel)のレイヤを設定する前に、*WorkingAxis*プロパティを使用します:

```
g.WorkingAxis = 0 'First layer
g.WorkingAxis = 1 'Second layer
```

軸レイヤのオーバーレイ

一つ以上のy軸レイヤを重ねる(この場合、軸はグラフの左側にスタックされます)ために、*OverlapMultiAxes*()配列を使用します。*OverlapMultiAxes*は、垂直方向に積み重ねられた軸の数を一つ以上の重複するセクションにグループ化します。次の例は、最初の二つのレイヤをオーバーレイし、三番目のレイヤを垂直方向の下に保持します:

```
g.OverlapMultiAxes(0) = 2 'The first two layers (top and middle) will overlap and
'form a single layer
g.OverlapMultiAxes(1) = 1 'The 3rd layer remain separated
```

カーブのプロットスタイルを設定

```
g.PlottingMethods(CurveIndex) = .....
```

右の軸を作成しデータをプロット

```
g.ForceRightYAxis = True 'Set .WorkingAxis first if you have more than one layer
```

次に"R"のが付いたすべてのプロパティを使用して軸を構成します(例えば、*ManualScaleControlRY*、*ManualMinRY*、*ManualMaxRY*、*RYAxisLabel*など)。*PlottingMethods*でカーブに1000を追加し、プロットのスケールを右側に表示します。

```
g.PlottingMethods(2) = 0 + 1000 '0 = Line
```

組み込みのダイアログの一つを起動(カスタム、印刷など)

```
g.PEActions = ......
```

注釈を追加

```
g.ShowAnnotations = True
```

シンボル/テキストを追加します:

```
n = n + 1 'Start with first annotation = 0
g.GraphAnnotationType(n) = Type
```

注釈の種類:

0 = テキストのみ
37 = アンカー付きテキスト
15 = テキスト+丸
16 = テキスト+塗りつぶし丸
17 = テキスト+四角
18 = テキスト+塗りつぶし四角
19 = テキスト+菱形
20 = テキスト+塗りつぶし菱形
21 = テキスト+三角
22 = テキスト+塗りつぶし三角
92 = テキスト+矢印N
94 = テキスト+矢印E
96 = テキスト+矢印S
98 = テキスト+矢印W
93 = テキスト+矢印NE
97 = テキスト+矢印SW
99 = テキスト+矢印NW
95 = テキスト+矢印

```
g.GraphAnnotationText(n) = "My annotation"
g.GraphAnnotationX(n) = .... 'Position X
g.GraphAnnotaionY(n) = ..... 'Position Y
g.GraphAnnotationColor(n) = ....
g.GraphAnnotationAxis(n) = ..... '(0,1,2... only if you have more than one 'axis
layer)
```

すべての注釈のグローバル テキストとシンボル サイズを設定します:

```
g.GraphAnnotationTextSize = ..... '(20....100)
g.GraphAnnotationSizeCntl = .... '(0.1....10)
```

水平線または垂直線を追加します:

```
  h = h + 1 'First annotation must start with 0
  g.HorzLineAnnotation(h) = 155 'Position on y-axis
  g.HorzLineAnnotationType(h) = 0 'Line type
  g.HorzLineAnnotationText(h) = "|RRight Text"
  g.HorzLineAnnotationColor(h) = 255
  g.HorzLineAnnotationAxis(h) = ..... '(0,1,2... only if you have more than 'one y
axis layer)

  v = v + 1 'First annotation must start with 0
  g.VertLineAnnotation(v) = 155 'Position on y-axis
  g.VertLineAnnotation(v) = 0
  g.VertLineAnnotation(v) = "|RRight Text"
  g.VertLineAnnotation(v) = Red
```

線の種類:

0 = 細線
1 = 破線
2 = 点線
3 = 鎖線
4 = 二点鎖線
5 = 実線
6 = 太線
6以上も可能です。

ローレベルのテーブル(スプレッドシート)プログラミング

この章では、catmanのテーブルオブジェクトのすべての内部関数にアクセスする方法について説明します。ローレベルのテーブルプログラミングにより、*EA_Panel*メソッドや*EasyScript*プロパティで公開していないテーブルのすべての内部関数にアクセスすることができます。スクリプトで作成したデータを入力する場合や、catmanからアクセスできないプロパティを変更をする場合(テーブル内に複数のシートを作成するなど)、この関数は非常に使いやすいインスタンスです。

⚠ これらのオブジェクトを使用した作業は、VBおよびcatmanの高度な知識を持つ上級プログラマおよび、システムインテグレータが行うべきです。HBM catmanプログラマのように、実際にオブジェクトを操作することができます。間違ったプロパティの変更やメソッドの呼び出しは、catmanがクラッシュする恐れがあります。本書では、公開されているグラフオブジェクトの、すべてのプロパティおよびメソッドについて説明していません。ただし、多くのプロパティおよびメソッドは説明不要(インテリセンスにより公開)です。

詳細はHBMにお問い合わせください。

⚠ 自身でテーブルプログラムを作成する場合、組み込みのcatmanルーチンがが干渉して、変更が破棄されないことを確認してください。。グラフ制御の取得に*EA_Panel.GetDirectControl*メソッドを使用します。*EA_Panel.ReleaseDirectControl*を呼び出して制御をcatmanに戻します。

最初のステップでは、以降のテーブルオブジェクトの参照を保持する変数を宣言する必要があります。

```
Dim oTable As fpSpread
```

二番目に、グラフにアクセスするテーブルをこの変数に割り当てます。これには、*EA.GetObject*メソッドを使用します:

```
Set oTable = EA.GetObject(0, "TABLE_1")
```

"TABLE_1"は、グラフに付けた名前の例です。インテリセンスシステムは、テーブルオブジェクトによって公開されている、すべての関数がリスト表示されます。*oTable*の後にドットを入力すると、すべてのメソッド/プロパティのリストがポップアップ表示されます。

次のステップで、catmanの組み込み描画ルーチンがプログラムに干渉するのを防ぎます:

```
EA_Panel.GetDirectControl("TABLE_1")
```

次項より、最も一般的なプロパティおよびメソッドのいくつかを紹介します("="を入力すると、インテリセンスによって可能な設定オプションを表示します):

操作したいセルのアドレスを指定

```
oTable.Row = r
oTable.Col = c
```

セルの内容にアクセス

```
oTable.Text
oTable.Value 'If celltype is a checkbox, button or a numerical cell
```

セルの種類を設定

```
oTable.CellType = ......
```

パネルに独自の可視化コントロール(OCX)を追加

catmanAPのバージョン3.2以降、*実行時に*スクリプトで可視化コントロールを作成することができます。組み込みコントロール(例えばデジタル表示器やグラフ)は実行時に作成できませんが、他のコントロールタイプ(例えばスクロールバー)は作成することができます。

この関数を使用すると文字通り、自由に任意のコントロール(OCX)をパネルに含めることができます。典型的な例は、*treeview*です。このコントロールはcatmanの可視化オブジェクトギャラリーに含まれていませんが、階層化構造データのために非常に便利なユーザインタフェース要素です。イベント時(マウスクリックなど)に反応するコントロールを作成するには、スクリプトプロジェクトに対応するイベントハンドラ関数(例えば "MyTreeView_OnNodeClick")を作成します。

コントロールを操作する主なツールは、*EA_Controls*コレクションです。

一般的にOCXにはライセンスが必要で、すなわち、OCXコンポーネントを購入する必要があります。catmanに組み込まれているコントロールは例外です - これらのライセンスは必要ありません。これらのコントロールはすぐに使用できます。コントロールのいくつかを、この章の最後の"catmanに組み込まれているOCXコンポーネントのProgID"テーブルで説明しています。

コントロールの作成および破棄

```
EA_Controls.Add(ByVal ProgID As String, ByVal Key As Variant, ByVal Name As String)
As Long
```

パラメータ	説明
ProgID	文字列は作成したい一意のコントロールタイプを定義します(例: "MScomctlLib.TreeCtrl.2")。OCXコンポーネントのProgIDは、Microsoft OLE Viewerのツールでリストすることができます。この章ではOCX Prog IDについて、いくつかの有用な情報があります。 ほとんどのProgIDは例のようにバージョン情報が含まれている(".2")ことを確認してください。これは使用するコンピュータにインストールされたコンポーネントのバージョンにより、異なる場合があります。
Key	一意にコントロールを識別するために使用する文字列です。例えば*.Remove*メソッドや、*.Item*メンバーにアクセスする場合に使用します。
Name	コントロール名です。これはコントロールのイベントハンドラルーチンで使用します。例: ツリービューに"MyTreeView"と名前を付けた場合、catmanがNodeClickのプロパティハンドルを処理できるよう、"MyTreeView_OnNodeClick"という名前のプロシージャを作成する必要があります。

例として、ツリービューコントロールを作成します:

```
EA_Controls.Add "MScomctlLib.TreeCtrl.2","TV_1","MyTreeView"
```

コントロールの削除は次のメソッドを使用します:

```
EA_Controls.Remove(ByVal Key As Variant) As Long
```

すべてのメソッドを削除するには、.Clearメソッドを使用します:

```
EA_Controls.Clear() As Long
```

コントロールを作成する場合、それはアクティブな可視化パネルの子になります。コントロールを作成(*EA_Panel.ShowPanel*)する前に、ホストするコントロールを選択することが重要です。

コントロールの参照を取得し組み込みのプロパティおよびメソッドを呼び出す

コントロールを作成し、組み込みプロパティおよびメソッドのすべてにアクセスするために、コントロールへの参照が必要です。*EA_Controls*コレクションの*Item*メンバーを使用や、短い表記として*Item*オブジェクトの参照をオブジェクト変数にコピーすることもできます。次の例では、前のセクションで作成したツリービューコントロールを移動し、表示する方法を示しています:

```
EA_Controls.Item("TV_1").Move 1000, 1000, 2000, 6000
EA_Controls.Item("TV_1").Visible = True
```

次のコードも同様です:

```
Dim tv As Object
Set tv = EA_Controls.Item("TV_1")
tv.Move 1000, 1000, 2000, 6000
tv.Visible = True
```

構文は、これまでの説明では少数の一般的なプロパティおよびメソッドにアクセスすることができます(*Visible*、*Enabled*、*Move*、*Left*、*Top*、*Width*、*Height*)。コントロールの完全な関数セットにアクセスするには、.Objectプロパティが必要です:

```
EA_Controls.Item("TV_1").Object.LineStyle = 1
```

次のコードも同様です:

```
Dim tv As Object
Set tv = EA_Controls.Item("TV_1").Object
tv.LineStyle = 1
```

市場で入手可能な多くのOCXコンポーネントのメソッドおよびプロパティをリストすることは、本書の範囲を超えています。各ベンダーのドキュメントを参照してください。ただし、スクリプトオブジェクトにOCXの参照を追加した場合、EasyScript組み込みの*インテリセンス*のサポートが助けになります(EasyScript editorの"Object referencesダイアログ"を参照してください)。再び、例のツリービューを試します:

- EasyScript editorで"Project/Object references"をクリックします
- しばらくすると、コンピュータ上で利用可能なすべてのOCXコントロールのリストが開きます
- "Microsoft Windows Common Controls 6.0"を選択(チェック)します
- スクリプトで変数を宣言します

```
Dim tv As MSComCtlLib.TreeView
```

- そして従来通りにオブジェクトを参照します

```
Set tv = EA_Controls.Item("TV_1").Object
```

コントロールのライセンス

多くの場合、特定のタイプのコントロールを作成する前に、このコントロールタイプのライセンス情報を入力する必要があります。catmanで使用しているコントロールのみ、必要ではありません。コントロールを作成するときにライセンスが不足しているというエラーメッセージが表示される場合、ライセンスを追加してください。EA_Controlsコレクションは、このためのEA_Controls.AddLicenseメソッドを提供しています:

```
EA_Controls.AddLicense(ByVal ProgID As String, Optional ByVal LicenseKey As String)
As String
```

この場合ProgIDは、*EA_Controls.Add*メソッドで記載したものと同じ記述です。使用するコンピュータにOCXコンポーネントをインストールし、ベンダーライセンスを正しく入力した場合、ライセンス情報がすでにわかっているため、*LicenseKey*パラメータは省略することができます。
関数はライセンスキーを返します。

イベント処理

catmanはコントロールのイベントを検出した場合(例えばマウスクリック)、以下のように、スクリプトプロジェクトで構成する必要のあるイベントハンドラプロシージャを検索します:

NameOfControl_OnEventName([Param1],[Param2],.....)

例:

```
MyTreeView_OnClick
MyTreeView_OnMouseUp
MyTreeView_OnNodeClick
```

イベントが追加のパラメータを提供している場合(例えば*MouseDown*イベントの座標x、y)、引数Paramに渡されます。*MouseDown*イベントがどのように動作するかを確認します。*MouseDown*イベントは四つのパラメータを渡します:Button、shift、x座標、y座標。したがって、イベントハンドラルーチンは以下の通りです:

```
Sub MyTreeView_OnMouseDown(ByVal Button As Integer, ByVal Shift As Integer, ByVal
x As Single, ByVal y As Single)
```

この章の最後の"一般的なイベント"では、いくつかの一般的なイベントとともに、引数を示しています。

catmanに組み込まれているOCXコンポーネントのProgID(ライセンスの必要なし)

ほとんどのProgIDには、バージョン情報が含まれていることを注目してください。これは使用するコンピュータにインストールされたコンポーネントのバージョンにより、異なる場合があります。

ProgID	説明
MSComctlLib.TreeCtrl.2	Microsoft TreeViewです。コントロールのメソッドおよびプロパティの取得の説明はMSDN(Microsoft Developer Network)を参照してください。
MSComctlLib.ListViewCtrl.2	Microsoft ListView controlです。コントロールのメソッドおよびプロパティの取得の説明はMSDN(Microsoft Developer Network)を参照してください。
Shell.Explorer.2	Microsoft WebBrowser controlです。Webページを表示する強力なコントロールです(ページへのアクセスは".Navigate URL"メソッドを使用します)。
WMPlayer.OCX.7	Microsoft Windows Media Playerです。
PVADO.PVCtrl.5	スプレッドシートオブジェクト(catman内部で使用するものと同じ)です。コントロールのメソッドおよびプロパティの取得の説明はFarpoint Technologiesを参照してください。
PESGOE.PesgoeCtrl.1	み込みの標準グラフ オブジェクトです。コントロールのプロパティとメソッドの説明は、Gigasoft Inc. ProEssentials を参照してください。
CWUIControlsLib.CWGraph.1	National Instrumentsのグラフコンポーネントです。コントロールのメソッドおよびプロパティの取得の説明はNational InstrumentsのMeasurement Studioを参照してください。

一般的なイベント

catmanはコントロールのイベントを検出した場合(マウスクリックのような)、以下のように、スクリプトプロジェクトで構成する必要のあるイベントハンドラプロシージャを検索します。

NameOfControl_OnEventName([Param1],[Param2],.....)

イベント名	説明
MouseMove	Sub NameOfControl_OnMouseMove(ByVal Button As Integer, ByVal Shift As Integer, ByVal x As Single, ByVal y As Single)
MouseUp	Sub NameOfControl_OnMouseUp(ByVal Button As Integer, ByVal Shift As Integer, ByVal x As Single, ByVal y As Single)
MouseDown	Sub NameOfControl_OnMouseDown(ByVal Button As Integer, ByVal Shift As Integer, ByVal x As Single, ByVal y As Single)
KeyUp	Sub NameOfControl_OnKeyUp(ByVal KeyCode As Integer, ByVal Shift As Integer)
KeyDown	Sub NameOfControl_OnKeyDown(ByVal KeyCode As Integer, ByVal Shift As Integer)
Click	Sub NameOfControl_OnClick()
DblClick	Sub NameOfControl_OnDblClick()
DragDrop	Sub NameOfControl_OnDragDrop(Source As Object, ByVal x As Single, ByVal y As Single) ドラッグしているオブジェクトの詳細情報(主にチャネル名)は、*EA.GetDragInfo*より取得できます。
DragOver	Sub NameOfControl_OOnDragOver(Source As Object, ByVal x As Single, ByVal y As Single) ドラッグしているオブジェクトの詳細情報(主にチャネル名)は、*EA.GetDragInfo*より取得できます。
LostFocus	Sub NameOfControl_OnLostFocus()
GotFocus	Sub NameOfControl_OnGotFocus()

SystemLogオブジェクトの使用

*SystemLog*オブジェクトを使用すると、catmanの内部システムログデータベースにアクセスすることができます。このデータベースはMS Accessのデータベースです。システムログは通常、ほとんどの場合catmanで管理しており、エントリの追加や取得をするだけです。

最初に、システムログオブジェクトの参照を取得する必要があります:

```
Const SYSLOG = 14
Dim obLog As Object
Set obLog = EA.GetObject(OB_SYSLOG, "")
```

次のセクションより、最も有用なプロパティおよびメソッドを示します:

1) 新規のログエントリを追加

```
Public Sub AddLog(LogText As String, Level As Integer, Optional Comment As String,
Optional AuxFile As String)
```

引数	説明
LogText	ログメッセージテキスト。
Level	0 = 情報、1 = 警告、2 = エラー。
Comment	追加するコメントテキスト。
AuxFile	ファイルが指定されている場合、Log Viewer機能の"Show descriptor file"を表示します。ビューアはプレーンテキストファイル、画像、HTML、MS Wordドキュメントを表示できます。

2) ログエントリの取得:

```
Public Sub GetLog(ByVal Index As Long, LogText As String, Level As Integer, Comment
As String, AuxFile As String, DateTimeCreated As Double)
```

引数	説明
Index	取得するログエントリのインデックス。ログ内の最新のインデックスを取得するため、*LastIndex*プロパティを使用します。これを*NumEntries*プロパティと混同しないでください - これはデータベース内のインデックスを正しく反映しません!
LogText	ログメッセージテキスト。
Level	0 = 情報、1 = 警告、2 = エラー。
Comment	追加するコメントテキスト。
AuxFile	追加の記述ファイル。
DateTimeCreated	OLEの時間形式(1900年1月1日からの秒)。

3) 現在のログ内のエントリ数および、最後(最新)エントリのインデックスを決定します。インデックス値は通常、*NumEntries*と同じではありません(例えば、インデックス範囲は5000から5099でも*NumEntries*は100の可能性があります - これは追加/削除操作に依存します)。

```
Public Property NumEntries
Public Property LastIndex
```

4) ログエントリの削除:

```
Public Sub DeleteLog(ByVal Index As Long)
```

引数	説明
Index	取得するログエントリのインデックス。ログ内の最新のインデックスを取得するため、*LastIndex*プロパティを使用します。これを*NumEntries*プロパティと混同しないでください - これはデータベース内のインデックスを正しく反映しません!

5) Log Viewerの起動:

```
Public Sub obLog.ShowViewerEx(ByVal Mode As Integer)
```

引数	説明
Mode	0 = モーダル、1 = モードレス

キャンバスオブジェクトの操作

catman可視化オブジェクトギャラリーは、オブジェクト タイプ"Canvas"を公開します。これを使用して独自のコンテンツを描画できます。パネルの背景に直接描画するのとは対照的に、キャンバスオブジェクトに描画すると、ユーザは所有者が描画したオブジェクトの周囲を移動できます。

参照: EA_Panel.CanvasSetText

キャンバスオブジェクトにアクセスするには、まずキャンバスのホスト コンテナオブジェクトを取得します。このオブジェクトは画像ボックスではありません - 次に、ホストの描画キャンバスにアクセスする必要があります。キャンバスオブジェクトはVB標準のピクチャボックスであり、このコントロールタイプでサポートされているすべてのメソッドとプロパティを適用できます。ただし、1つの例外があります。VBピクチャボックスメソッド*Print*(テキスト出力) は使用できません。テキスト出力を実現するには、ホストコンテナオブジェクトの*DrawText(Text As String)*メソッドを使用します。また、キャンバスピクチャボックスの.Lineメソッドと.Circleメソッドを使用する代わりに、ホスト コンテナの組み込みメソッド*DrawLine*と*DrawCircle*を使用して線と円を描画することをお勧めします。

以下のコードスニペットは、このメカニズムを示しています:

```
Dim oContainer As Object
Dim oCanvas As PictureBox
Dim i As Integer
Dim x As Long
Dim y As Long
Dim r as Long
Dim xcenter as Long
Dim ycenter as Long

'Get the host container
Set oContainer=EA.GetObject OB_VISUALOBJECT_BY_NAME,"CANVAS_1"
'Get the canvas picture box
Set oCanvas=oContainer.GetChildObject(0)

oCanvas.AutoRedraw=True
oCanvas.BackColor=vbRed
oCanvas.ForeColor=vbWhite
oCanvas.FontSize=12
oCanvas.CurrentX=50
oCanvas.CurrentY=100
oContainer.DrawText "Hello world"
oContainer.DrawLine 0,0,300,300,vbRed
xcenter=100
ycenter=100
r=50
oContainer.DrawCircle xcenter, ycenter, r, vbRed

For i=0 To 100
  x=i
  y=i
  oCanvas.PSet x,y,vbWhite
Next

oCanvas.Picture=LoadPicture("C:\MyPic.jpg")
```

注記

- 描画の座標系はピクセル単位です!

QuantumXモジュールと直接通信

EA_IO.QuantumX二つのメソッドを使用します。オブジェクトのWriteXMLおよびReadXMLは、QuantumXと直接データを交換することができます。データ交換は、XMLデータブロック形式で行います。

⚠ QuantumXシステムのXMLデータ構造の内容は本書に含まれていません。catmanEasyインストールフォルダのScriptWebHelp\Referencesサブフォルダにある、XMLおよびXSDファイルを参照してください。

すべてのXMLデータブロックは、XPathで指定した場所から読み込み/書き込みが行われます。XPathは特定のコネクタやチャネルを指す場合があります。

XPathの設定

XPathはQXWriteおよびQXReadメソッドのXMLSectionのパラメータで指定することができます。パス表記は非常に長いため、QXWriteのコマンド文字列で事前にXPathを選択する方が簡単です。以降のXMLデータ転送は変更されない限り、このパスを参照します。以下の例は、DAQプロジェクトで最初のモジュールにXPathを設定する方法を示しています:

```
'Select 2nd connector of the module
EA_IO.QuantumX.WriteXML 1, "", "SELECT_CONNECTOR 2"
```

コネクタがCAN-Busモードで動作している場合、追加コマンドによりコネクタ1のサブチャネルを選択できます:

```
EA_IO.QuantumX.WriteXML 1, "", "SELECT_CHANNEL 28"
```

いくつかのモジュールタイプは、チャネル毎に複数の信号を持っています。すべての信号フィルタおよびサンプルレートを個別に設定することができます。

```
EA_IO.QuantumX.WriteXML 1, "", "SELECT_SIGNAL 1"
EA_IO.QuantumX.WriteXML 1, "", "<SyncSignal><OutputRate>1200</OutputRate></
SyncSignal>"
```

Views

QuantumXモジュールは内部に、"Views"と呼ばれる複数の異なるパラメータグループを持っています(例えばView 5は、チャネルに関するすべてのパラメータが含まれています: センサの設定、フィルタなど)。以降の*QXWrite/QXRead*の呼び出しでアドレス指定されるパラメータグループは、次のコマンドによって設定されます:

```
EA_IO.QuantumX.WriteXML 1, "", "SELECT_VIEW 5"
EA_IO.QuantumX.WriteXML 1, "", "SELECT_CONNECTOR 2"
EA_IO.QuantumX.WriteXML 1, "", "SELECT_SIGNAL 1"
EA_IO.QuantumX.WriteXML 1, "", "<SyncSignal><OutputRate>1200</OutputRate></
SyncSignal>"
```

利用可能なViews	説明
1	通信設定(例えばEthernetアドレス)
3	システム設定(例えばモジュール名やタイプ)
5	チャネルパラメータ(例えばセンサ設定)
6	アナログ信号の信号パラメータ(サンプルレート、フィルタ)
7	CAN-bus信号の信号パラメータ(サンプルレート、フィルタ)

先のセクションで説明したXPathを選択するショートカットメソッドは、View 5、6または7でのみ(チャネルまたは信号パラメータ)可能です。他のすべてのViewsは、XMLSectionパラメータで、明示的にXPathを指定する必要があります。

```
strBuf = Space$(256)
EA_IO.QuantumX.WriteXML 1, "", "SELECT_VIEW 3"
EA_IO.QuantumX.ReadXML 1, "System/ModuleName", strBuf, 255
```

データの書き込み(QXWrite)

ターゲットにXPathを指定したら、パラメータ化情報が含まれているパラメータデータ内のXMLデータブロックを渡します:

```
EA_IO.QuantumX.WriteXML 1, "", "<ChannelName>Pressure_1</ChannelName>"
```

または:

```
EA_IO.QuantumX.WriteXML 1, "", "<Sensor><Sensortype><Bridge><ExciteAmpl>1</
ExciteAmpl></Bridge></Sensortype></Sensor>"
```

データの読み込み(QXRead)

ターゲットにXPathを指定(コネクタ、チャネルや信号を指します)したら、XMLSectionパラメータでパスの残りの部分(例えばセンサのプロパティ)を渡します:

```
strBuf = Space$(256)
EA_IO.QuantumX.WriteXML 1, "", "SELECT_VIEW 5"
EA_IO.QuantumX.WriteXML 1, "", "SELECT_CONNECTOR 2"
EA_IO.QuantumX.ReadXML 1, "/ChannelName", strBuf, 255
EA_IO.QuantumX.WriteXML 1, "", "SELECT_SIGNAL 1"
EA_IO.QuantumX.ReadXML 1, "/SyncSignal/SignalName", strBuf, 255
```

または:

```
strBuf = Space$(256)
EA_IO.QuantumX.WriteXML 1, "", "SELECT_VIEW 5"
EA_IO.QuantumX.WriteXML 1, "", "SELECT_CONNECTOR 2"
EA_IO.QuantumX.ReadXML 1, "/Sensor/Sensortype/Bridge/ExciteAmpl", strBuf, 255
EA_IO.QuantumX.ReadXML 1, "/Sensor/Identification/SensorName", strBuf, 255
```

注記

⚠ MGCplusやSpide8に慣れているユーザ: これらデバイスでサポートされている3文字のコマンドインタフェースは、QuantumXで使用できません。

⚠ XMLSectionパラメータのすべてのXMLタグは、大文字と小文字を区別します。

EasyScriptリファレンス

この章には、EasyScriptのすべてのオブジェクト、プロパティおよびメソッドへのリファレンスが含まれています。プロパティでマークした参照内のすべての項目は、変数として使用することができます。すなわち、ステートメント内でそれらを使用することができます(`v = Property + 5`)。他のすべての項目は関数やメソッドです。データ型"`As Long`"は、戻り値を変数に代入できる存在であることを示します:

```
RetValue = Method1(Arg1, Arg2, ...)
```

しかし、変数を戻り値に代入することはオプションによります。また戻り値を評価せずに、直接メソッドや関数を呼び出すこともでき、ます。この場合、括弧を省略するすることがあります:

```
Method1 Arg1, Arg2, ...
```

また、暗黙的に戻り値を使用できますが、この場合は括弧を使用する必要があります:

```
If Method1(Arg1, Arg2, ...) < 0 Then
```

関数	説明
EA	このオブジェクトにより、例えばエラーステータスのクエリ、DAQモードと解析モード間の動作モードの選択などが可能になります。Code Builderの"General"グループに、このオブジェクトの関数があります。
EA_IO	デバイスと通信するためのメソッドを提供します。このオブジェクトを使用すると、デバイス内の任意の設定を行ったり、読読み取りができます。Code Builderの"I/O Services"グループに、このオブジェクトの関数があります。
EA_Job	このオブジェクトを使用すると、DAQジョブの管理、設定(サンプルレート、チャネルの有効化、トリガ、ストレージモードなど)、開始や停止が行えます。Code Builderの"DAQ jobs"グループに、このオブジェクトの関数があります。
EA_DAQ	このオブジェクトを使用して、計測中に現在の計測値にアクセスします。Code Builderの"DAQ"グループに、このオブジェクトの関数があります。
EA_Test	解析プロジェクトへのアクセスを提供します。このオブジェクトのメソッドは、テストの読み込み/削除、データのグラフ表示、テストパラメータとチャネルプロパティの読み取り、計算の実行およびデータのエクスポートを行います。Code Builderの"Test data" (解析モード)グループに、このオブジェクトの関数があります。
EA_Panel	このオブジェクトを使用して、(例えばパネルのTextboxから)ユーザエントリのクエリ、パネルに情報を出力、表示オブジェクトやグラフの構成、プロット表示が行えます。Code Builderの"Panels and visualization"グループに、このオブジェクトの関数があります。
EA_Ribbon	オブジェクトは対話のための、カスタムのリボンインタフェースの機能を拡張するメソッドを提供します。Code Builderの"User interface"グループに、このオブジェクトの関数があります。
EA_Graph	このオブジェクトを使用すると、グラフのタイトル、色、軸の設定および印刷やコピーといったプロパティの変更が可能になります。Code Builderの"Graphs"グループに、このオブジェクトの関数があります。
EA_Comm	このオブジェクトを使用すると、catmanEasy/APのDAQプロジェクトの一部ではないデバイスと直接通信ができるようになります。Code Builderの"Communication"グループに、このオブジェクトの関数があります。
EA_Math	このオブジェクトを使用して、基礎となる数学エンジンおよびスクリプトの変数や配列を操作します。Code Builderの"Mathematical functions"グループに、このオブジェクトの関数があります。
EA_DataView	このオブジェクトを使用して、DataViewウィンドウの基本的な機能にアクセスします。Code Builderの"DataView"グループに、このオブジェクトの関数があります。
EA_Video	このオブジェクトを使用すると、カメラオブジェクトを作成し、スクリプト毎に解像度、圧縮、キャプチャの開始/停止などの設定することができます。Code Builderの"Video camera"グループに、このオブジェクトの関数があります。

EA

このオブジェクトを通じて、catmanEasy/APの一般的な関数とメソッドにアクセスします。

このオブジェクトを使用すると、特に次のことが可能になります:

- catmanEasy/APの動作モード、"DAQモード"および"解析モード"間を切り替える: EA.OperationMode
- プロジェクトを開く: EA.OpenProject
- 他のプロジェクトから設定をインポートする: EA.ImportFromProject
- catmanEasy/APのメニューおよびツールバーを表示/非表示にする: EA.ConfigureGUI
- メッセージおよびダイアログを表示/非表示にする: EA.ShowTaskDialog、EA.MessageBox、EA.CloseMessage
- catmanEasy/APアプリケーションをスリープ状態にする: EA.Wait
- スクリプトの実行やcatmanEasy/APアプリケーションを終了する: EA.Terminate
- スクリプト エラーを発生させる: EA.ErrorAlert
- 最後に発生した内部エラーの番号および説明の読み取り: EA.LastError、EA.LastErrorText
- catmanEasy/APの一般的な設定の変更/読み取り: EA.SensorDatabase、EA.AppDir、EA.ScriptDir、EA.GetUserDir、EA.ProjectDir
- catmanEasy/APの内部ウィンドウを表示/非表示にする: EA.ShowWindow、EA.HideWindow
- catmanEasyメインウィンドウのタイトルを設定する: EA.SetMainWindowCaption
- 音声ファイルまたはトーンを再生する: EA.PlaySoundFile、EA.PlayTone
- 通知を送信する: `EA.SendEmail`、EA.SendPushNotification
- マウスポインタを設定する: EA.SetMousePointer
- プロセス変数にアクセスし、AutoSequenceまたは直接スクリプトコードスニペット間でデータを交換
- catmanの内部オブジェクトを取得する: EA.GetObject
- ユーザが選択しているオブジェクト名を取得する: EA.GetSelectedObjectName
- catmanのシステム情報を取得する: EA.GetSystemInfo
- 現在の言語を取得する: EA.GetLanguage
- 現在のシステムの日付と時刻を取得します: EA.GetUTCTimeString
- SystemLogオブジェクトを使用する
- ログとステータスを管理する: EA.WriteSysLog、EA.WriteStatus、EA.WriteProgress
- catmanEasy/APのアドインを起動する: EA.StartAddIn

このオブジェクトでは以下のメソッドが公開されています:

EA.AppDir (Read only) ・・・

EA.HighResTime ・・・

EA.AppDir (Read only)

catmanEasy/APのインストールディレクトリを返します。

```
Property EA.AppDir
```

EA.CloseMessage

EA.MessageBoxで開かれた非モーダルウィンドウを閉じます。

```
Function EA.CloseMessage
```

EA.ConfigureGUI

catmanEasy/APのメニューおよびツールバーを、表示または非表示にします。さらに、コンテキストメニュー(ポップアップメニュー)の動作を決定します。

この関数は廃止されました! 代わりにEA_Ribbon.SetRibbonStateおよびEA_Ribbon.SetSysTabStateを使用してください。

コマンドラインの引数/HideGUIで、すべてのGUI要素(リボンバー、メニューおよびコンポーネントウィンドウ)を非表示にすることができます。詳細はcatmanのGUIを非表示にする方法を参照してください。

```
EA.ConfigureGUI(ByVal MenuBar As Integer, ByVal Toolbar As Integer, ByVal StatusBar As Integer, ByVal PopupMenu As Integer) As Long
```

引数	説明
MenuBar	0 = メニューバーの非表示 1 = メニューバーの表示
ToolBar	0 = すべてのツールバーを非表示 1 = メインツールバーのみを表示 2 = パネルツールバーのみを表示 3 = すべてのツールバーを表示
StatusBar	0 = ステータスバーの非表示 1 = ステータスバーの表示
PopupMenu	0 = コンテキストメニューを無効 1 = コンテキストメニューを有効

戻り値	説明
0	エラーなし。
<0	エラー。

EA.CreateProcessVariable

catmanのバージョン5.3以降では、いわゆる"グローバルコード"を導入しています。これにより、共通データの共有がさらに簡単になり、プロセス変数とは対照的に、共通関数の共有も可能になります。詳細については、EA.SetGlobalCodeを参照してください。

新規のプロセス変数を作成します。プロセス変数を使用すると、グローバルスコープに変数を持たない個々の直接スクリプトコードブロック間でデータを交換できます(たとえば、直接スクリプトコードを備えたボタンを使用する場合)。このように、プロセス変数はプロジェクト全体で"グローバル"変数のように機能します。

```
  EA.CreateProcessVariable(ByVal Name As Variant, Optional ByVal Preset As Variant)
As Long
```

引数	説明
Name	プロセス変数名。プロセス変数名は、例えば"%FileName%"のように%文字で囲まなければなりません。ただし、%文字を省略した場合でも、catmanは自動的に追加します。
Preset	任意です。指定した場合、新規変数はこの値を取得し、それ以外は0に初期化します。

戻り値	説明
0	エラーなし。
<0	エラー。既に同名の変数が存在しています。

EA.SetProcessVariable、EA.GetProcessVariableも参照してください。

例

```
  Dim val As String
  EA.CreateProcessVariable("%FileName%")
  EA.SetProcessVariable("%FileName%", "C:\MYDATA\Test.dat")
  EA.GetProcessVariable("%FileName%", val)
```

EA.ErrorAlert

スクリプトエラーのメッセージを表示し、EasyScriptエディタを表示します。

```
EA.ErrorAlert(ByVal ErrorText As String, ByVal ErrLine as String)
```

引数	説明
ErrorText	表示するエラーメッセージ。
ErrLine	エラーの発生したスクリプトエディタの行。同時にメッセージも表示します。

EA.ExecuteAddInMethod

catmanAP用に特別に作成されたActiveXコンポーネントのインスタンスメソッドを実行します。

```
  EA.ExecuteAddInMethod(ByVal ClassName As String, ByVal MethodName As String, Args()
As Variant)
```

引数	説明
ClassName	AddInコンポーネントのクラス名。たとえば周期的データ保存モジュールのクラス名は"CyclicDataSave.clsExternal"、マルチレコーダモジュールのクラス名は"catMultiRecorder.clsMain"です。
MethodName	クラスオブジェクトのメソッド名。
Args	呼び出すメソッドの引数リストに応じた、任意の数のバリアント引数。

EA.GetLanguage

現在のcatmanEasy/APの言語を返します。

```
EA.GetLanguage(LangID As Integer)
```

引数	説明
LangID	言語識別子: 0 = 独語 1 = 英語 2 = 仏語

EA.GetObject

さらに処理をするために、指定したオブジェクトを取得します。公開されていない他のEasyScriptのメソッドやプロパティの内部機能へのアクセスを提供します。グラフやテーブルオブジェクトのコントロールにアクセスしたい場合、このインスタンスは非常に有用です。詳細は、下記の注意事項を参照してください。

これらのオブジェクトを使用した作業は、VBAおよびcatmanの高度な知識を持つ上級プログラマおよびシステムインテグレータを対象としています。HBM catmanプログラマのように、実際にオブジェクトを操作することができます。間違ったプロパティの変更やメソッドの呼び出しは、catmanがクラッシュする恐れがあります。本書では、公開されているオブジェクトの、すべてのプロパティおよびメソッドについて説明していません。詳細はHBMに連絡してください。

```
EA.GetObject(ByVal ObjectType as Integer, ByVal ObjectName As String, Optional
ByVal Panel As Variant) As Object
```

引数	説明
ObjectType	取得するオブジェクトの種類: 0 = 名前による可視化(パネル)オブジェクト。オブジェクトは通常、スコープまたはフローティングパネル上に置くことができます。 スコープやフローティングパネルの場合、Panelパラメータは必須です。 1 = 現在選択している可視化オブジェクト。 これは通常のパネルのみを考慮しています。スコープおよびフローティングパネルはサポートしていません。 2 = リボンバー 3 = ステータスバー 4 = (予約) 5 = チャネル構成ウィンドウ 6 = チャネル情報ウィンドウ 7 = センサリストウィンドウ 8 = テストエクスプローラウィンドウ 9 = Real-time数学ダイアログウィンドウ 10 = 光学インテロゲータのウィンドウ 11 = 制限値構成ダイアログウィンドウ 12 = catmanEasy/APのメインウィンドウ 13 = 可視化パネルの背景(VBのピクチャボックスで、*Cls*、*BackColor*、*Print*、*Line*等のようなメソッドを使用して、背景に独自のコンテンツを描画しも構いません)。*CurrentX*および*CurrentY*プロパティ(TWIPS内)は、*Print*ステートメントの開始点を示します。 14 = システムログオブジェクト。"SystemLogオブジェクトの使用"で、このオブジェクトのメソッドおよびプロパティを使用する方法について説明しています。 15 = catmanEasy/APの印刷エンジン 16 = チャネル構成リスト 17 = 現在選択してるチャネル 18 = 現在選択しているパネルの背景領域 19 = catmanEasy/APのリボンシステム 20 = スコープパネル 21 = フローティングパネル
ObjectName	可視化(パネル)オブジェクト名。
Panel	任意です。省略した場合、オブジェクトのすべての通常のパネルを検索します。 スコープやフローティングパネルの場合、Panelパラメータは必須です。

戻り値	説明
Object	

注記:
catmanの内部オブジェクトは膨大な数のメソッドおよびプロパティを提供してしていますが、その中で関心があるのはほんの僅かです。そのため、catmanのメインウィンドウやリボンシステムのプロパティとは、おそらく干渉することはないでしょう。しかし、オブジェクトへの直接アクセスでの一部領域ではあるかもしれません。例を挙げます:

- 可視化(パネル)オブジェクト(グラフやテーブルなど)の操作。catmanが対話型ユーザに豊富な関数セットを公開していても、これらオブジェクトの強力な関数のサブセットにすぎません。
- リボンコントロールを無効化/非表示
- フォームの色、キャプションの変更など

内部オブジェクトの非常に複雑なインタフェースが理解できるよう、EasyScriptはデフォルトでこれらのデータ型を提供します。変数*As Object*を宣言する代わりに、適切な型を参照して宣言します。そしてインテリセンス機能は、オブジェクトによって公開されているすべての関数がリストされています。型は以下のオブジェクトがあります:

```
  Dim oGraph As pesgo 'Standard graphs, polar diagrams, histogram
  Dim oCursorGraph As cwGraph 'Cursor graph
  Dim oTable As fpSpread 'Tables on a panel, builtin tables of catman window (e.g.
channel list)
  Dim oPrinter As vsPrinter 'catman printing engine
  Dim oCanvas As PictureBox 'catman panel background area
```

canvasオブジェクトは標準のVBピクチャボックスオブジェクトです。使用する前にObject referencesダイアログで、"Visual Basic objects and procedures"が有効になっていることを確認してください。テキストをcanvasに出力するには、組み込みメソッドのEA_Panel.CanvasSetTextを使用する必要があることに注意してください。他のすべての関数は、このデータ型で公開されます。

グラフやパネルオブジェクトにアクセスする場合、組み込みのcatmanルーチン(例えば再描画プロセス)がコードと干渉しないことを確認してください。EA_Panel.GetDirectControlを使用して、catmanがオブジェクトに接触することを防ぎます。これがないとcatmanはオブジェクトに対して実行したアクションを上書き/アンドゥをする可能性があります。

例1: グラフ名"GRAPH_1"にデータをプロット

```
  Dim oGraph As pesgo
  Dim i As Integer

  Set oGraph=EA.GetObject(0, "GRAPH_1")
  EA_Panel.GetDirectControl "GRAPH_1"
  oGraph.PEactions=REINITIALIZE

  oGraph.MainTitle="Test"
  oGraph.Subsets=1
  oGraph.Points=100

  For i=0 To 99
  oGraph.XData(0,i)=i
  oGraph.YData(0,i)=Sqr(i)
  Next

  oGraph.PEactions=REINITIALIZE_RESETIMAGE
```

例2:　チャネルテーブルにアクセスし、現在選択している行を取得

```
  Dim oList As fpSpread
  Dim Row As Integer
  Set oList=EA.GetObject(OB_CHANCONFIG_LIST, "")
  Row=oList.ActiveRow
```

例3:　catmanの可視化パネルの背景領域にアクセスし、テキストを描画

```
  Dim oPanel As PictureBox
  Set oPanel = EA.GetObject(OB_PANELCANVAS, "")
  oPanel.Cls 'Erase background
  oPanel.CurrentX=500 'TWIPS
  oPanel.CurrentY=200 'TWIPS
  'Observe that in order to actually output the text you must use the builtin method
of the EA_Panel class. Text is always output at the CurrentX, CurrentY coordinates.
  EA_Panel.CanvasSetText("Hello world")
```

EA.GetProcessVariable

catmanのバージョン5.3以降では、いわゆる"グローバルコード"を導入しています。これにより、共通データの共有がさらに簡単になり、プロセス変数とは対照的に、共通関数の共有も可能になります。詳細については、EA.SetGlobalCodeを参照してください。

*プロセス変数*の値を取得します。プロセス変数は主にAutoSequenceで使用され、このコマンドはEasyScriptとAutoSequence間のデータ交換が目的です(例えばAutoSequenceにEasyScriptコードのアクションが含まれている場合)。ただし、個々の直接スクリプトブロック間で、データ交換にプロセス変数を使用することもできます(例えば直接スクリプトコードを備えたボタンを使用する場合)。このようにプロセス変数は、プロジェクト全体の"グローバル"変数のように振る舞います。

また、スクリプトからプロセス変数を作成することができます。詳細はEA.CreateProcessVariableを参照してください。

```
EA.GetProcessVariable(ByVal VarName As String, Value As Variant) As Integer
```

引数	説明
Name	プロセス変数名。プロセス変数名は、例えば"%FileName%"のように%文字で囲まなければなりません。%文字を省略した場合でも、catmanは自動的に追加します。
Value	変数の内容。

戻り値	説明
0	エラーなし。
<0	エラー。既に同名の変数が存在している場合など。

EA.SetProcessVariableも参照してください。

例

```
Dim val As String
EA.CreateProcessVariable("%FileName%")
EA.SetProcessVariable("%FileName%", "C:\MYDATA\Test.dat")
EA.GetProcessVariable("%FileName%", val)
```

EA.GetProcessVariableList

*プロセス変数*の名前リストを取得します。プロセス変数は主にAutoSequenceで使用され、このコマンドはEasyScriptとAutoSequence間でのデータ交換が目的です(例えばAutoSequenceにEasyScriptコードアクションが含まれている場合)。ただし、個々の直接スクリプトブロック間で、データ交換にプロセス変数を使用することもできます(例えば、直接スクリプトコードを備えたボタンを使用する場合)。このようにプロセス変数は、プロジェクト全体の"グローバル"変数のように振る舞います。

また、スクリプトからプロセス変数を作成することができます。詳細はEA.CreateProcessVariableを参照してください。

```
EA.GetProcessVariableList(NofVars As Integer, VarNames() As String) As Long
```

引数	説明
NofVars	プロセス変数の数。
VarNames	プロセス変数名のコレクション。

戻り値	説明
0	エラーなし。
<0	エラー。変数が存在しない場合など。

EA.GetProcessVariable、EA.SetProcessVariableも参照してください。

例

```
String i As Integer
Dim NofVars As Integer
Dim VarNames() As String
Dim val As String
EA.CreateProcessVariable("%FileName1%")
EA.SetProcessVariable("%FileName1%", "C:\MYDATA\Test1.dat")
EA.CreateProcessVariable("%FileName2%")
EA.SetProcessVariable("%FileName2%", "C:\MYDATA\Test2.dat")
EA.CreateProcessVariable("%FileName3%")
EA.SetProcessVariable("%FileName3%", "C:\MYDATA\Test3.dat")
EA.GetProcessVariableList NofVars, VarNames
For i = 0 To NofVars - 1
  EA.GetProcessVariable(VarNames(i), val)
Next
```

EA.GetSelectedObjectName

ユーザが選択しているオブジェクト名(例えばチャネルリスト内のチャネル)を返します。

```
EA.GetSelectedObjectName(ByVal ObjectType As Integer, ObjectName As String) As Long
```

引数	説明
ObjectType	0 = リボンタブの"Channel configuration"チャネルリスト内のチャネル 1 = 可視化チャネル選択リストのチャネル 2 = DAQジョブ構成のチャネル固有の設定リストのチャネル 4 = テストエクスプローラで読み込まれているテストのチャネル(解析モード) 5 = DAQジョブ構成ウィンドウで、現在選択されているジョブ 6 = 現在の可視化パネルのキャプション(EA_Panel.GetCurrentPanelを参照) 7 = 可視化パネル上の現在選択しているオブジェクト名 8 = グラフの現在選択しているプロット名 複数のチャネルを選択している場合、"|"を区切り文字として名前を連結します。
ObjectName	オブジェクト名。選択していない場合は空。

戻り値	説明
0	エラーなし。
<0	エラー。

EA.GetSystemInfo

catmanのシステム情報を返します。

```
EA.GetSystemInfo(InfoList() As Variant) As Long
```

引数	説明
InfoList	配列には、少なくとも11の要素が必要です。この関数は配列要素内にcatmanのシステム情報を返します。 InfoList(0) = catmanのバージョン番号 InfoList(1) = catmanのライセンス番号r InfoList(2) = DAQおよび演算チャネルの最大数 InfoList(3) = 一時的データ保管の作業ディレクトリ InfoList(4) = 固定記憶サイズを使用するかどうか: 0 = no, 1 = yes InfoList(5) = 周期記憶モードを使用するかどうか: 0 = no, 1 = yes InfoList(6) = 一時記憶の固定記憶域サイズ(GB単位) InfoList(7) = 現在のセンサデータベースの場所 InfoList(8) = 現在のDAQまたは解析プロジェクトの場所 InfoList(9) = 現在のスクリプトプロジェクトファイルの場所 InfoList(10) = スクリプトの実行中にボタンオブジェクトをロックモードにする 0 = unlocked, 1 = locked ボタンオブジェクトがロックされている場合、サイズの変更/移動や構成が行えません。

戻り値	説明
0	エラーなし。
<0	エラー。無効な配列の次元。

例

```
Dim catInfo(10)
EA.GetSystemInfo catInfo
```

EA.GetUserDir (Read only)

現在のユーザーの作業ディレクトリを返します。

```
Property EA.GetUserDir
```

EA.GetUTCTimeString

現在のシステムの日付と時刻を取得します。システム時間は協定世界時(UTC)で表されます。

```
EA.GetUTCTimeString(strUTC As String)
```

引数	説明
strUTC	yyyy-mm-ddThh-mm-ssZ 形式の現在のUTC時刻(例: 2017-02-28T08-54-32Z)

EA.HighResTime

Windowsオペレーティング システムが起動してから経過した合計秒数を(1マイクロ秒の高解像度で)取得します。この関数を使用すると、高精度の時間間隔測定が可能になります。

```
EA.HighResTime() As Double
```

引数	説明
strUTC	yyyy-mm-ddThh-mm-ssZ 形式の現在のUTC時刻(例: 2017-02-28T08-54-32Z)

例

```
Sub MeasureTimeElapsed()
  Dim T0 As Double
  Dim T1 As Double
  T0 = EA.HighResTime()
  'Your code's execution here...
  T1 = EA.HighResTime()
  MsgBox "Duration of your code's execution in s: " & (T1 - T0)
End Sub
```

EA.HideWindow

catmanEasy/APの内部ウィンドウを閉じます。

この関数は廃止されました! 代わりにEA_Ribbon.SetRibbonStateおよびEA_Ribbon.SetSysTabStateを使用してください。

コマンドラインの引数/HideGUIで、すべてのGUI要素(リボンバー、メニューおよびコンポーネントウィンドウ)を非表示にすることができます。詳細はcatmanのGUIを非表示にする方法を参照してください。

```
EA.HideWindow(ByVal ID As Integer) As Long
```

引数	説明
ID	1 = チャネル設定 2 = DAQジョブの構成 3 = 制限値の構成 4 = テストエクスプローラ 5 = チャネルリスト 6 = スクリプトツールバー

注記

- IDが1および5のウィンドウはDAQモードでのみ可能で、IDが4のウィンドウは解析モードのみです。

EA.ImportFromProject

現在のプロジェクトに、別のプロジェクトの設定をインポートします。

対話モードでは、メインメニューのインポート機能に対応しています。

```
  EA.ImportFromProject(ByVal FileName As String, ByVal ImportType As Integer,
Optional ByVal DoNotOverwrite As Integer) As Long
```

引数	説明
FileName	インポートするプロジェクトファイル名。ファイルが見つからない場合、default\PROJECTフォルダを検索します。
ImportType	0 = 演算チャネル(DAQモード) 1 = 制限値およびイベント監視の設定(DAQモード) 2 = DAQジョブの設定(DAQモード) 3 = 可視化(DAQおよび解析モード) 4 = 演算(解析モード) 5 = チャネル設定(DAQモード) 6 = DAQプロジェクト(DAQモード) 7 = ハードウェアデバイスのないDAQプロジェクト(DAQモード)
DoNotOverwrite	Trueに設定され、*ImportType*が1、2、または3の場合、既存の設定に加えて設定がインポートされます。それ以外の場合、インポートされた設定はそれらを上書きします。*ImportType* =0(演算チャネル)は常に既存の計算にチャネルを追加します。

戻り値	説明
0	エラーなし。
<0	エラー。例えばファイルが存在しない。

EA.LastError (Read only)

最後に呼び出されたcatmanEasy/APメソッドの拡張エラーステータス(catman内部のエラー番号)を返します。
EA.LastErrorTextからテキスト形式でエラーの説明を取得することができます。

```
Property EA.LastError
```

戻り値	説明
0	エラーなし。
<>0	エラー番号。

Errオブジェクト(VBAエラー)も参照してください。

注記

- *EA.LastError*はクエリにより、ゼロにリセットされます。

EA.LastErrorText (Read only)

最後に呼び出されたcatmanEasy/APメソッドのエラーの説明を返します。

```
Property EA.LastErrorText
```

戻り値	説明
	エラーがある場合、空の文字列またはテキスト。

EA.LastErrorも参照してください。

EA.MessageBox

メッセージの表示やユーザに確認を求めます(例えば、はい/いいえ)。VBA標準のMsgBox関数ではなく、このメソッドはcatmanEasy/APの外観でメッセージボックスを表示します。またメッセージウィンドウは、非モーダルで表示することができます。この場合メッセージは、更なるスクリプトの実行やcatmanEasy/APのDAQ転送をブロックしません。

```
EA.MessageBox(ByVal Msg As String, ByVal Title As String, ByVal MsgType As Integer,
ByVal ButtonStyle As Integer, Optional ByVal NonModal As Integer, Optional ByVal
Timeout As Long) As Long
```

引数	説明
Msg	表示するメッセージ。
Title	ウィンドウタイトル。
MsgType	メッセージに表示するアイコンを決定: 0 =情報 1 =警告 2 =エラー 3 =質問 4 =待機
ButtonStyle	表示するボタンを決定: 0 =OK 1 =Yes/No 2 =Yes/No/Cancel -1 = ボタン無し。この引数は、非モーダルウィンドウに許可されています。この場合、ウィンドウはEA.CloseMessageを使用して閉じる必要があります。
NonModal	任意です。Trueの場合、メッセージウィンドウは非モーダルで表示します。この場合メッセージは、以降のスクリプトの実行をブロックしません。ウィンドウが自動的に閉じるまでの時間を指定することができます。 DAQジョブがを実行中の場合、DAQ転送を阻まないよう、メッセージウィンドウは常に非モーダルで表示します。 非モーダル表示のメッセージウィンドウを使用して、ユーザから戻り値を取得することはできません。
Timeout	任意です。メッセージが非モーダルの場合、この引数で指定したミリ秒経過後にメッセージウィンドウを閉じます。この値を指定しないか0の場合、ウィンドウはボタンをクリックすることでのみ閉じることができます。

戻り値	説明
ユーザにより押されたボタン	
6	vbYes(OKまたは、はい)
7	vbNo(Noまたは、いいえ)
2	vbCancel(Cancelまたは、キャンセル)

注記

- 非モーダルで表示しているメッセージボックスは、EA.CloseMessageを使用して閉じる必要があります。
- 非モーダルで表示しているメッセージボックスの戻り値は常に0です。

EA.OpenProject

新規のDAQまたは解析プロジェクトを開きます。

```
EA.OpenProject(ByVal FileName As String) As Long
```

引数	説明
FileName	プロジェクトファイル名(拡張子.MEPまたは.OFP)。

戻り値	説明
0	エラーなし。
<0	エラー。例えばファイルが存在しない。

注記

- ファイルにパスが含まれていない場合、ユーザのドキュメントのデフォルトのプロジェクトフォルダである"\HBM\catmanEasy\PROJECTS"または、EA.ProjectDirで定義したプロジェクトディレクトリを使用します。
  - これは対話型モードと異なります: 対話型モードでのcatmanは、常に最後に使用したフォルダを記憶しています。
- 開くプロジェクトにEasyScriptが含まれている場合、このスクリプトプロジェクトは無視されます。
- このプロジェクトは、現在のcatmanEasy/APの動作モードと一致している必要があります。すなわち、DAQモードで解析プロジェクトを開くことはできません。EA.OperationModeも参照してください。

EA.OperationMode

catmanEasy/APの動作モードを設定または取得します。いくつかのメソッドは特定の動作モードでのみ使用することができないため、このメソッドをしようしてアクティブなモードを問い合わせることができます。

この関数は対話モードで、既に DAQ またはテスト モードになっている場合の動作モードの切り替え(**File → Change to Analysis mode**または**Change to DAQ mode**)に対応します。

```
Property EA.OperationMode
```

値	説明
0	DAQモード。
1	解析モード。

注記

- このメソッドは、catmanEasy/APのすべての状態を呼び出すことができないため、EA.LastErrorを実行して正しいかどうかを確認してください。
- DAQジョブおよびテストパラメータの内部データ構造はクリアされないため、再度切り替えた時に再び有効になります。

例

```
Sub Analysis()
  EA.OperationMode = 1 'Change to Analysis mode.
  EA_Test.Load "C:\MYDATA\TEST5.TST" 'Load test.
  EA_Test.PlotYT 1, "Valve pressure", 1, "GRAPH1", 0, vbBlue 'Display a channel
from this test.
End Sub
```

EA.PausePlaySoundFile

音声ファイルの再生を一時停止します。一時停止中に実行すると、再生の再開を試みます。

```
EA.PausePlaySoundFile() As Long
```

EA.PlaySoundFile、EA.StopPlaySoundFileも参照してください。

EA.PlaySoundFile

音声ファイル(.WAV、.MPEGなど)を再生します。

```
EA.PlaySoundFile(ByVal SoundFile As String) As Long
```

引数	説明
SoundFile	再生するファイル。指定した場所にファイルが見つからない場合、/Projectsフォルダ内を検索します。 関数は再生が終了するのを待たずに、すぐに戻ることに注意してください!

戻り値	説明
0	エラーなし。
<0	エラー。例えばファイルが存在しない。

EA.PausePlaySoundFile、EA.StopPlaySoundFileも参照してください。

EA.PlayTone

指定した周波数および継続時間の単一トーンを再生します。

```
EA.PlayTone(ByVal Frequency As Long, ByVal Duration As Long) As Long
```

引数	説明
Frequency	周波数(Hz)。
Duration	継続時間(ms)。関数は継続時間を待たずに、すぐに戻り値を返すことに注意してください!

EA.ProjectDir

DAQおよび解析プロジェクトのデフォルトディレクトリの設定、または取得。パス無しでプロジェクト名を指定した場合、プロジェクトはこのディレクトリを検索または保存します。

```
Property EA.ProjectDir
```

EA.OpenProject、EA_Test.DataDirも参照してください。

注記

- ディレクトリが存在しない、またはディレクトリ名が無効の場合、古いディレクトリ名は変更せずに、EA.LastErrorはエラーコード-27(不正なディレクトリ)を報告します。
- 対話型モードと異なり、スクリプトが開始されていると、catmanは、最後に使用したフォルダを記憶しません。そこで、EA.ProjectDirで設定したディレクトリまたは、ユーザのドキュメントのデフォルトのプロジェクトフォルダである"\HBM\catmanEasy\PROJECTS"を使用します。

例

```
Sub NewProject()
  EA.ProjectDir = "C:\MYDATA\VALVE TESTS"
  EA.OpenProject "PressureTest1.mep"
End Sub
```

EA.ScriptDir (Read only)

現在実行しているスクリプトのディレクトリを返します。

```
Property EA.ScriptDir
```

EA.SendEmail

スクリプト内から電子メールを送信します。受信者や件名だけでなく、変数*email*型より電子メールプロバイダの接続設定などのパラメータを渡します。

⚠ 電子メールを送信するためには、PCがインターネットにアクセスできることが必要です!
EA.SendEmailを呼び出す前に、email構造体のフィールド(サーバ、ポート、送信者、受信者など)に記入する必要があります。

```
EA.SendEmail(oEmail As email) As Long
```

引数	説明
oEmail	電子メールのすべての内容および接続の設定。

email構造体:

フィールド	説明
Server As String	サーバ名。これは電子メールをホストしているサーバです(例えばsmtp.hbm.com)。
Port As Long	ポート番号。標準のSMTPポートは25です。セキュアなSMTPポートは587です。
Authentication As Boolean	認証モード。使用する場合はtrueを指定します。多くの公共SMTPは必要としています。
User As String	ユーザ名。これは認証モードで使用します。
Password As String	パスワード。これは認証モードで使用します。
SSL As Boolean	SSL。サーバへセキュアな接続を使用する場合はtrueを指定します。多くの公共SMTPサーバは、セキュリティ保護された接続のみを受け入れます。
From As String	送信者のメールアドレス。これは多くの場合、ユーザ名です(例えばusername@hbm.com)。
To As String	受信者のメールアドレス。複数のアドレスは、セミコロン";"で区切ります。
Cc As String	カーボンコピー受信者のメールアドレス。複数のアドレスは、セミコロン";"で区切ります。
Bcc As String	ブラインドカーボンコピー受信者のメールアドレス。複数のアドレスは、セミコロン";"で区切ります。
Subject As String	件名。送信する電子メールのタイトルです。
Message As String	本文。送信する電子メールの内容です。
Attachments As String	添付ファイル。電子メールにファイルを添付します。複数のファイルは、セミコロン";"で区切ります(例えば"C:\test.txt;D:\MyFolder\anotherFile.zip")。一つ以上のファイルが見つからない場合、それ以外を送信します。
LastErrorDescription As String	最後に発生したエラーので説明。このプロパティは読み取り専用です。エラーが発生した場合、catmanが記入します。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- エラーが発生した場合、oEmailの"*LastErrorDescription*"プロパティにエラーの説明が記入されます。
- 誤った構成(サーバ名やポート番号の誤り)ではエラーを返しません。

例

```
Sub SendEmailViaHBMmail()
   Dim MyEmail As email
   Dim RetVal As Long
   MyEmail.Server = "smtp.hbm.com"
   MyEmail.Port = 587
   MyEmail.Authentication = True
   MyEmail.User = "myAccount@hbm.com"
   MyEmail.Password = "myPassword"
   MyEmail.SSL = True
   MyEmail.From = "myAccount@hbm.com"
   MyEmail.To = "someone@domain.com;someoneelse@domain2.com"
   MyEmail.Subject = "Subject of the message"
   MyEmail.Message = "Body of the message"
   MyEmail.Attachments = "C:\FileToBeAttached.txt"

  If EA.SendEmail(MyEmail) < 0 Then
    MsgBox MyEmail.LastErrorDescription
  End If
End Sub
```

EA.SendPushNotification

プッシュ通知を送信します。

⚠ 電子メールを送信するためには、PCがインターネットにアクセスできることが必要です!

```
  EA.SendPushNotification(ByVal PushID As String, ByVal Subject As String, ByVal
Message As String, ByVal Color As Long) As Long
```

引数	説明
PushID	メッセージのルーティング先となるプッシュチャネルを一意に識別するキー。*PushID*は、catmanイベント監視設定でリクエストできます("Push notification"を選択し、"Edit..."をクリックする必要があります)。
Subject	プッシュ通知の件名。
Message	プッシュ通知の本文。
Color	プッシュ通知の色

戻り値	説明
0	エラーなし。
<0	エラー。

ℹ プッシュ通知機能を使用するには、EasyMonitoringモジュールまたはcatmanAPライセンスが必要です。

例

```
Sub SendPushMsg()
  Dim MV As Double
  EA_DAQ.CurrentReading "Valve pressure", MV
  If MV > 7.5 Then
    EA.SendPushNotification "00000000", "Warning", "Value over range", vbRed
  End If
End Sub
```

EA.SensorDatabase

catmanEasy/APが使用するSensor Databaseを読み込み、または設定します。

対話型モードでは、**Options** → **Sensors**、**Sensor Database**タブのSensor Databaseで設定や読み取りが行えます。

```
Property EA.SensorDatabase
```

例

```
Sub ChangeSensordatabase()
  EA.SensorDatabase = "C:\catmanEasy/AP\HBM_SENSORDATABASE.SDB" 'Change Sensor Database.
End Sub
```

EA.SetGlobalCode

このメソッドを使用すると、その後の直接スクリプト コードの実行間でグローバルに使用できる変数宣言と関数(ボタン、自動プロシージャ実行、補助チャネルなど)を提供できます。そのためグローバルコードでは、プロセス変数(EA.CreateProcessVariable、EA.GetProcessVariable、EA.SetProcessVariable)を必要とせずに、そのようなダイレクトスクリプト スニペット間で情報を共有できます。

参照: 直接スクリプトコードの使用方法

```
EA.SetGlobalCode(ByVal Code As String, Optional ByVal FileName As String)
```

引数	説明
Code	グローバルに使用するためにロードされるコード(例: "Global MyVar As Variant")。
FileName	オプション。指定した場合、コードは*FileName*(パスを含む)で指定したテキストファイルから取得します。この場合、Codeパラメータは無視されます。

戻り値	説明
0	エラーなし。
<0	エラー。

通常、プロジェクトの最初にグローバル コードをロードします。自動実行ステージ"After catman start"または"After project load"。ただし、後からグローバル コードを変更することは問題ありません。

注記

- *直接スクリプト*コードにのみ適用されます。成熟したスクリプトプロジェクトの場合、グローバル変数を宣言できる可能性がすでにあるため、これを行う必要はありません。
- グローバル コードに"Private Sub Main"が含まれている場合は、グローバルコードがロードされると実行されます。キーワード"Private"が必須であることに注意してください。そうでないと、グローバルコードの*Main*関数が、グローバルコードを使用するスクリプトの*Main*関数と競合します。
- *Main*以外の関数やSubを追加する場合は、必ずそれらを*Public*として宣言してください。
- *Global*によってグローバルにアクセスできるようにする変数を必ず宣言してください。

例

次の例で、すべての直接スクリプト項目でグローバルに使用できる変数をいくつか示します:

```
Dim Code As String

Code = "Global MaxSize As Integer" & vbcrlf
Code = Code & "Global Username As String" & vbcrlf
Code = Code & "Global Timeout As Double"

EA.SetGlobalCode Code
```

次の例では、いくつかの初期化コードを追加します:

```
Dim Code As String

Code = "Global MaxSize As Integer" & vbcrlf
Code = Code & "Global Username As Double" & vbcrlf
Code = Code & "Global Timeout As Double" & vbcrlf
Code = "Private Sub Main" & vbcrlf
Code = Code & "Timeout = 5000" & vbcrlf
Code = Code & "End Sub"

EA.SetGlobalCode Code
```

COMオートメーションオブジェクトをグローバルに提供することも可能です:

```
Dim Code As String

Code = "Global gExcel As Object" & vbcrlf
Code = Code & "Private Sub Main" & vbcrlf
Code = Code & "Set gExcel = CreateObject(""Excel.Application"")" & vbcrlf
Code = Code & "End Sub" & vbcrlf

EA.SetGlobalCode Code
```

この後、それ以降のすべての直接スクリプト実行では、Excelのリファレンス*gExcel*を利用できます。
グローバルコードは、関数のコンテナとしても機能します:

```
Dim Code As String

Code = Code & "Public Function DoSomething(vIn As Double) As Double" & vbcrlf
Code = Code & "DoSomething = sqrt(vIn)" & vbcrlf
Code = Code & "End Function" & vbcrlf

Code = Code & "Public Function DoSomethingElse(vIn As Double) As Double" & vbcrlf
Code = Code & "DoSomethingElse = log(vIn)" & vbcrlf
Code = Code & "End Function" & vbcrlf

EA.SetGlobalCode Code
```

EA.SetMainWindowCaption

catmanEasyメインウィンドウのタイトルを設定します。

```
EA.SetMainWindowCaption(ByVal Caption As String)
```

引数	説明
Caption	catmanメインウィンドウンの新しいタイトルテキスト。

注記

- タイトルの変更は、現行のセッション中のみ、利用可能です。プロジェクトの読み込み/保存や、作業モードを切り替えると(DAQモード <-> 解析モード)、再びデフォルトのタイトルに上書きされます。

EA.SetMousePointer

catmanに表示されるマウスポインタを設定します。

```
EA.SetMousePointer(ByVal MousePointer As Integer, Optional ByVal MouseIcon As String) As Long
```

引数	説明
MousePointer	catman に表示されるマウスポインタを設定します: 0 = デフォルト 1 = Arrow 2 = Cross(十字カーソル) 3 = Ibeam 4 = Icon(正方形内の小さな正方形) 5 = Size(東西南北を指す四方向の矢印) 6 = SizeNESW(北東と南西を指す二重矢印) 7 = SizeNS(北と南を指す二重矢印) 8 = SizeNWSE(北西と南東を指す二重矢印) 9 = SizeWE(西と東を指す二重矢印) 10 = UpArrow(上方向矢印) 11 = Hourglass(砂時計) 12 = No(不可) 13 = AppStatting(矢印と砂時計) 14 = Help(矢印と疑問符) 15 = SizeAll(サイズ変更) 99 = MouseIconパラメータで指定されたカスタムアイコン
MouseIcon	オプション。カスタム アイコンのファイル名のパス。 *MousePointer*が99に設定されている場合にのみ必要です。

戻り値	説明
0	エラーなし。
<0	エラー。

EA.SetProcessVariable

*プロセス変数*の値を取得します。プロセス変数は主にAutoSequenceで使用され、このコマンドはEasyScriptとAutoSequence間のデータ交換が目的です(例えばAutoSequenceにEasyScriptコードのアクションが含まれている場合)。ただし、個々の直接スクリプトブロック間で、データ交換にプロセス変数を使用することもできます(例えば直接スクリプトコードを備えたボタンを使用する場合)。このようにプロセス変数は、プロジェクト全体の"グローバル"変数のように振る舞います。

また、スクリプトからプロセス変数を作成することができます。詳細はEA.CreateProcessVariableを参照してください。

```
EA.SetProcessVariable(ByVal VarName As String, ByVal NewValue As Variant) As
Integer
```

引数	説明
VarName	プロセス変数名。プロセス変数名は、例えば"%FileName%"のように%文字で囲まなければなりません。ただし、%文字を省略した場合でも、catmanは自動的に追加します。
NewValue	変数の内容。

戻り値	説明
0	エラーなし。
<0	エラー。変数が存在しない場合など。

EA.GetProcessVariableも参照してください。

例

```
Dim val As String
EA.CreateProcessVariable("%FileName%")
EA.SetProcessVariable("%FileName%", "C:\MYDATA\Test.dat")
EA.GetProcessVariable("%FileName%", val)
```

EA.ShowTaskDialog

確認のためのメッセージやプロンプトを表示します(例えば、はい/いいえ)。VBA標準関数のMsgBoxではなく、このメソッドはcatmanEasy/APの外観でメッセージボックスを表示します。

```
EA.ShowTaskDialog(ByVal Title As String, ByVal MainText As String, ByVal
ContentText As String, ByVal ExpandedText As String, ByVal FooterText As String,
ByVal MainIcon As Long, ByVal FooterIcon As Long, ByVal ButtonStyle As Integer,
Optional DontShowAgain As Integer, Optional ByVal DontShowAgainText As String,
Optional ByVal MainIconFile As String, Optional ByVal FooterIconFile As String) As
Long
```

引数	説明
Title	ウィンドウタイトル。
MainText	太字のヘッダ行("task")。
ContentText	ヘッダ行の下に続く内容。
ExpandedText	"Details"ボタンを押したときに表示されるテキスト。
FooterText	ボタンの下に表示するテキスト。
MainIcon	0 = アイコン無し 1 = 警告 2 = エラー 3 = 情報 4 = セキュリティ 5 = カスタムアイコン。*MainIconFile*を指定(フルパスが必要)します。
FooterIcon	0 = アイコン無し 1 = 警告 2 = エラー 3 = 情報 4 = セキュリティ 5 = カスタムアイコン。*FooterIconFile*を指定(フルパスが必要)します。
ButtonStyle	表示するボタンを決定: 0 = OK 1 = OK + CANCEL 2 = YES + NO 3 = YES+NO+CANCEL 4 = RETRY + CANCEL 5 = YES+NO+RETRY 6 = IGNORE + CANCEL 7 = IGNORE + RETRY + CANCEL 8 = NONE
DontShowAgain	trueの場合、ダイアログは"Don't show again"チェックボックスが表示されます。チェックボックスのキャプションは、*DontShowAgainText*で設定することができます。ユーザがチェックした場合、メソッドの戻り値はtrueとなり、それ以外はfalseです。
DontShowAgainText	Don't show againチェックボックスのキャプション。空の文字列の場合、"Don't show this dialog again"を表示します。
MainIconFile	MainIcon =5の場合のアイコンのファイルパス。
FooterIconFile	FooterIcon =5の場合のアイコンのファイルパス。

戻り値	説明
ユーザにより押されたボタン	
1	vbOK (OK)
6	vbYes(はい)
7	vbNo(Nいいえ)
2	vbCancel(キャンセル)
5	vbIgnore(無視)
4	vbRetry(再試行)

EA.ShowWindow

catmanEasy/APの内部ウィンドウを表示します。

```
EA.ShowWindow(ByVal ID As Integer) As Long
```

引数	説明
ID	0 = メインウィンドウ 1 = チャンネル設定 2 = DAQジョブ構成 3 = 制限値構成(モーダル) 4 = テストエクスプローラ 5 = チャンネルリスト 6 = スクリプトツールバー 7 = クラウド データ ストリーミング 8 = CANバス構成

戻り値	説明
0	エラーなし。
<0	エラー。ウィンドウが存在しない場合など。

注記

- IDが1、2、3および5のウィンドウはDAQモードのみで、IDが4のウィンドウは解析モードでのみ許可されています。
- 非モーダルウィンドウを閉じるにはEA.HideWindowを使用します。
- フローティングパネルのみを表示したい場合、メインウィンドウを閉じると便利です - メインウィンドウ非表示の場合でも、フローティングパネル非表示になりません。

EA.StartAddIn

catmanAP用に特別に作成されたActiveXコンポーネントを起動します。

```
EA.StartAddIn(ByVal ClassName As String)
```

引数	説明
ClassName	AddInコンポーネントのクラス名。たとえば周期的データ保存モジュールのクラス名は"CyclicDataSave.clsExternal"、マルチレコーダモジュールのクラス名は"catMultiRecorder.clsMain"です。

EA.StopPlaySoundFile

音声ファイルの再生を停止します。

```
EA.StopPlaySoundFile() As Long
```

EA.PlaySoundFile、EA.PausePlaySoundFileも参照してください。

EA.Terminate

それ以上のクエリを行わずにcatmanEasy/APを終了します。

```
EA.Terminate(Optional ByVal Mode As Integer) As Long
```

引数	説明
Mode	任意です。 0または不明 = スクリプトの実行を終了 1 =catmanEasy/APを終了 2 =catmanEasy/APを再起動

戻り値	説明
<0	終了不可。

EA.Wait

アプリケーションをスリープします。これはバックグラウンドで実行中のプロセスの終了を待つために必要になる場合があります。

```
EA.Wait(ByVal Milliseconds As Long)
```

引数	説明
Milliseconds	ミリ秒単位の待機時間。

EA.WatchDogControl

ウォッチドッグプロセスを制御します。ウォッチドッグプロセスは、DAQジョブの実行中にcatmanの実行異常が発生したかどうかを検出するために使用します。

```
EA.WatchDogControl(ByVal Mode As eWatchDogMode) As Long
```

引数	説明
Mode	ウォッチドッグが実行する必要のあるアクション: 0 = ウォッチドッグ プロセスを開始します。起動後、ウォッチドッグはデフォルトで無効になります。 1 = ウォッチドッグを有効にしてcatmanを監視します。 2 = ウォッチドッグを無効にしてcatmanを監視します。 3 = ウォッチドッグプロセスを終了します。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- この関数は、ウォッチドッグを手動で制御する場合にのみ必要です。レジストリエントリ "HKEY_CURRENT_USER\Software\VB and VBA Program Settings\catmanEasy\DEFAULTS\WATCHDOG"が存在し、すでに"1"に設定されている場合、ウォッチドッグはcatmanによって自動的に制御されます。ウォッチドッグはDAQジョブの開始時に有効化され、DAQ ジョブ終了時に無効化されます。

- ウォッチドッグサービスは、catmanによって定期的に(間隔 = 1秒)更新されるテキストファイル(catmanユーザディレクトリ内のWATCHDOG.TXT)の最終変更時刻を毎秒監視し、DAQジョブの実行中に実行異常が発生したかどうかを検出します(例: catman not responding)。たとえば、catmanがクラッシュした場合、ウォッチドッグサービスは30秒以上エントリが追加されていないことを検出します。この場合、ウォッチドッグプロセスはまずcatmanを終了し(必要に応じて、たとえばcatmanが応答しない場合)、次にcatmanを再起動し、最後に自身を終了します。catmanを新たに開始すると、ウォッチドッグプロセスが再び有効になります。

例

```
Sub WatchTask()
  Dim status As Long
  EA.WatchDogControl WATCHDOG_START
  EA_Job.Start 1
  EA.WatchDogControl WATCHDOG_ACTIVATE
  Do
    'Your code here...
    status = EA_Job.Status
  Loop While status > 0
  EA.WatchDogControl WATCHDOG_DEACTIVATE
  EA.WatchDogControl WATCHDOG_STOP
End Sub
```

EA.WriteProgress

プログレスバーの進行状況を更新します。

```
EA.WriteProgress(ByVal vNewValue As Integer, Optional ByVal Msg As String) As Long
```

引数	説明
vNewValue	現在の進行状況の値。
Msg	オプション。表示するメッセージ。

EA.WriteStatus

指定したステータスバーにステータスを表示します。

```
  EA.WriteStatus(ByVal Panel As Integer, ByVal Msg As String, Optional ByVal ImgIndex
As Integer) As Long
```

引数	説明
Panel	ステータスバーのインデックス: 0 = メインウィンドウの左下隅にあるメイン ステータスバー 1 = メインステータス バーの右側にあるカスタムステータス バー 2 = ジョブのステータスを表示するステータスバー 3 = ファイルエクスポートのステータスを表示するステータスバー 4 = これまでに取得したサンプルの数を表示するステータスバー 5 = 残り時間の値を表示するステータスバー 6 = サンプルレートを表示するステータスバー 7 = RT-Lagを表示するステータスバー 8 = スクリプトの実行ステータスを表示するステータスバー 9 = 進行状況を表示するステータスバー 10 = DAQ ステータスを表示するステータスバー 11 = マスター/スレーブ情報を表示するステータスバー
Msg	表示するメッセージ。
ImgIndex	オプション。ステータスバーに表示するアイコンのID。

EA.WriteSysLog

システムログにメッセージを追加します。

```
  EA.WriteSysLog(ByVal Msg As String, ByVal Level As Integer, Optional ByVal Comment
As String, Optional ByVal AuxFile As String) As Long
```

引数	説明
Msg	システム ログに書き込むメッセージ。
Level	ステータスバーのインデックス: 0 = 情報 1 = 警告 2 = エラー 3 = 警報 4 = イベント
Comment	オプション。ログメッセージにコメントを付けます。
AuxFile	オプション。ログ メッセージによって参照される補助ファイル。画像ファイル(Windowsメタファイル、ビットマップ、JPEGまたはGIF)またはテキスト(テキストまたはHTML)ファイルを使用できます。

戻り値	説明
0	エラーなし。
<0	エラー。

SystemLogオブジェクトの使用

*SystemLog*オブジェクトを使用すると、catmanの内部システムログデータベースにアクセスすることができます。このデータベースはMS Accessのデータベースです。通常、システム ログはcatmanに開いて管理しているため、ほとんどの場合、エントリの追加や取得するだけで済みます。

最初にシステムログオブジェクトへの参照を取得する必要があります:

```
Const SYSLOG = 14
Dim obLog As Object
Set obLog = EA.GetObject(OB_SYSLOG, "")
```

次のセクションより、最も有用なプロパティおよびメソッドを示します:

1) 新規のログ項目を追加

```
Public Sub AddLog(LogText As String, Level As Integer, Optional Comment As String, Optional AuxFile As String)
```

引数	説明
LogText	ログメッセージテキスト。
Level	0 = 情報、1 = 警告、2 = エラー。
Comment	追加するコメントテキスト。
AuxFile	ファイルが指定されている場合、Log Viewer機能の"Show descriptor file"を表示します。ビューアはプレーンテキストファイル、画像、HTML、MS Wordドキュメントを表示できます。

2) ログ項目の取得:

```
Public Sub GetLog(ByVal Index As Long, LogText As String, Level As Integer, Comment As String, AuxFile As String, DateTimeCreated As Double)
```

引数	説明
Index	取得するログ項目のインデックス。ログ内の最新のインデックスを取得するため、*LastIndex*プロパティを使用します。これを*NumEntries*プロパティと混同しないでください - これはデータベース内のインデックスを正しく反映しません!
LogText	ログメッセージテキスト。
Level	0 = 情報、1 = 警告、2 = エラー。
Comment	追加するコメントテキスト。
AuxFile	追加の記述ファイル。
DateTimeCreated	OLEの時間形式(1900年1月1日からの秒)。

3) 現在のログ内の項目数および、最後(最新)項目のインデックスを決定します。インデックス値は通常、*NumEntries*と同じではありません(例えば、インデックス範囲は5000から5099でも*NumEntries*は100の可能性があります - これは追加/削除操作に依存します)。

```
Public Property NumEntries
Public Property LastIndex
```

4) 項目の削除:

```
Public Sub DeleteLog(ByVal Index As Long)
```

引数	説明
Index	削除するログ項目のインデックス。ログ内の最新のインデックスを取得するため、*LastIndex*プロパティを使用します。これを*NumEntries*プロパティと混同しないでください - これはデータベース内のインデックスを正しく反映しません!

5) Log Viewerの起動:

```
Public Sub obLog.ShowViewerEx(ByVal Mode As Integer)
```

引数	説明
Mode	0 = モーダル、1 = 非モーダル

EA_Comm

*EA_Comm*オブジェクトを使用すると、catmanEasy/APのDAQプロジェクトではないデバイス(電圧計などの非HBMデバイス)と直接通信することができます。この通信はDAQジョブから独立しています。そのため、DAQ操作を行っているデバイスと干渉しないよう、注意する必要があります。

通信用の一般的なコマンド。

- コマンドのタイムアウトを設定: EA_Comm.SetTimeout
- 現在時刻を秒単位で返す: EA_Comm.TIME_GetSystemTime

以下のコマンドは、COM1 ... COM4のRS-232デバイスと通信します。

- 接続を開く: EA_Comm.RS232_OpenPort
- 接続を閉じる: EA_Comm.RS232_ClosePort
- コマンド送信: EA_Comm.RS232_WritePort
- 文字列受信: EA_Comm.RS232_ReadPort
- バイト受信: EA_Comm.RS232_ReadPortBinary
- バッファのクリア: EA_Comm.RS232_FlushInQ

以下のコマンドは、GPIB(IEEE488)デバイスと通信します。NI 488準拠のボードおよびドライバのみをサポートしています。これらのコマンドを使用する前に、ボード用のドライバソフトウェアをインストールする必要があります。

- 接続を開く: EA_Comm.GPIB_OpenPort
- 接続を閉じる: EA_Comm.GPIB_ClosePort
- 指定したアドレスのデバイスを選択: EA_Comm.GPIB_SelectAddress
- コマンド送信: EA_Comm.GPIB_WritePort
- 文字列受信: EA_Comm.GPIB_ReadPort
- バイト受信: EA_Comm.GPIB_ReadPortBinary

以下のコマンドは、TCP/IPソケット(イーサネット)通信を使用します。

- 接続を開く: EA_Comm.TCPIP_OpenPort
- 接続を閉じる: EA_Comm.TCPIP_ClosePort
- コマンド送信: EA_Comm.TCPIP_WritePort
- 文字列受信: EA_Comm.TCPIP_ReadPort
- バイト受信: EA_Comm.TCPIP_ReadPortBinary

以下のコマンドは外部時刻ソースのNTP (Network Time Protocol)およびPTP(Precision Time Protocol)を制御します。

- NTPクライアントの NTPサービスを開始: EA_Comm.NTP_StartClient
- NTPクライアントの NTPサービスを停止: EA_Comm.NTP_StopClient
- NTPの同期状態を取得: EA_Comm.NTP_GetSyncStatus
- PTPの同期状態を取得: EA_Comm.PTP_GetSyncStatus

以下のコマンドは、ローカルファイルをFTP/SFTPサーバにアップロードするために使用します。

- FTP/SFTP接続の初期化: EA_Comm.FTP_Init
- FTP/SFTP接続用のプロキシサーバを初期化: EA_Comm.FTP_InitProxy
- ログインしてFTP/SFTPサーバに接続: EA_Comm.FTP_LogIn
- ログオフし、FTP/SFTPサーバへの接続を終了: EA_Comm.FTP_LogOut
- FTP/SFTPサーバ上の作業ディレクトリを設定: EA_Comm.FTP_SetWorkingDirectory
- ローカルファイルをFTP/SFTPサーバにアップロード: EA_Comm.FTP_UploadFile
- FTP/SFTPアップロードのステータスを取得: EA_Comm.FTP_UploadStatus
- FTP/SFTPサーバへのファイルのアップロードを一時停止: EA_Comm.FTP_PauseUpload
- FTP/SFTPサーバへのファイルのアップロードを再開: EA_Comm.FTP_ResumeUpload
- ファイルとフォルダを FTP/SFTPサーバからローカルディレクトリにダウンロード: EA_Comm.FTP_DownloadFile
- FTP/SFTPアップロード中に発生した最後のエラー番号を取得: EA_Comm.FTP_LastError
- FTP/SFTPアップロード中に発生した最後のエラーの説明を取得: EA_Comm.FTP_LastErrorDescription

以下のコマンドは、ファイル操作を実行します。

- ファイルを開く: EA_Comm.FILE_Open
- ファイルを閉じる: EA_Comm.FILE_Close
- ファイルが存在するかどうかを確認: EA_Comm.FILE_Exists
- ファイル ポインタの移動: EA_Comm.FILE_SetFilePointer
- データ型に応じてファイルの内容を読み取り: EA_Comm.FILE_Read
- データ型に応じてファイルの内容を書き込み: EA_Comm.FILE_Write

以下のコマンドは、時系列データベースシステムInfluxDBの操作をサポートします。

- influxDBサーバに接続: EA_Comm.INFLUX_Connect
- influxDBサーバ上に新しいデータベースを作成: EA_Comm.INFLUX_CreateDatabase
- InfluxDBデータベースからのデータのクエリ: EA_Comm.INFLUX_Query
- データベースへのデータの書き込み: EA_Comm.INFLUX_WriteField、EA_Comm.INFLUX_WriteMultipleFields、EA_Comm.INFLUX_WriteSeries、EA_Comm.INFLUX_WriteFile

以下のコマンド セットは、基本的なインターネット操作を実行するために使用します。

- サーバにログイン: EA_Comm.INET_LogIn
- リモートサーバへのリクエストを実行: EA_Comm.INET_Execute
- データの取得: EA_Comm.INET_GetData
- 接続状態の取得: EA_Comm.INET_GetState
- 現在のリクエストをキャンセルし、すべての接続を閉じる: EA_Comm.INET_Cancel

このオブジェクトでは以下のメソッドが公開されています:

EA_Comm.FILE_Close ・・・

EA_Comm.FTP_PauseUpload ・・・

EA_Comm.FILE_Close

このメソッドは、ファイルへの直接アクセスを可能にするメソッドのグループに属しています(catman標準の.BINファイルを使用する場合、最も便利です)。FILE_xxxx グループのメソッドは、ファイルアクセス用のVBA組み込み関数よりもはるかに強力です。したがって、たとえば、FILE_SetFilePointerメソッドは2^32を超えるオフセットを処理できます(VBAのSeek()メソッドでは不可能です)。

FILE_Openで以前に開いたファイルを閉じます。

```
Public Function FILE_Close(ByVal FileHandle As Long) As Long
```

引数	説明
FileHandle	FILE_Openによって返されたファイル ハンドル。

参照

FILE_Open、FILE_Exists、FILE_SetFilePointer、FILE_Read、FILE_Write

EA_Comm.FILE_Exists

このメソッドは、ファイルへの直接アクセスを可能にするメソッドのグループに属しています(catman標準の.BINファイルを使用する場合、最も便利です)。FILE_xxxx グループのメソッドは、ファイルアクセス用のVBA組み込み関数よりもはるかに強力です。したがって、たとえば、FILE_SetFilePointerメソッドは2^32を超えるオフセットを処理できます(VBAのSeek()メソッドでは不可能です)。

ファイルが存在するかどうかを確認します。この関数は、VBA組み込みのDir$()関数よりも強力です(たとえば、この関数はネットワークドライブを常に正しく処理するとは限りません)。

```
Public Function FILE_FileExists(ByVal FileName As String) As Long
```

引数	説明
FileName	パスを含むファイル名。

戻り値	説明
TRUE	ファイルが存在する。

参照

FILE_Open、FILE_Close、FILE_Exists、FILE_SetFilePointer、FILE_Read、FILE_Write

EA_Comm.FILE_Open

このメソッドは、ファイルへの直接アクセスを可能にするメソッドのグループに属しています(catman標準の.BINファイルを使用する場合、最も便利です)。FILE_xxxx グループのメソッドは、ファイルアクセス用のVBA組み込み関数よりもはるかに強力です。したがって、たとえば、FILE_SetFilePointerメソッドは2^32を超えるオフセットを処理できます(VBAのSeek()メソッドでは不可能です)。

書き込み、読み取り、またはその両方のためにファイルを開き、後続のすべてのFILE_xxxx呼び出しで使用されるハンドルを返します。

```
  Public Function FILE_Open(ByVal FileName As String, ByVal AccessMode As Integer)
As Long
```

引数	説明
FileName	パスを含むファイル名。

戻り値
後続のすべての FILE_xxxx 呼び出しで使用されるハンドル。

参照

FILE_Close、FILE_Exists、FILE_SetFilePointer、FILE_Read、FILE_Write

EA_Comm.FILE_Read

このメソッドは、ファイルへの直接アクセスを可能にするメソッドのグループに属しています(catman標準の.BINファイルを使用する場合、最も便利です)。FILE_xxxx グループのメソッドは、ファイルアクセス用のVBA組み込み関数よりもはるかに強力です。したがって、たとえば、FILE_SetFilePointerメソッドは2^32を超えるオフセットを処理できます(VBAのSeek()メソッドでは不可能です)。

読み取りたいデータ型に応じて、いくつかの型依存関数を使用できます:

```
  Public Function FILE_ReadDouble(ByVal FileHandle, ByVal ByteCount As Long, lpData
As Double)
  Public Function FILE_ReadSingle(ByVal FileHandle, ByVal ByteCount As Long, lpData
As Single)
  Public Function FILE_ReadLong(ByVal FileHandle, ByVal ByteCount As Long, lpData As
Long)
  Public Function FILE_ReadInt(ByVal FileHandle, ByVal ByteCount As Long, lpData As
Integer)
  Public Function FILE_ReadByte(ByVal FileHandle, ByVal ByteCount As Long, lpData As
Byte)
```

引数	説明
FileHandle	FILE_Openによって返されたファイルハンドル。
ByteCount	読み取るバイト数。

lpData	データが書き込まれる場所へのポインタ。データを配列に読み込む場合は、この配列の0番目の要素Arr(0)を指定します。
	例:
	`Dim IntVariable As Integer`
	`Dim DoubleVariable As Double`
	`Dim DoubleArray(100) As Double`
	
	`FILE_ReadByteIn fhnd, 2, IntVariable`
	`FILE_ReadDouble fhnd, 8, DoubleVariable`
	`FILE_ReadDouble fhnd, 8*100, DoubleArray(0)`

戻り値
成功すると、関数は新しいファイルポインタの位置をバイト単位で返します。NewPos=0およびRelative=FILE_CURRENTを指定してこの関数を呼び出すと、ファイルの先頭を基準とした現在のファイルポインタの位置が返されます。

参照

FILE_Open、FILE_Close、FILE_Exists、FILE_SetFilePointer、FILE_Write

例

```
Sub Main
Dim i As Integer
Dim DataFile As String
Dim ChanCount As Integer
Dim SampleCount As Long
Dim ChanOffset As Double
Dim ChanIndex As Integer
Dim fhnd As Long
Const PAR_NAME=0
Const PAR_LEN=2
Const PAR_OFFSET=5

DataFile = "E:\ShipEngineTest.bin"
EA_Test.GetDataFileChanCount DataFile, ChanCount
```

```
ReDim ParamList(ChanCount, 64) As Variant
EA_Test.GetDataFileInfo DataFile, ParamList
'Look for the channel named "p_Cyl". Observe that the channel index in ParamList is
zero based!
For i=0 To ChanCount-1
  If StrComp(ParamList(i, PAR_NAME),"p_Cyl",vbTextCompare)=0 Then
  ChanIndex=i
  'Retrieve number of samples and offset of the data section for this channel
  SampleCount = ParamList(i,PAR_LEN)
  ChanOffset = ParamList(i,PAR_OFFSET)
  Exit For
  End If
Next

'Now use the FILE_xxx methods of EA_Comm to read the channel data into an array
ReDim CData(SampleCount) As Double

fhnd = EA_Comm.FILE_Open (DataFile,ACCESS_READ)
EA_Comm.FILE_SetFilePointer fhnd, ChanOffset,FILE_BEGIN
EA_Comm.FILE_ReadDouble fhnd, SampleCount*8, CData(0)
EA_Comm.FILE_Close fhnd

End Sub
```

EA_Comm.FILE_SetFilePointer

このメソッドは、ファイルへの直接アクセスを可能にするメソッドのグループに属しています(catman標準の.BINファイルを使用する場合、最も便利です)。FILE_xxxx グループのメソッドは、ファイルアクセス用のVBA組み込み関数よりもはるかに強力です。したがって、たとえば、FILE_SetFilePointerメソッドは2^32を超えるオフセットを処理できます(VBAのSeek()メソッドでは不可能です)。

指定したファイルのファイルポインタを移動します。

```
  Public Function FILE_SetFilePointer(ByVal FileHandle As Long, ByVal NewPos As
Double, ByVal Offset As eFilePointerOffset) As Double
```

引数	説明
FileHandle	FILE_Openによって返されたファイルハンドル。
NewPos	ファイルの先頭または現在のファイルポインタ位置を基準としたファイルポインタの新しい位置(バイト単位)。
Offset	0: FILE_BEGIN 1: FILE_CURRENT

戻り値
成功すると、関数は新しいファイルポインタの位置をバイト単位で返します。NewPos=0およびOffset=1を指定してこの関数を呼び出すと、ファイルの先頭を基準とした現在のファイルポインタの位置が返されます。

参照

FILE_Open、FILE_Close、FILE_Exists、FILE_Read、FILE_Write

EA_Comm.FILE_Write

このメソッドは、ファイルへの直接アクセスを可能にするメソッドのグループに属しています(catman標準の.BINファイルを使用する場合、最も便利です)。FILE_xxxx グループのメソッドは、ファイルアクセス用のVBA組み込み関数よりもはるかに強力です。したがって、たとえば、FILE_SetFilePointerメソッドは2^32を超えるオフセットを処理できます(VBAのSeek()メソッドでは不可能です)。

書き込むデータ型に応じて、いくつかの型依存関数を使用できます:

```
  Public Function FILE_WriteDouble(ByVal FileHandle, ByVal ByteCount As Long, lpData
As Double)
  Public Function FILE_WriteSingle(ByVal FileHandle, ByVal ByteCount As Long, lpData
As Single)
  Public Function FILE_WriteLong(ByVal FileHandle, ByVal ByteCount As Long, lpData
As Long)
  Public Function FILE_WriteInt(ByVal FileHandle, ByVal ByteCount As Long, lpData As
Integer)
  Public Function FILE_WriteByte(ByVal FileHandle, ByVal ByteCount As Long, lpData
As Byte)
```

引数	説明
FileHandle	FILE_Openによって返されたファイルハンドル。
ByteCount	書き込むバイト数。

lpData	書き込むデータへのポインタ。データを配列を書き込みたい場合は、この配列の0番目の要素Arr(0)を指定します。
	例:
	`Dim IntVariable As Integer`
	`Dim DoubleVariable As Double`
	`Dim DoubleArray(100) As Double`
	`Dim ByteVariable As Byte`
	`Dim ByteArray(1000) As Byte`
	
	`FILE_WriteByteIn fhnd, 2, IntVariable`
	`FILE_WriteDouble fhnd, 8, DoubleVariable`
	`FILE_WriteDouble fhnd, 8*100, DoubleArray(0)`
	`FILE_WriteByte fhnd, 1, ByteVariable`
	`FILE_WriteByte fhnd, 1000, ByteArray(0)`

戻り値
書き込まれたバイト数。

参照

FILE_Open、FILE_Close、FILE_Exists、FILE_SetFilePointer、FILE_Read

EA_Comm.FTP_DownloadFile

ファイルとフォルダをFTP/SFTPサーバからローカルディレクトリにダウンロードします。

⚠ セキュリティ情報: データの悪用や損失を避けるため、FTPアップロードのアカウント情報を含むプロジェクトを誰にも提供しないでください。ただし、その人が本当に本人であり、その情報にアクセスできるはずであると確信できる場合を除きます。

```
  EA_Comm.FTP_DownloadFile(ByVal SourceName As String, ByVal TargetDir As String) As
Long
```

引数	説明
SourceName	サーバ上のソースファイルまたはフォルダを指定する絶対パスまたは相対パス(作業ディレクトリからの)。ディレクトリ区切り文字として"/"を使用します。ディレクトリ内のすべてのファイルとサブディレクトリをダウンロードするには、パスの後ろに"/*"を書き込みます。 FTP/SFTPサーバによっては、名前の大文字と小文字が区別される場合があります。
TargetDir	取得したファイルまたはフォルダーを保存する宛先ディレクトリを指定するパス。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- FTP/SFTPサーバで大文字と小文字を区別する必要がある場合、関数に渡すソース名がサーバ上の名前と正確に一致していることを確認してください。
- ダウンロード前にEA_Comm.FTP_Initを呼び出して接続設定を準備し、EA_Comm.FTP_Loginを呼び出してサーバに接続します。
- ファイル転送にFTPアップロード機能を使用するには、EasyMonitoringモジュールまたはcatmanAPのライセンスが必要です。
- FTPアップロードのサンプル: EA_Comm.FTP_Init

例

```
'----------FTP download----------
Sub Main

  Dim retVal As Long

  'Initialize FTP server with login data.
  EA_Comm.FTP_Init "ftp://catmantest.hbm.com", "catman", "catman", 1, 21

' Connect to FTP server.
  retVal = EA_Comm.FTP_LogIn
  If retVal = 0 Then

    'Set working directory on FTP server and download file by specifying the file name
only.
    retVal = EA_Comm.FTP_SetWorkingDirectory("/catman_Test/FTP_Test")
    retVal = EA_Comm.FTP_DownloadFile("testfile1.bin", "C:\HBM\catmanEasy\DATA")
```

```
    'Or download file by specifying the absolute file path on the FTP server.
    retVal = EA_Comm.FTP_DownloadFile("/catman_Test/FTP_Test/testfile2.bin", "C:\HBM\
catmanEasy\DATA")

    'Download all files and subdirectories in the specified directory on the FTP
server.
    retVal = EA_Comm.FTP_DownloadFile("/catman_Test/FTP_Test/road_test_folder/*",
"C:\HBM\catmanEasy\DATA\Road_Test")

    'Download all .bin files that match the search pattern in the specified directory
on the FTP server.
    retVal = EA_Comm.FTP_DownloadFile("/catman_Test/FTP_Test/road_test_folder/*.bin",
"C:\HBM\catmanEasy\DATA\Road_Test_Copy")

    'Upload file to the FTP server.
    retVal = EA_Comm.FTP_UploadFile("C:\HBM\catmanEasy\DATA\testfile3.bin", "/catman_
Test/FTP_Test")

    'Disconnect from FTP server.
    EA_Comm.FTP_LogOut

  End If

End Sub
```

EA_Comm.FTP_Init

FTP/SFTP接続を初期化します。

⚠ セキュリティ情報: データの悪用や損失を避けるため、FTPアップロードのアカウント情報を含むプロジェクトを誰にも提供しないでください。ただし、その人が本当に本人であり、その情報にアクセスできるはずであると確信できる場合を除きます。

```
EA_Comm.FTP_Init(ByVal Server As String, ByVal User As String, ByVal Password As String, Optional ByVal PassiveMode As Integer, Optional ByVal Port As Long, Optional ByVal AuthType As Integer, Optional ByVal PEM As String, Optional ByVal Passphrase As String, Optional ByVal ServerType As Integer, Optional ByVal Zip As Integer) As Long
```

引数	説明
Server	FTP/SFTPサーバ名。
User	FTP/SFTPサーバ上のアカウントのユーザ名。
Password	FTP/SFTPサーバ上のユーザに関連付けられているアカウントのパスワード。
PassiveMode	オプション。FTPを使用する場合にのみ必要です。FTPデータ接続を確立する方法を決定します(0 = アクティブモード、1 = パッシブモード)。
Port	オプション。接続先のリモート ポート。 FTP接続のデフォルトポート: 21 SFTP接続のデフォルトのポート: 22
AuthType	オプション。SFTPを使用する場合にのみ必要です。SFTP認証方式のタイプ(0 = パスワード認証、1 = 公開キー認証)。 省略した場合は、デフォルトでパスワード認証が使用されます。
PEM	オプション。SFTP公開キー認証用のPEM秘密キー。
Passphrase	オプション。SFTP PEM秘密キーを暗号化するためのパスフレーズ。
ServerType	オプション。サーバの種類(0 = FTP、それ以外の場合はSFTP)。 省略した場合、デフォルトでFTPが使用されます。
Zip	オプション。ローカルファイルをZIPアーカイブに圧縮します(0 = zip圧縮をしない。それ以外の場合はzipを使用します)。 省略した場合、デフォルトではファイルは圧縮されません。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- FTP/SFTPサーバの設定は、DAQプロジェクト内のすべてのDAQジョブとレコーダに適用されます。
- ファイル転送にFTPアップロード機能を使用するには、EasyMonitoringモジュールまたはcatmanAPライセンスが必要です。

例

```
'----------FTP upload----------
Sub Main
  Dim retVal As Long
  Dim progress As Double
  Dim idxFile As Integer
  Dim nofFiles As Integer
  Dim fileName As String
  Dim bytesWritten As Variant
  Dim bytesTotal As Variant
```

```
  'Initialize FTP server with login data.
  EA_Comm.FTP_Init "ftp://catmantest.hbm.com", "catman", "catman", 1, 21

  'Use proxy server, Type: SOCKS V4 proxy.
  'EA_Comm.FTP_InitProxy 1, "168.192.1.60", "1080", "", ""

  'Connect to FTP server.
  EA_Comm.FTP_LogIn

  'Upload local file to FTP server.
  EA_Comm.FTP_UploadFile "C:\testfile.bin", "/catman_Test/FTP_Test", 1

  'Wait until upload is finished.
  retVal=1
  While retVal = 1
    retVal = EA_Comm.FTP_UploadStatus(progress, idxFile, nofFiles, fileName,
bytesWritten, bytesTotal)
    Debug.Print progress
    DoEvents
  Wend

  'Disconnect from FTP server.
  EA_Comm.FTP_LogOut
End Sub

'----------SFTP upload using public-key authentication----------
Sub Main
  Dim retVal As Long
  Dim progress As Double
  Dim idxFile As Integer
  Dim nofFiles As Integer
  Dim fileName As String
  Dim bytesWritten As Variant
  Dim bytesTotal As Variant
  Dim privateKey As String
  Dim f As Integer
  Dim s As String

  'Read private key from key file.
  f = FreeFile
  Open "C:\PrivateKey_PEM.ppk" For Input As #f
  While EOF(f) = 0
    Line Input #f, s
    privateKey = privateKey & s
  Wend
  Close #f

  'Initialize SFTP server with login data.
  EA_Comm.FTP_Init "168.192.1.20", "catman", "", , 22, 1, privateKey, "catman", 1

  'Connect to SFTP server.
  EA_Comm.FTP_LogIn

 'Upload local file to SFTP server.
  EA_Comm.FTP_UploadFile "C:\testfile.bin", "/catman_Test/FTP_Test", 1

  'Wait until upload is finished.
  retVal=1
```

```
  While retVal = 1
    retVal = EA_Comm.FTP_UploadStatus(progress, idxFile, nofFiles, fileName,
bytesWritten, bytesTotal)
    Debug.Print progress
    DoEvents
  Wend

  'Disconnect from SFTP server.
  EA_Comm.FTP_LogOut
End Sub

'----------SFTP upload using password authentication----------
Sub Main
  Dim retVal As Long
  Dim progress As Double
  Dim idxFile As Integer
  Dim nofFiles As Integer
  Dim fileName As String
  Dim bytesWritten As Variant
  Dim bytesTotal As Variant

'Initialize SFTP server with login data.
  EA_Comm.FTP_Init "168.192.1.20", "catman", "catman", , 22, 0, , , 1

  'Connect to SFTP server.
  EA_Comm.FTP_LogIn

  'Upload local file to SFTP server.
  EA_Comm.FTP_UploadFile "C:\testfile.bin", "/catman_Test/FTP_Test", 1

  'Wait until upload is finished.
  retVal=1
  While retVal = 1
    retVal = EA_Comm.FTP_UploadStatus(progress, idxFile, nofFiles, fileName,
bytesWritten, bytesTotal)
    Debug.Print progress
    DoEvents
  Wend

  'Disconnect from SFTP server.
  EA_Comm.FTP_LogOut
End Sub
```

EA_Comm.FTP_InitProxy

FTP/SFTP接続用のプロキシサーバを初期化します。

⚠ セキュリティ情報: データの悪用や損失を避けるため、FTPアップロードのアカウント情報を含むプロジェクトを誰にも提供しないでください。ただし、その人が本当に本人であり、その情報にアクセスできるはずであると確信できる場合を除きます。

```
  EA_Comm.FTP_InitProxy(ByVal ProxyType As Integer, ByVal ProxyHost As String, ByVal
ProxyPort As String, ByVal ProxyUser As String, ByVal ProxyPassword As String) As
Long
```

引数	説明
ProxyType	FTP/SFTP接続に使用されるプロキシサーバのタイプ。 FTP接続の場合: 0 = プロキシサーバーは使用しません 1 = SOCKS V4 2 = SOCKS V5 3 = サイトホスト 4 = ログイン後 5 = ログインなし 6 = OPENプロキシ 7 = PIPEプロキシ SFTP接続の場合: 0 = プロキシサーバーは使用しません 1 = HTTPプロキシ
ProxyHost	リクエストがルーティングされるプロキシサーバの名前またはアドレス。
ProxyPort	リクエストがルーティングされるプロキシサーバが使用するポート。 FTP接続のデフォルトポート: 1080 SFTP接続のデフォルトポート: 6588
ProxyUser	認証を必要とするプロキシサーバ経由のすべてのリクエストで送信されるユーザ名。
ProxyPassword	認証を必要とするプロキシサーバ経由のすべてのリクエストで送信されるパスワード。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- FTP/SFTPサーバの設定は、DAQプロジェクト内のすべてのDAQジョブとレコーダに適用されます。
- ファイル転送にFTPアップロード機能を使用するには、EasyMonitoringモジュールまたはcatmanAPライセンスが必要です。
- FTPアップロードのサンプル: EA_Comm.FTP_Init

EA_Comm.FTP_LastError

FTP/SFTPアップロード中に発生した最後のエラー番号を返します。

⚠ セキュリティ情報: データの悪用や損失を避けるため、FTPアップロードのアカウント情報を含むプロジェクトを誰にも提供しないでください。ただし、その人が本当に本人であり、その情報にアクセスできるはずであると確信できる場合を除きます。

```
Property EA_Comm.FTP_LastError
```

戻り値	説明
0	エラーなし。
<>0	エラー番号。

ℹ 参照: EA_Comm.FTP_LastErrorDescription

EA_Comm.FTP_LastErrorDescription

FTP/SFTPアップロード中に発生した最後のエラーの説明を返します。

セキュリティ情報: データの悪用や損失を避けるため、FTPアップロードのアカウント情報を含むプロジェクトを誰にも提供しないでください。ただし、その人が本当に本人であり、その情報にアクセスできるはずであると確信できる場合を除きます。

```
Property EA_Comm.FTP_LastErrorDescription
```

戻り値
エラーが発生した場合は最後のエラーの説明テキスト。それ以外の場合は、空の文字列を返します。

参照: EA_Comm.FTP_LastError

EA_Comm.FTP_LogIn

FTP/SFTPサーバにログインして接続します。

⚠ セキュリティ情報: データの悪用や損失を避けるため、FTPアップロードのアカウント情報を含むプロジェクトを誰にも提供しないでください。ただし、その人が本当に本人であり、その情報にアクセスできるはずであると確信できる場合を除きます。

```
EA_Comm.FTP_LogIn() As Long
```

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- ログイン前にEA_Comm.FTP_Initを使用して接続設定を準備します。
- ファイル転送にFTPアップロード機能を使用するには、EasyMonitoringモジュールまたはcatmanAPライセンスが必要です。
- FTPアップロードのサンプル: EA_Comm.FTP_Init

EA_Comm.FTP_LogOut

ログオフし、FTP/SFTPサーバへの接続を終了します。

セキュリティ情報: データの悪用や損失を避けるため、FTPアップロードのアカウント情報を含むプロジェクトを誰にも提供しないでください。ただし、その人が本当に本人であり、その情報にアクセスできるはずであると確信できる場合を除きます。

```
EA_Comm.FTP_LogOut() As Long
```

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- ファイル転送にFTPアップロード機能を使用するには、EasyMonitoringモジュールまたはcatmanAPライセンスが必要です。

- FTPアップロードのサンプル: EA_Comm.FTP_Init

EA_Comm.FTP_PauseUpload

FTP/SFTPサーバへのファイルのアップロードを一時停止します。

⚠ セキュリティ情報: データの悪用や損失を避けるため、FTPアップロードのアカウント情報を含むプロジェクトを誰にも提供しないでください。ただし、その人が本当に本人であり、その情報にアクセスできるはずであると確信できる場合を除きます。

```
EA_Comm.FTP_PauseUpload() As Long
```

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- ファイル転送にFTPアップロード機能を使用するには、EasyMonitoringモジュールまたはcatmanAPライセンスが必要です。
- アップロードを再開するには、EA_Comm.FTP_ResumeUploadを使用します。
- FTPアップロードのサンプル: EA_Comm.FTP_Init

EA_Comm.FTP_ResumeUpload

FTP/SFTPサーバへのファイルのアップロードを再開します。

セキュリティ情報: データの悪用や損失を避けるため、FTPアップロードのアカウント情報を含むプロジェクトを誰にも提供しないでください。ただし、その人が本当に本人であり、その情報にアクセスできるはずであると確信できる場合を除きます。

```
EA_Comm.FTP_ResumeUpload() As Long
```

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- ファイル転送にFTPアップロード機能を使用するには、EasyMonitoringモジュールまたはcatmanAPライセンスが必要です。
- アップロードを一時停止するには、EA_Comm.FTP_PauseUploadを使用します。
- FTPアップロードのサンプル: EA_Comm.FTP_Init

EA_Comm.FTP_SetWorkingDirectory

FTP/SFTPサーバ上の作業ディレクトリを設定します。

⚠ セキュリティ情報: データの悪用や損失を避けるため、FTPアップロードのアカウント情報を含むプロジェクトを誰にも提供しないでください。ただし、その人が本当に本人であり、その情報にアクセスできるはずであると確信できる場合を除きます。

```
EA_Comm.FTP_SetWorkingDirectory(ByVal DirName As String) As Long
```

引数	説明
DirName	FTP/SFTP サーバ上の目的のディレクトリを指定するパス。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- ファイル転送にFTPアップロード機能を使用するには、EasyMonitoringモジュールまたはcatmanAPライセンスが必要です。
- FTPアップロードのサンプル: EA_Comm.FTP_Init

EA_Comm.FTP_UploadFile

ローカルファイルをFTP/SFTPサーバにアップロードします。

⚠ セキュリティ情報: データの悪用や損失を避けるため、FTPアップロードのアカウント情報を含むプロジェクトを誰にも提供しないでください。ただし、その人が本当に本人であり、その情報にアクセスできるはずであると確信できる場合を除きます。

```
  EA_Comm.FTP_UploadFile(ByVal FileName As String, Optional ByVal TargetDir As
String, Optional ByVal DeleteOriginalFile As Integer) As Long
```

引数	説明
FileName	アップロードするローカルファイルの名前。
TargetDir	オプション。ファイルがアップロードされるFTP/SFTP上の作業ディレクトリ。省略した場合は、現在の作業ディレクトリが使用されます。
DeleteOriginalFile	オプション。アップロード後にローカルファイルシステムからソースファイルを削除します(0 = 削除しません。それ以外の場合は削除します)。省略した場合、ソースファイルは削除されません。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- EA_Comm.FTP_Initを呼び出して接続設定を準備し、EA_Comm.FTP_Loginを呼び出してアップロード前にサーバに接続します。
- 現在のアップロードステータスを取得するには、EA_Comm.FTP_UploadStatusメソッドを使用します。
- ファイル転送にFTPアップロード機能を使用するには、EasyMonitoringモジュールまたはcatmanAPライセンスが必要です。
- FTPアップロードのサンプル: EA_Comm.FTP_Init

EA_Comm.FTP_UploadStatus

FTP/SFTPアップロードのステータスを取得します。

⚠ セキュリティ情報: データの悪用や損失を避けるため、FTPアップロードのアカウント情報を含むプロジェクトを誰にも提供しないでください。ただし、その人が本当に本人であり、その情報にアクセスできるはずであると確信できる場合を除きます。

```
  EA_Comm.FTP_UploadStatus(Optional progressValue As Double, Optional ixFile
As Integer, Optional NofFiles As Integer, Optional FileName As String, Optional
BytesWritten As Variant, Optional BytesTotal As Variant) As Long
```

引数	説明
progressValue	コピーしているファイルの転送したバイトの割合(0 ～ 100)。
ixFile	現在コピーしているファイルのインデックス。
NofFiles	アップロードするファイルの総数。
FileName	現在コピーしているファイルの名前。
BytesWritten	これまでに転送されたバイト数。
BytesTotal	転送するファイルの合計バイト数。

戻り値	説明
0	アップロード完了。
1	アップロード実行中。
<0	エラー。

注記

- EA_Comm.FTP_Initを呼び出して接続設定を準備し、EA_Comm.FTP_Loginを呼び出してアップロード前にサーバに接続します。
- 現在のアップロードステータスを取得するには、EA_Comm.FTP_UploadStatusメソッドを使用します。
- ファイル転送にFTPアップロード機能を使用するには、EasyMonitoringモジュールまたはcatmanAPライセンスが必要です。
- FTPアップロードのサンプル: EA_Comm.FTP_Init

EA_Comm.GPIB_ClosePort

既存のGPIB通信を閉じます。

```
EA_Comm.GPIB_ClosePort() As Long
```

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.GPIB_OpenPort

GPIB(IEEE488)を開き、デバイスと通信します。

```
  EA_Comm.GPIB_OpenPort(ByVal EOIOnWrite As Integer, ByVal StopReadOnEOS As Integer,
ByVal SetEOIOnEOS As Integer, ByVal EOSByte As Byte, ByVal RepeatAddressing As
Integer, ByVal Timeout As Long) As Long
```

引数	説明
EOIOnWrite	EOI送信: 1 = 最後の文字とともにEOIを送信(デフォルト) 0 = EOIを送信しない
StopReadOnEOS	EOS検出: 1 = EOS検出時点で読み取り終了 0 = 読み取りを終了しない(デフォルト)
SetEOIOnEOS	EOI設定: 1 = EOS受信時にEOIを設定 0 = EOIを設定しない(デフォルト)
EOSByte	バイト数: 0..255
RepeatAddressing	リピートアドレッシング: 1 = Yes (デフォルト) 0 = No
Timeout	ミリ秒単位のタイムアウト(通常は5000)。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.GPIB_ReadPort

GPIB接続を開き、EA_Comm.GPIB_SelectAddressで選択したデバイスから文字を読み取ります。

```
  EA_Comm.GPIB_ReadPort(ByVal Terminator As Integer, ByVal MaxChars As Integer,
RcvBuf As String) As Long
```

引数	説明
Terminator	読み取り動作を停止するASCIIコード(例えば、改行文字ですべての応答を終了する場合は10)。終端文字は文字列の受信後に削除してください(例えばReplace$関数で)
MaxChars	終端文字に関係なく読み取る最大文字数。
RcvBuf	受信文字列。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- 終端文字に関係なく、指定した文字数を読み取りたい場合、*Terminator*を0に設定します。

EA_Comm.GPIB_ReadPortBinary

GPIB接続を開き、EA_Comm.GPIB_SelectAddressで選択したデバイスから指定したバイト数を読み取ります。

```
EA_Comm.GPIB_ReadPortBinary(ByVal MaxBytes As Long, RcvBuf() As Byte) As Long
```

引数	説明
MaxBytes	読み取るバイト数。
RcvBuf	受信したバイト配列。配列のサイズは少なくとも*MaxBytes*を呼び出し側で割り当てる必要があります。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.GPIB_SelectAddress

以降の読み取り/書き込み操作のため、GPIBデバイスを選択します。

```
EA_Comm.GPIB_SelectAddress(ByVal Address As Integer) As Long
```

引数	説明
Address	アドレス。 1...32

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.GPIB_WritePort

EA_Comm.GPIB_SelectAddressで選択したデバイスにコマンドを送信します。

```
EA_Comm.GPIB_WritePort(ByVal Cmd As String) As Long
```

引数	説明
Cmd	コマンド文字列。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- このメソッドのの使用には、デバイスでサポートしている構文の正確な知識が必要です。ほとんどのデバイスはコマンドの終わりを示す終端文字(例えば改行文字)が必要であることに注意してください。

  例:

  ```
  EA_Comm.GPIB_WritePort "IDN?" & vbLf
  ```

- DAQジョブの実行中は、DAQ操作を行っているデバイスに、任意のコマンドを送るべきではありません。

EA_Comm.INET_Cancel

現在のリクエストをキャンセルし、現在確立されている接続をすべて閉じます。

```
EA_Comm.INET_Cancel()
```

EA_Comm.INET_Execute

リモートサーバーへのリクエストを実行します。特定のプロトコルに対して有効なリクエストのみを送信できます。

```
  EA_Comm.INET_Execute(ByVal URL As String, ByVal Command As String, ByVal InputData
As String, ByVal RequestHeader As String) As Long
```

引数	説明
URL	コントロールの接続先となるURLを指定する文字列。ここでURLが指定されていない場合は、EA_Comm.INET_LogInメソッドで指定されたURLが使用されます。
Command	実行する操作の種類を指定する文字列。以下の操作がサポートされています:
	サポートされているHTTPコマンド: GET: URLプロパティで指定された URLからデータを取得します。 HEAD: リクエストヘッダを送信します。 POST: データをサーバに送信します。データは、*InputData*引数にあります。これはGETの代替メソッドであり、*InputData*引数で追加の命令が指定されます。 PUT: プット操作。置換されるページの名前は、*InputData*引数にあります。
	サポートされているFTPコマンド: CD file1: ディレクトリを変更します。file1で指定したディレクトリに移動します。 CDUP: 親ディレクトリに変更します。"CD.."に相当します。 CLOSE: 現在のFTP接続を閉じます。 DELETE file1: file1で指定したファイルを削除します。 DIR file1: ディレクトリ。file1で指定したディレクトリを検索します(ワイルドカードは許可されますが、構文はリモートホストによって決まります)。file1が指定されていない場合は、現在の作業ディレクトリの完全なディレクトリが返されます。 GET file1 file2: file1で指定したリモートファイルを取得し、file2で指定した新しいローカルファイルを作成します。 LS file1: リスト。file1で指定したディレクトリを検索します(ワイルドカードは許可されますが、構文はリモート ホストによって決まります)。GetChunkメソッドを使用して、ファイル ディレクトリデータを返します。 MKDIR file1: ディレクトリを作成します。file1で指定したディレクトリを作成します。成功するかどうかは、リモート ホスト上のユーザ権限に依存します。 PUT file1 file2: file1で指定したローカルファイルをfile2で指定そたリモートホストにコピーします。 PWD: 作業ディレクトリを出力します。現在のディレクトリ名を返します。GetChunk メソッドを使用してデータを返します。 QUIT: 現在のユーザを終了します。 RECV file1 file2: file1で指定したリモートファイルを取得し、file2で指定した新しいローカルファイルを作成します。GETと同等です。 RENAME file1 file2: file1で指定したリモート ファイルの名前を、file2 で指定された新しい名前に変更します。成功するかどうかは、リモート ホスト上のユーザー権限に依存します。 RMDIR file1: ディレクトリを削除します。file1で指定したリモートディレクトリを削除します。成功するかどうかは、リモートホスト上のユーザ権限に依存します。 SEND file1 file2: file1で指定したローカルファイルをfile2で指定したリモートホストにコピーします。PUTと同等です。 SIZE file1: file1で指定したディレクトリのサイズを返します。

InputData	操作のためにデータを指定する文字列。
RequestHeader	リモートサーバから送信される追加ヘッダを指定する文字列。これらの形式は次のとおりです: header name: header value vbCrLf

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.INET_GetData

データを取得します。

```
  EA_Comm.INET_GetData(ByVal Format As Integer, bBuf() As Byte, sBuf As String,
NofBytes As Long)
```

引数	説明
Format	データのタイプを指定する整数。 0 = icString: デフォルト。データを文字列として取得します。 1 = IcByteArray: データをバイト配列として取得します。
bBuf	データ型IcByteArray(1)が使用されている場合、バイナリとして取得されるデータ。
sBuf	データ型icString(0)が使用されている場合、文字列として取得されるデータ。
NofBytes	取得するデータのサイズ。

EA_Comm.INET_GetState

現在の接続状態を取得します。

```
EA_Comm.INET_GetState() As Long
```

戻り値	説明
0	報告するステータスはありません。
1	コントロールは、指定されたホストコンピュータのIPアドレスを検索しています。
2	コントロールは、指定されたホストコンピュータのIPアドレスを正常に検出しました。
3	コントロールはホストコンピュータに接続中です。
4	コントロールはホストコンピュータに正常に接続しました。
5	コントロールはホストコンピュータにリクエストを送信中です。
6	コントロールはリクエストを正常に送信しました。
7	コントロールはホストコンピュータからの応答を受信中です。
8	コントロールはホストコンピュータからの応答を正常に受信しました。
9	コントロールがホストコンピュータから切断中です。
10	コントロールはホストコンピュータから正常に切断しました。
11	ホストコンピュータとの通信でエラーが発生しました。
12	リクエストが完了し、すべてのデータを受信しました。

EA_Comm.INET_LogIn

EA_Comm.INET_Executeメソッドで使用されるURL、ユーザのログオン名とパスワードを設定します。

```
  EA_Comm.INET_LogIn(ByVal URL As String, ByVal UserName As String, ByVal Password
As String) As Long
```

引数	説明
URL	EA_Comm.INET_Executeメソッドで使用するURLを指定します。
UserName	ユーザのログオン名。
Password	ユーザのパスワード。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.INFLUX_Connect

catman EasyScriptのバージョン5.2以降では、時系列データベース システムとして代表的な**Influxdb**のサポートを導入しています。Influxdbデータベースにデータを書き込むと、これらのデータをGrafanaなどのさまざまなWebベースのダッシュボードシステムで利用できるようになります。catmanのスクリプトメソッドは、ローカルネットワークまたはインターネットに存在するInfluxdbデータベースのサーバにアクセスします。作成したすべてのデータベースと書き込むすべてのデータがサーバ上に存在することを確認してください!

スクリプトの作成者は、*Influxdb*データベースの概念に精通していることが求められます。*Measurement*、*Tags*、*Timestamps*などの用語を理解する必要があります。このヘルプトピックでは、catmanスクリプトのメソッドについてのみ説明します。*Influxdb*に関する概要とリファレンスについては、"https://docs.influxdata.com"を参照してください。

```
  Public Function INFLUX_Connect(ByVal ServerAddr As String, ByVal ServerPort As
Long, Optional ByVal User As String, Optional ByVal Password As String) As Long
```

この関数は、他の`INFLUX_....`メソッドよりも前に呼び出す必要があります。これは、Influxdbサーバに関する接続の詳細と、必要に応じてユーザの承認をcatmanに伝えます。Influxdbサーバの構成によっては、ユーザのログインが必要になる場合があります。そうでない場合、パラメータ"User"および"Password"を省略できます。

引数	説明
`ServerAddr`	InfluxdbサーバのIPアドレス。あるいは、"influxdata.com"などのURLを指定することもできます。 URLの前に`HTTP://`を付けないでください。これはcatmanが自動的に行います。
`ServerPort`	サーバポート。デフォルトは`8086`です。
`User`	ユーザー名。サーバから承認したアクセスを期待している場合にのみ必要です。
`Password`	パスワード。サーバから承認したアクセスを期待している場合にのみ必要です。

戻り値	説明
`0`	エラーなし。
`<0`	エラー。

EA_Comm.INFLUX_CreateDatabase

catman EasyScriptのバージョン5.2以降では、時系列データベース システムとして代表的な**Influxdb**のサポートを導入しています。Influxdbデータベースにデータを書き込むと、これらのデータをGrafanaなどのさまざまなWebベースのダッシュボードシステムで利用できるようになります。catmanのスクリプトメソッドは、ローカルネットワークまたはインターネットに存在するInfluxdbデータベースのサーバにアクセスします。作成したすべてのデータベースと書き込むすべてのデータがサーバ上に存在することを確認してください!

スクリプトの作成者は、*Influxdb*データベースの概念に精通していることが求められます。*Measurement*、*Tags*、*Timestamps*などの用語を理解する必要があります。このヘルプトピックでは、catmanスクリプトのメソッドについてのみ説明します。*Influxdb*に関する概要とリファレンスについては、"https://docs.influxdata.com"を参照してください。

```
  Public Function INFLUX_CreateDatabase(ByVal DBName As String, Response As String)
As Long
```

サーバ上に新しいデータベースを作成します。

引数	説明
DBName	新しいデータベースの名前。
Response	サーバの応答。成功した場合、応答が空であることに注意してください!

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.INFLUX_Query

catman EasyScriptのバージョン5.2以降では、時系列データベース システムとして代表的な**Influxdb**のサポートを導入しています。Influxdbデータベースにデータを書き込むと、これらのデータをGrafanaなどのさまざまなWebベースのダッシュボードシステムで利用できるようになります。catmanのスクリプトメソッドは、ローカルネットワークまたはインターネットに存在するInfluxdbデータベースのサーバにアクセスします。

スクリプトの作成者は、*Influxdb*データベースの概念に精通していることが求められます。*Measurement*、*Tags*、*Timestamps*などの用語を理解する必要があります。このヘルプトピックでは、catmanスクリプトのメソッドについてのみ説明します。*Influxdb*に関する概要とリファレンスについては、"https://docs.influxdata.com"を参照してください。

```
  Public Function INFLUX_Query(ByVal DBName As String, ByVal Query As String,
Response As String) As Long
```

Influxdbデータベースからデータをクエリします。

引数	説明
DBName	新しいデータベースの名前。
Query	クエリは、データベースの検索に適用するフィルタの文字列表現であり、InfluxDBクエリ言語(InfluxQL)構文を持ちます。InfluxQLの正式な説明の詳細な仕様を確認するには、influxdbのページにアクセスしてください。
Response	サーバの応答。書き込みが成功した場合、応答が空であることに注意してください!

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.INFLUX_WriteField

catman EasyScriptのバージョン5.2以降では、時系列データベース システムとして代表的な**Influxdb**のサポートを導入しています。Influxdbデータベースにデータを書き込むと、これらのデータをGrafanaなどのさまざまなWebベースのダッシュボードシステムで利用できるようになります。catmanのスクリプトメソッドは、ローカルネットワークまたはインターネットに存在するInfluxdbデータベースのサーバにアクセスします。

スクリプトの作成者は、*Influxdb*データベースの概念に精通していることが求められます。*Measurement*、*Tags*、*Timestamps*などの用語を理解する必要があります。このヘルプトピックでは、catmanスクリプトのメソッドについてのみ説明します。*Influxdb*に関する概要とリファレンスについては、"https://docs.influxdata.com"を参照してください。

```
  Public Function INFLUX_WriteField(ByVal DBName As String, ByVal Measurement As
String, ByVal NofTags As Integer, TagName() As String, TagValue() As Variant, ByVal
FieldName As String, TimeStamp As Double, Value As Double, ByVal Resolution As
Integer, ByVal WaitResponse As Integer, Response As String) As Long
```

単一のフィールドを既存のデータベースに書き込みます。実際のデータポイントに加えて、タグを指定できます。

引数	説明
DBName	既存データベースの名前。
Measurement	*Measurement*はデータベース内のサブグループとして仮定します。*Measurement*の一般的な名前は、catmanのテストファイル名です。*Measurement*がデータベースにまだ存在しない場合は、作成します。
NofTags	タグの数。
TagName()	各タグの名前を含む*NofTags*要素(最初はインデックス0)を含む配列。タグがデータベースにまだ存在しない場合は、作成します。 *TagName*にはスペースを含めることはできません。catmanは自動的にスペースをアンダースコアに置き換えます。他にもいくつかの文字は禁止されています: "="、","、"&"、一重引用符または二重引用符
TagValue()	各タグの値を含む*NofTags*要素を含む配列。 タグの典型的な例は次のとおりです: Channel=Pressure
FieldName	値が割り当てられるフィールド。これは通常、データの取得元のチャネルのチャネル名になります。データが別のソースから取得されたものである場合、(例: スクリプト内の変数の場合)、この名前を*FieldName*として使用します。 FieldNameにはスペースを含めることはできません。catmanは自動的にスペースをアンダースコアに置き換えます。他にもいくつかの文字は禁止されています: "="、","、"&"、一重引用符または二重引用符
TimeStamp	この時点のタイムスタンプ。タイムスタンプは、1.1.1970 00:00:00(Unixエポック)からの秒数が期待されます。catmanは、*Influxdb*サーバ用にそれらをナノ秒に変換します。次の例は、現在の時刻を反映するタイムスタンプを1ミリ秒の分解能で生成する方法を示しています。 `Timestamp=EA_Comm.TIME_GetSystemTime()` タイムスタンプのもう1つのソースは、QuantumXモジュールのハードウェアタイムチャネルです(たとえば、チャネルのすべてのサンプルを転送したい場合)。 💡 TimeStampに-1を指定すると、サーバに到着したときの値がタイムスタンプされます。
Value	ポイント値。catmanは、*Resolution*で設定した解像度(小数点以下の桁数)で値をInfluxdbサーバに送信します。
Resolution	小数点以下の桁数。0の場合、catmanは絶対値のサイズに応じて自動フォーマットを適用します。
WaitResponse	Influxdbサーバからの応答を受信するまでスクリプトの実行を停止する場合は、TRUEに設定します。
Response	サーバの応答。書き込みが成功した場合、応答が空であることに注意してください!

戻り値	説明
0	エラーなし。
<0	エラー。

例

次の例は、新しいデータブロックがハードウェアからフェッチされるたびにチャネルの現在値を転送する方法を示しています(このため、スクリプトは"データ転送後"の自動実行ステージにフックされます)。

```
Sub Main

  Dim Timestamp As Double
  Dim Sample As Double
  Dim TagNames(1) As String
  Dim TagVals(1) As Variant
  Dim Measurement As String
  Dim DBName As String
  Dim ChannelName As String

  DBName="catman Test Database"
  Measurement="TestRun"
  ChannelName="GPS_Speed"
  TagNames(0)="Weather"
  TagVals(0)="Cloudy"
  TagNames(1)="Track"
  TagVals(1)="Wet"

  EA_Comm.INFLUX_Connect "172.19.1.81", 8086

  EA_DAQ.CurrentReading ChannelName, Sample, DAQ_NEWEST

  Timestamp=EA_Comm.TIME_GetSystemTime()

  EA_Comm.INFLUX_WriteField DBName, Measurement, 2 ,TagNames(), TagVals(),
ChannelName, Timestamp, Sample, 3, True, Response

End Sub
```

EA_Comm.INFLUX_WriteFile

catman EasyScriptのバージョン5.2以降では、時系列データベース システムとして代表的な**Influxdb**のサポートを導入しています。Influxdbデータベースにデータを書き込むと、これらのデータをGrafanaなどのさまざまなWebベースのダッシュボードシステムで利用できるようになります。catmanのスクリプトメソッドは、ローカルネットワークまたはインターネットに存在するInfluxdbデータベースのサーバにアクセスします。

スクリプトの作成者は、*Influxdb*データベースの概念に精通していることが求められます。*Measurement*、*Tags*、*Timestamps*などの用語を理解する必要があります。このヘルプトピックでは、catmanスクリプトのメソッドについてのみ説明します。*Influxdb*に関する概要とリファレンスについては、"https://docs.influxdata.com"を参照してください。

```
  Public Function INFLUX_WriteFile(ByVal DBName As String, ByVal FileName As String,
Response As String) As Long
```

この関数は、ファイルに含まれる複数行を一度に書き込みます。行には、有効なInfluxdb形式を含めることができます。この関数は、サーバが応答した後にのみ返されます。

引数	説明
DBName	既存データベースの名前。

FileName	値を含むファイルのファイル名(パスを含む)。このファイルは、LFで終わる値行を含むプレーンテキストファイルです。
	ファイル内容の例:
	`mymeas,mytag1=1 value=21 1463689680000000000`
	`mymeas,mytag1=1 value=34 1463689690000000000`
	`mymeas,mytag2=8 value=78 1463689700000000000`
	`mymeas,mytag3=9 value=89 1463689710000000000`

Response	サーバの応答。書き込みが成功した場合、応答が空であることに注意してください!

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.INFLUX_WriteMultipleFields

catman EasyScriptのバージョン5.2以降では、時系列データベース システムとして代表的な**Influxdb**のサポートを導入しています。Influxdbデータベースにデータを書き込むと、これらのデータをGrafanaなどのさまざまなWebベースのダッシュボードシステムで利用できるようになります。catmanのスクリプトメソッドは、ローカルネットワークまたはインターネットに存在するInfluxdbデータベースのサーバにアクセスします。

スクリプトの作成者は、*Influxdb*データベースの概念に精通していることが求められます。*Measurement*、*Tags*、*Timestamps*などの用語を理解する必要があります。このヘルプトピックでは、catmanスクリプトのメソッドについてのみ説明します。*Influxdb*に関する概要とリファレンスについては、"https://docs.influxdata.com"を参照してください。

```
  Public Function INFLUX_WriteMultipleFields(ByVal DBName As String, ByVal
Measurement As String, ByVal NofTags As Integer, TagName() As String, TagValue() As
Variant, ByVal NofFields As Integer, FieldNames() As String, TimeStamp As Double,
Values() As Double, ByVal Resolution As Integer, ByVal WaitResponse As Integer,
Response As String) As Long
```

同じタイムスタンプに属する複数のフィールドを一度に書き込みます。実際のデータポイントに加えて、タグを指定できます。

"field"がcatmanチャンネルであると仮定します。つまり、このメソッドを使用すると、多くのチャネルに1つの値を一度に書き込むことができます。もちろん、フィールドは別のものになる(スクリプト内の変数)可能性もあります。

引数	説明
DBName	既存データベースの名前。
Measurement	*Measurement*はデータベース内のサブグループとして仮定します。*Measurement*の一般的な名前は、catmanのテストファイル名です。*Measurement*がデータベースにまだ存在しない場合は、作成します。
NofTags	タグの数。
TagName()	NofTags要素(最初はインデックス0)を含むタグ名の配列。タグがデータベースにまだ存在しない場合は作成します。 最も一般的なタグはチャネル名であり、通常、チャネルグループに基づいてデータにアクセスする場合には常に必要になります。
TagValue()	各タグの値を含むNofTags要素を含む配列。 タグの典型的な例は次のとおりです: Channel=Pressure
NofFields	データベースに書き込む値の数。各値とタグにより、*DBName/Measurement*に新しいレコードが生成されます。
FieldName	NofPoints要素を含むフィールド名の配列(最初はインデックス0)。 値が割り当てられるフィールド。これは通常、データの取得元のチャネルのチャネル名になります。データが別のソースから取得されたものである場合、(例: スクリプト内の変数の場合)、この名前を*FieldName*として使用します。 FieldNameにはスペースを含めることはできません。catmanは自動的にスペースをアンダースコアに置き換えます。他にもいくつかの文字は禁止されています: "="、","、"&"、一重引用符または二重引用符
TimeStamp	この書き込み操作のタイムスタンプ。タイムスタンプは、1.1.1970 00:00:00(Unixエポック)からの秒数が期待されます。catmanは、*Influxdb*サーバ用にそれらをナノ秒に変換します。次の例は、現在の時刻を反映するタイムスタンプを1ミリ秒の分解能で生成する方法を示しています。 `Timestamp=EA_Comm.TIME_GetSystemTime()` タイムスタンプのもう1つのソースは、QuantumXモジュールのハードウェアタイムチャネルです(たとえば、チャネルのすべてのサンプルを転送したい場合)。 TimeStampに-1を指定すると、サーバに到着したときの値がタイムスタンプされます。
Values()	各フィールドに属する値を含むNofFields要素の配列(最初はインデックス 0)。catmanは、*Resolution*で設定した解像度(小数点以下の桁数)で値をInfluxdbサーバに送信します。
Resolution	小数点以下の桁数。0の場合、catmanは絶対値のサイズに応じて自動フォーマットを適用します。

WaitResponse	Influxdbサーバからの応答を受信するまでスクリプトの実行を停止する場合は、TRUEに設定します。
Response	サーバの応答。書き込みが成功した場合、応答が空であることに注意してください!

戻り値	説明
0	エラーなし。
<0	エラー。

例

次の例は、新しいデータブロックがハードウェアからフェッチされるたびに複数チャネルの現在値を転送する方法を示しています(このため、スクリプトは"データ転送後"の自動実行ステージにフックされます)。

```
Sub Main

  Dim i As Integer
  Dim Timestamp As Double
  Dim Values(2) As Double
  Dim TagNames(2) As String
  Dim TagVals(2) As Variant
  Dim Measurement As String
  Dim DBName As String
  Dim ChannelNames(2) As String

  DBName="catman Test Database"
  Measurement="TestRun"
  TagNames(0)="Weather"
  TagVals(0)="Cloudy"
  TagNames(1)="Track"
  TagVals(1)="Wet"
  ChannelNames(0)="GPS_Latitude"
  ChannelNames(1)="GPS_Longitude"
  ChannelNames(2)="GPS_Speed"

  EA_Comm.INFLUX_Connect "172.19.1.81", 8086

  For i=0 To 2
    EA_DAQ.CurrentReading ChannelNames(i), Values(i), DAQ_NEWEST
  Next

  Timestamp=EA_Comm.TIME_GetSystemTime()

  EA_Comm.INFLUX_WriteMultipleFields DBName, Measurement, 2 ,TagNames(), TagVals(),
3, ChannelNames(), Timestamp, Values(), 3, True, Response

End Sub
```

EA_Comm.INFLUX_WriteSerie

catman EasyScriptのバージョン5.2以降では、時系列データベース システムとして代表的な**Influxdb**のサポートを導入しています。Influxdbデータベースにデータを書き込むと、これらのデータをGrafanaなどのさまざまなWebベースのダッシュボードシステムで利用できるようになります。catmanのスクリプトメソッドは、ローカルネットワークまたはインターネットに存在するInfluxdbデータベースのサーバにアクセスします。

スクリプトの作成者は、*Influxdb*データベースの概念に精通していることが求められます。*Measurement*、*Tags*、*Timestamps*などの用語を理解する必要があります。このヘルプトピックでは、catmanスクリプトのメソッドについてのみ説明します。*Influxdb*に関する概要とリファレンスについては、"https://docs.influxdata.com"を参照してください。

```
Public Function INFLUX_WriteSeries(ByVal DBName As String, ByVal Measurement As
String, ByVal NofTags As Integer, TagName() As String, TagValue() As Variant, ByVal
FieldName As String, ByVal NofPoints As Long, Timestamps() As Double, Values() As
Double, ByVal Resolution As Integer, Response As String) As Long
```

複数のシリーズを既存のデータベースに一度に書き込みます。実際のデータポイントに加えて、タグを指定できます。

引数	説明
DBName	既存データベースの名前。
Measurement	*Measurement*はデータベース内のサブグループとして仮定します。*Measurement*の一般的な名前は、catmanのテストファイル名です。*Measurement*がデータベースにまだ存在しない場合は、作成します。
NofTags	タグの数。
TagName()	NofTags要素(最初はインデックス0)を含むタグ名の配列。タグがデータベースにまだ存在しない場合は作成します。 最も一般的なタグはチャネル名であり、通常、チャネルグループに基づいてデータにアクセスする場合には常に必要になります。
TagValue()	各タグの値を含むNofTags要素を含む配列。 タグの典型的な例は次のとおりです: Channel=Pressure
FieldName	値が割り当てられるフィールド。これは通常、データの取得元のチャネルのチャネル名になります。データが別のソースから取得されたものである場合、(例: スクリプト内の変数の場合)、この名前を*FieldName*として使用します。 FieldNameにはスペースを含めることはできません。catmanは自動的にスペースをアンダースコアに置き換えます。他にもいくつかの文字は禁止されています: "="、","、"&"、一重引用符または二重引用符
NofPoints	データベースに書き込む値の数。各値とタグにより、*DBName/Measurement*に新しいレコードが生成されます。
TimeStamp	この時点のタイムスタンプ。タイムスタンプは、1.1.1970 00:00:00(Unixエポック)からの秒数が期待されます。catmanは、*Influxdb*サーバ用にそれらをナノ秒に変換します。次の例は、現在の時刻を反映するタイムスタンプを1ミリ秒の分解能で生成する方法を示しています。 `Timestamp=EA_Comm.TIME_GetSystemTime()` タイムスタンプのもう1つのソースは、QuantumXモジュールのハードウェアタイムチャネルです(たとえば、チャネルのすべてのサンプルを転送したい場合)。 💡 TimeStampに-1を指定すると、サーバに到着したときの値がタイムスタンプされます。
Values()	各フィールドに属する値を含む*NofFields*要素の配列(最初はインデックス 0)。catmanは、*Resolution*で設定した解像度(小数点以下の桁数)で値をInfluxdbサーバに送信します。
Resolution	小数点以下の桁数。0の場合、catmanは絶対値のサイズに応じて自動フォーマットを適用します。
Response	サーバの応答。書き込みが成功した場合、応答が空であることに注意してください!

戻り値	説明
0	エラーなし。
<0	エラー。

例

次の例は、新しいデータブロックがハードウェアからフェッチされるたびにチャネルの現在の読み取りブロックを転送する方法を示しています(このため、スクリプトは"データ転送後"の自動実行ステージにフックされます)。タイムスタンプはシステムのタイム チャネルから取得されます。

```
Sub Main

  Dim nSamples As Integer
  Dim Timestamps() As Double
  Dim Values() As Double
  Dim TagNames(2) As String
  Dim TagVals(2) As Variant
  Dim Measurement As String
  Dim DBName As String
  Dim ChannelName As String
  Dim TimeChannel As String

  DBName="catman Test Database"
  Measurement="TestRun"
  ChannelName="GPS_Speed"
  TagNames(0)="Weather"
  TagVals(0)="Cloudy"
  TagNames(1)="Track"
  TagVals(1)="Wet"
  TimeChannel="GPS_Time"

  EA_DAQ.CurrentBlockSize ChannelName, nSamples
  Redim Values(nSamples)
  Redim Timestamps(nSamples)
  EA_DAQ.GetCurrentBlock ChannelName, Values
  EA_DAQ.GetCurrentBlock TimeChannel, Timestamps

  EA_Comm.INFLUX_Connect "172.19.1.81", 8086

  EA_Comm.INFLUX_WriteSeries DBName, Measurement, 2 ,TagNames(), TagVals(),
ChannelName, nSamples, Timestamps(), Values(), 3, Response

End Sub
```

EA_Comm.NTP_GetSyncStatus

NTP(Network Time Protocol)の同期状態を取得します。サポートしているデバイスはHBMのMGCplus、QuantumX、光インテロゲータ、およびPMXです。

```
  EA_Comm.NTP_GetSyncStatus(ByVal Device As Variant, Offset As Single, Jitter As
Single, PRC22 As Integer, ntpqResponse As String)
```

引数	説明
Device	デバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。 EA_IO.DevicePosメソッドを使用して、デバイスの名前、IPアドレス、またはUUIDからこのインデックスを取得します。
Offset	基準時刻とシステムクロック間の差。
Jitter	複数の時間クエリ間のジッタの大きさ。
PRC22	NTPデーモンの状態。MGCplusの同期状態を確認するためにのみ必要です。 0 = 停止 1 = 開始
ntpqResponse	時刻同期品質のNTPの応答。

戻り値	説明
0	エラーなし。
<0	エラー。サポートしていないデバイスなど。

EA_Comm.NTP_StartClient

NTPクライアントのNTPサービスを開始します。

```
  EA_Comm.NTP_StartClient(ByVal Device As Variant, ByVal NTPServerAddr As String) As
Long
```

引数	説明
Device	コマンドの送信先となるデバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。 EA_IO.DevicePosメソッドを使用して、デバイスの名前、IPアドレス、またはUUIDからこのインデックスを取得します。
NTPServerAddr	NTPサーバのIPアドレス。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.NTP_StopClient

NTPクライアントのNTPサービスを停止します。

```
EA_Comm.NTP_StopClient(ByVal Device As Variant) As Long
```

引数	説明
Device	コマンドの送信先となるデバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。 EA_IO.DevicePosメソッドを使用して、デバイスの名前、IPアドレス、またはUUIDからこのインデックスを取得します。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.PTP_GetSyncStatus

PTP (Precision Time Protocol)の同期状態を取得します。サポートしているデバイスはQuantumXファミリのモジュールです。

```
  EA_Comm.PTP_GetSyncStatus(ByVal Device As Variant, Offset As Single, MasterID As
String)
```

引数	説明
Device	コマンドの送信先となるデバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。 EA_IO.DevicePosメソッドを使用して、デバイスの名前、IPアドレス、またはUUIDからこのインデックスを取得します。
Offset	マスタとスレーブ間のクロック差。
MasterID	グランドマスタクロックのクロックID。

戻り値	説明
0	エラーなし。
<0	エラー。サポートしていないデバイス、PTPが現在のタイムソースでないなど。

EA_Comm.RS232_ClosePort

既存のRS-232通信を閉じます。

```
EA_Comm.RS232_ClosePort(ByVal Port As Integer) As Long
```

引数	説明
Port	ポート番号1…4(または列挙型COM1…COM4)。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.RS232_FlushInQ

開いているRS-232通信の受信バッファをクリアします。

```
EA_Comm.RS232_FlushInQ(ByVal Port As Integer) As Long
```

引数	説明
Port	ポート番号1…4(または列挙型COM1…COM4)。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.RS232_OpenPort

RS-232を開き、デバイスと通信します。

```
  EA_Comm.RS232_OpenPort(ByVal Port As Integer, ByVal Baudrate As Long, ByVal Parity
As Integer, ByVal DataBits As Integer, ByVal StopBits As Integer, ByVal RcvBuf As
Long, ByVal Timeout As Long) As Long
```

引数	説明
Port	ポート番号1...4(または列挙型COM1...COM4)。
Baudrate	ボーレート。例えば9600。
Parity	パリティ: 0 = なし、1=奇数、2=偶数
DataBits	データビット。7または8。
StopBits	ストップビット。1または2。
RcvBuf	バイト単位の受信バッファサイズ(一般的には4096)。
Timeout	ミリ秒単位のタイムアウト(一般的には5000)。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.RS232_ReadPort

開いたRS-232接続から文字列を読み取ります。

```
  EA_Comm.RS232_ReadPort(ByVal Port As Integer, ByVal Terminator As Integer, ByVal
MaxChars As Integer, RcvBuf As String) As Long
```

引数	説明
Port	ポート番号1...4(または列挙型COM1...COM4)。
Terminator	読み取り動作を停止するASCIIコード(例えば、改行文字(LF)ですべての応答を終了する場合は10)。終端文字は文字列の受信後に削除してください(例えばReplace$関数で)。
MaxChars	終端文字に関係なく読み取る最大文字数。
RcvBuf	受信文字列。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- 終端文字に関係なく、指定した文字数を読み取りたい場合、*Terminator*を0に設定します。

EA_Comm.RS232_ReadPortBinary

開いたRS-232接続から指定したバイト数を読み取ります。

```
  EA_Comm.RS232_ReadPortBinary(ByVal Port As Integer, ByVal MaxBytes As Long,
RcvBuf() As Byte) As Long
```

引数	説明
Port	ポート番号1…4(または列挙型COM1…COM4)。
MaxBytes	読み取るバイト数。
RcvBuf	要求したバイトを受け取るバイト配列。この配列は呼び出し元によって割り当てる必要があり、少なくともMaxBytesサイズでなければなりません。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.RS232_WritePort

デバイスにコマンドを送信します。

```
EA_Comm.RS232_WritePort(ByVal Port As Integer, ByVal Cmd As String) As Long
```

引数	説明
Port	ポート番号1…4(または列挙型COM1…COM4)。
Cmd	コマンド文字列。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- このメソッドのの使用には、デバイスでサポートしている構文の正確な知識が必要です。ほとんどのデバイスはコマンドの終わりを示す終端文字(例えば改行文字)が必要であることに注意してください。
  例:
  ```
  EA_Comm.RS232_WritePort COM1, "IDN?" & vbLf
  ```
- DAQジョブの実行中は、DAQ操作を行っているデバイスに、任意のコマンドを送るべきではありません。

EA_Comm.SetTimeout

コマンドのタイムアウトを設定します。

```
EA_Comm.SetTimeout(ByVal TimeoutInMilliseconds As Long) As Long
```

引数	説明
TimeoutInMilliseconds	タイムアウト(ms)。

EA_Comm.TCPIP_ClosePort

既存のTCP/IP通信ポートを閉じます。

```
EA_Comm.TCPIP_ClosePort(ByVal ID As Long) As Long
```

引数	説明
ID	EA_Comm.TCPIP_OpenPortで指定したID。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.TCPIP_OpenPort

デバイスへのイーサネットTCP IP通信を開きます。

```
  EA_Comm.TCPIP_OpenPort(ByVal ID As Long, ByVal IPAddress As String, ByVal Port As
Long, Optional ByVal RcvBufSize As Long) As Long
```

引数	説明
ID	以降の*EA_Comm_TCPIP*メソッドでデバイスを識別するための一意のID。通常、1から32の範囲を使用しているcatman内部デバイスと競合してはいけません。ID >= 100を推奨しています。
IPAddress	IPアドレス。例えば"172.19.169.135"。
Port	ポート番号。例えばMGCplusは7。
RcvBufSize	任意です。バイト単位の受信バッファサイズ。デフォルトは64kB。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.TCPIP_ReadPort

開いたTCP/IP接続から文字列を読み取ります。

```
  EA_Comm.TCPIP_ReadPort(ByVal ID As Long, ByVal Terminator As Integer, ByVal
MaxChars As Integer, RcvBuf As String, ByVal Timeout As Long) As Long
```

引数	説明
ID	EA_Comm.TCPIP_OpenPortで指定したID。
Terminator	読み取り動作を停止するASCIIコード(例えば、改行文字(LF)ですべての応答を終了する場合は10)。 終端文字は文字列の受信後に削除してください(例えばReplace$関数で)。
MaxChars	終端文字に関係なく読み取る最大文字数。
RcvBuf	受信文字列。
Timeout	ミリ秒単位のタイムアウト。

戻り値	説明
0	エラーなし。
<0	エラー。例えばタイムアウトの場合はデータを受け取っていません。

注記

- 終端文字に関係なく、指定した文字数を読み取りたい場合、*Terminator*を0に設定します。

EA_Comm.TCPIP_ReadPortBinary

開いたTCP/IP接続から指定したバイト数を読み取ります。

```
  EA_Comm.TCPIP_ReadPortBinary(ByVal ID As Long, ByVal MaxBytes As Long, RcvBuf() As
Byte, ByVal Timeout As Long) As Long
```

引数	説明
ID	EA_Comm.TCPIP_OpenPortで指定したID。
MaxBytes	読み取るバイト数。
RcvBuf	要求したバイトを受け取るバイト配列。この配列は呼び出し元によって割り当てる必要があり、少なくともMaxBytesサイズでなければなりません。
Timeout	ミリ秒単位のタイムアウト。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Comm.TCPIP_WritePort

デバイスにコマンドを送信します。

```
EA_Comm.TCPIP_WritePort(ByVal ID As Long, ByVal Cmd As String) As Long
```

引数	説明
ID	EA_Comm.TCPIP_OpenPortで指定したID。
Cmd	コマンド文字列。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- このメソッドのの使用には、デバイスでサポートしている構文の正確な知識が必要です。ほとんどのデバイスはコマンドの終わりを示す終端文字(例えば改行文字)が必要であることに注意してください。
  例:
  ```
  EA_Comm.TCPIP_WritePort 100, "IDN?" & vbLf
  ```
- DAQジョブの実行中は、DAQ操作を行っているデバイスに、任意のコマンドを送るべきではありません。

EA_Comm.TIME_GetSystemTime

1.1.1970:00:00:00(Unix時間)からの現在時刻を秒単位で返します。分解能は1ミリ秒です。

```
Public Function EA_Comm.TIME_GetSystemTime() As Double
```

EA_DAQ

計測の実行中にこのクラスから受信データおよび一時計測ストレージにアクセスします。SaveDataメソッドにより、常に一時ストレージの内容を保存します。

このグループ一部のメソッドは、DAQジョブの実行中のみ意味を持ちます。通常これらはAutomatic procedure execution ("After transfer DAQ data block"ステージ)に含まれるべきですが、自身ですべてのデータ転送(機器からPCへ)を制御することができます(EA_DAQ.ExecuteDataTransferを参照してください)。

このオブジェクトでは以下が行えます。

- チャネルから単一ポイントの読み取り値を取得: EA_DAQ.CurrentReading
- チャネルの最後のデータ転送ブロックのすべての計測データを取得し、最後の転送ブロックのデータを上書き: EA_DAQ.CurrentBlockSize、EA_DAQ.SetCurrentBlock、EA_DAQ.GetCurrentBlock
- 一時計測ストレージに保存されているチャネルから計測値数を取得: EA_DAQ.GetTotalSampleCount
- 一時計測ストレージに含まれるチャネルのすべての計測データの読み取り: EA_DAQ.GetTempStore
- チャネルの一時計測ストレージに値を書き込み: EA_DAQ.SetTempStore
- データのブロックを一時ストレージ内のチャネルに追加: EA_DAQ.AppendTempStore
- 一時計測ストレージに存在するチャネルのすべてのサンプルを削除: EA_DAQ.ClearTempStore
- 一時計測ストレージへのデータの保存のオン/オフを切り替え: EA_DAQ.TempStoreControl
- 一時計測ストレージに格納しているデータをディスクに保存: EA_DAQ.SaveData
- 機器からPCへのデータ転送を手動制御: EA_DAQ.DataTransferControl、EA_DAQ.ExecuteDataTransfer
- リミットスイッチの状態の取得や、実行時にレベルを変更: EA_DAQ.GetLIVState、EA_DAQ.SetLIVLevels
- 制限値の有効/無効: EA_DAQ.EnableLIV
- 手動トリガ: EA_DAQ.TriggerEvent
- DAQジョブイベントの取得: EA_DAQ.GetEvent
- 利用可能なDAQジョブイベント数を取得: EA_DAQ.EventCount
- DAQジョブイベント数をリセット: EA_DAQ.ResetEventCount
- DAQ ジョブの開始トリガ、停止トリガを起動: EA_DAQ.FireStartTrigger、EA_DAQ.FireStopTrigger
- パネル上のすべての表示器の統計をリセット: EA_DAQ.ResetAllStatistics
- DAQジョブ実行中にテストデータメタファイルにコメントを追加: EA_DAQ.AddComment
- 統計ジャーナルファイル名を設定/取得: EA_DAQ.StatJournalFile
- 統計ジャーナルを閉じて再開: EA_DAQ.CloseStatJournal
- 完全なDAQ周期の総処理時間を取得: EA_DAQ.CycleTime
- 演算チャネルの"cycle count"をゼロにリセット: EA_DAQ.ResetCycleCount
- DAQ ジョブの実行中に"Class count"タイプのオンライン計算の数値データを取得: EA_DAQ.GetClassCountData
- 最後に保存されたテストパラメータファイルのフルパスを取得: EA_DAQ.LastTestExported
- UDP計測の出力を開始/停止: EA_DAQ.UDPStart、EA_DAQ.UDPStop
- データレコーダアクションの実行: EA_DAQ.RecorderControl

このオブジェクトでは以下のメソッドが公開されています:

EA_DAQ.AddComment ・・・

EA_DAQ.EnableLIV ・・・

EA_DAQ.AddComment

DAQ ジョブの実行中に、テストデータメタファイル(.TST)にコメントを追加します。

対話型モードでの、リボンバーにある"Measure"グループにある"Comment"ボタンに相当します。

```
EA_DAQ.AddComment(ByVal Comment As String) As Long
```

引数	説明
Comment	追加するコメント。catmanはこれに、自動的に日付および時刻を接頭辞に付加します。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

EA_DAQ.AddEventDefinition

新しいイベントと、イベントが発生した場合に実行されるアクションを作成します。
EA_DAQ.ReadEventDefinitionも参照してください。

```
Public Function AddEventDefinition(Ev As catmanEvent) As Long
```

引数	説明
ev	catmanのイベント構造。

戻り値	説明
0	エラーなし。
-1	エラー。例: -22 = 重複したイベント名が存在します。

次の表では、catmanEventのメンバーの意味を説明します。

この表にリストされていないcatmanEvent構造体のパラメータはすべて、このスクリプトメソッドで使用することを意図していないため、変更しないでください。

引数	型(VBA)	説明
Name	String	イベントの名前。イベント名は一意である必要があります。
EventType	Integer	0 = 上位レベル交差(このタイプのイベントはチャネルに割り当てる必要があります。EA_DAQ.AssignEventToChannelを参照)。 1 = 下位レベル交差(このタイプのイベントはチャネルに割り当てる必要があります。EA_DAQ.AssignEventToChannelを参照)。 2 = 時間間隔(秒) 3 = デジタル入力 4 = キーボードイベント 5 = EasyScript EA_DAQ.TriggerEvent 6 = 周波数範囲内での交差を制限 7 = DAQジョブの実行中にエラーが発生 8 = CANメッセージを受信 9 = ディスクの空き容量が少ない 10 = RT-Lag(データがハードウェアから十分な速度で転送されない) 11 = ローカル保存時のエラー 12 = DAQ開始トリガが起動 13 = CX22BのUPSモードがアクティブ 14 = オーバーフローのチャネル(このタイプのイベントはチャネルに割り当てる必要があります。EA_DAQ.AssignEventToChannelを参照)。 15 = DAQ停止トリガが起動 16 = リモートのFTPデータをアップロード中にエラー 17 = DAQ ジョブ終了後 18 = DAQ ジョブ開始後t
LimitLevel	Single	交差イベントタイプのレベル。0(上位レベル交差)および1(下位レベル交差)。
Hysteresis	Single	交差イベントは、信号の*Hysteresis*が少なくとも*LimitLevel*を下回るか(イベントタイプ 0、上位レベル交差)上回らない(イベントタイプ 1、下位レベル交差)限り、発生しません。
DInChanName	String	イベントタイプ 3(デジタル入力)に必要です。デジタル入力を監視するチャネルの名前。
DInBit	Byte	イベントタイプ 3(デジタル入力)に必要です。ピン番号。
Disabled	Boolean	Trueの場合、イベントは最初はイベント監視から除外されます(EA_DAQ.EnableLIVを参照)。
KeyCode	Integer	キーボードイベントに使用します。ASCII キーコード、例: 13 = RETURN。

引数	型(VBA)	説明
Shift	Boolean	キーボードイベントに使用します。SHIFTキーを押す必要がある場合は、Trueに設定します。
Ctrl	Boolean	キーボードイベントに使用します。Ctrlキーを押す必要がある場合は、Trueに設定します。
Alt	Boolean	キーボードイベントに使用します。Altキーを押す必要がある場合は、Trueに設定します。
StartTime	Double	時間間隔イベントに使用されます。イベントアクションの開始時刻。
Interval	Single	時間間隔イベントに使用されます。イベントアクションを繰り返す間隔。
FFT_LowFreq	Single	周波数範囲イベントの制限交差に使用します。周波数の下限(Hz)。
FFT_HighFreq	Single	周波数範囲イベントの制限交差に使用します。周波数の上限(Hz)。
FFT_Level	Single	周波数範囲イベントの制限交差に使用します。振幅。
FFT_Hyst	Single	周波数範囲イベントの制限交差に使用します。ヒステリシス。
FFT_Source	String	周波数範囲イベントの制限交差に使用します。Live-FFT表示の名前。
ActionType	Integer	イベントは同時に複数のアクションをトリガする可能性があるため、ビットマスクです: Bit 0 = デジタル出力 Bit 1 = オーディオファイルの再生 Bit 2 = ログメッセージ Bit 3 = プロット色を変更 Bit 4 = EasyScriptコードを実行 Bit 5 = データストレージアクション Bit 6 = 電子メールの送信 Bit 7 = ビデオカメラアクション Bit 8 = ECU制御 Bit 9 = プッシュ通知 Bit 10 = UPS モードの場合、CX22Bのクリーンシャットダウン Bit 11 = HBMクラウド(Power BIなど)でのデータ転送のトリガ 通常、すべてのアクションには以下に説明する追加のパラメータが必要であることに注意してください!
DOutChanName	String	*ActionTypeのDigital output*で使用します。出力にリンクされているチャネルの名前。
DOutBit	Byte	*ActionTypeのDigital output*で使用します。ピン番号。
AlarmColor	Long	*ActionTypeのプロットの色の変更*で使用します。RGBカラー値。
SoundFile	String	*ActionTypeのオーディオ ファイルを再生*で使用します。音声ファイル名。
LogFilePath	String	*ActionTypeのログメッセージ*で使用します。パスを含むログファイルの名前。 LogFileFormat = 0の場合は無視されます。
LogFileFormat	Byte	*ActionTypeのログメッセージ*で使用します。 0 = catmanのシステム ログに書き込み 1 = テキストファイルに書き込み
LogMessage	String	*ActionTypeのログメッセージ*で使用します。ログテキスト。プレースホルダが含まれる場合があります。
StorageAction	Byte	*ActionTypeのデータストレージアクション*で使用します。 0 = 連続ストレージをオン 1 = 連続ストレージをオフ 2 = 単一の値を保存(スナップショット) 3 = データを保存 4 = データを保存し、一時ストアをクリア 5 = DAQジョブを終了し、データを保存してDAQジョブを再開 6 = 開始トリガを起動 7 = 停止トリガを起動
VBProc	String	*ActionTypeのEasyScript コードを実行*で使用します。実行するコード。

引数	型(VBA)	説明

MailAction	String	*ActionTypeのプッシュ通知と電子メールの送信*に使用します。これは、"|;|" 文字シーケンスで区切られたすべてのパラメータを含む文字列です。次の疑似VBA コードの例は、この文字列を構築する方法を説明しています。
		
		プッシュ通知の場合、次のパラメータが必要です:
		・プッシュID
		・件名のテキスト
		・メッセージ本文
		・プッシュ通知の色をRGB値で指定します
		
		`Dim ConfigString As String`
		`Dim sPushID As String, sPushSubject As String, sPushMessage As String`
		`Dim nPushColor As String`
		`Const SEPARATOR As String = "|;|"`
		`'The following parameters are initialized with demo data:`
		`sPushID = "h1b2m3c4a5t6m7a8n9" 'Push ID`
		`sPushSubject = "Push Test" 'Subject text`
		`sPushMessage = "This is a test notification" 'Message body`
		`nPushColor = vbBlue 'Color for push notification as RGB value`
		`'The complete construction of parameterization for push notification:`
		`ConfigString = "@PUSH" & SEPARATOR`
		`ConfigString = ConfigString & CStr(nPushColor ) & SEPARATOR`
		`ConfigString = ConfigString & "0" & SEPARATOR & SEPARATOR & Chr(1) & SEPARATOR & "0" & SEPARATOR & SEPARATOR`
		`'Reserved`
		`ConfigString = ConfigString & sPushID & SEPARATOR`
		`ConfigString = ConfigString & "1234567890" & SEPARATOR & SEPARATOR 'Reserved`
		`ConfigString = ConfigString & sPushSubject & SEPARATOR`
		`ConfigString = ConfigString & sPushMessage & SEPARATOR`
		
		電子メールの場合、次のパラメータが必要です:
		・メールサーバー名
		・SSLオプション
		・ポート
		・認証オプション
		・メールアドレス(送信者)
		・パスワード(送信者)
		・メールアドレス(受信者)
		・件名のテキスト
		・メッセージ本文
		・添付書類(オプション)

```
Dim ConfigString As String
Dim sServer As String, sUser As String, sPassword As
String, sFrom As String, sTo As String, sCc As String
Dim sSubject As String, sMessage As String
Dim nPort As Long
Dim bUseAuthentication As Boolean, bUseSSL As Boolean
Dim Attachments(1) As String 'Just an example for 2
attachments, could be more or less
Const SEPARATOR As String = "|;|"
'The following parameters are initialized with demo
data:
sServer = "mail.hbm.com" 'Name of Email server
bUseSSL = True 'SSL option
nPort = 465 'Port
bUseAuthentication = True 'Authentication option
sUser = "team.catman@hbm.com" 'Email address (Sender)
sPassword = "1234567890" 'Password (Sender)
sFrom = "team.catman@hbm.com" 'Email address (Sender)
sTo = "teamQX.testing@hbm.com" 'Email address (Receiver)
sCc = "teamMGC.testing@hbm.com" 'Email address (Cc)
sSubject = "Email Test" 'Subject text
sMessage = "This is a test Email" 'Message body
Attachments(0) = "C:\Users\catman\Documents\HBM\
catmanEasy\DATA\catman_MX840_TestSig.TST" 'First
attached file
Attachments(1) = "C:\Users\catman\Documents\HBM\
catmanEasy\DATA\catman_MX879_TestSig.TST" 'Second
attached file
'The complete construction of parameterization for
Email:
ConfigString = sServer & SEPARATOR
ConfigString = ConfigString & CStr(nPort) & SEPARATOR
If bUseAuthentication Then
  ConfigString = ConfigString & "1" & SEPARATOR
Else
  ConfigString = ConfigString & "0" & SEPARATOR
End If
ConfigString = ConfigString & sUser & SEPARATOR
ConfigString = ConfigString & sPassword & SEPARATOR
If bUseSSL Then
  ConfigString = ConfigString & "1" & SEPARATOR
Else
  ConfigString = ConfigString & "0" & SEPARATOR
End If
ConfigString = ConfigString & sFrom & SEPARATOR
ConfigString = ConfigString & sTo & SEPARATOR
ConfigString = ConfigString & sCc & SEPARATOR
ConfigString = ConfigString & SEPARATOR 'Reserved
ConfigString = ConfigString & sSubject & SEPARATOR
ConfigString = ConfigString & sMessage & SEPARATOR
ConfigString = ConfigString & Join(Attachments, ";") & ";"
```

引数	型(VBA)	説明
VideoCameraMask	Byte	*ActionTypeのビデオカメラアクション*に使用します。どのビデオカメラが関係しているかを示します。これはビットマスクです: Bit 0 = ビデオカメラ1 Bit 1 = ビデオ カメラ 2 Bit 2 = ビデオ カメラ 3 Bit 3 = ビデオ カメラ 4 Bit 4 = ビデオ カメラ 5 Bit 5 = ビデオカメラ6 Bit 6 = ビデオカメラ7 Bit 7 = ビデオカメラ 8
VideoCameraAction	Byte	*ActionTypeのビデオカメラアクション*に使用します。 0 = キャプチャを開始 1 = キャプチャを停止 2 = 単一フレーム

例

次の例では、レベルを監視し、信号が100kNのレベルを超えた場合に、ActionType*のログメッセージとプロット色を変更*をトリガします:

```
Dim ev As catmanEvent
With ev
  .Name = "Alarm 1"
  .EventType = 0 '0 = Upper level crossing
  .LimitLevel = 100
  .Hysteresis = 1
  .ActionType = 4 'Bit 2 = Log message
  .LogfileFormat = 0 '0 = Write to catman system log
  .LogMessage = "Upper level crossing"
  .ActionType = ev.ActionType + 8 'Bit 3 = Change plot color
  .AlarmColor = vbMagenta 'Plot color -> Magenta
End With
EA_DAQ.AddEventDefinition ev
EA_DAQ.AssignEventToChannel ev.Name, "MY_MX840_CH_1", 0
```

次の例では、レベルを監視し、信号が100kNのレベルを下回った場合*オーディオファイルの再生*アクションをトリガします:

```
Dim ev As catmanEvent
With ev
  .Name = "Alarm 1"
  .EventType = 1 '1 = Lower level crossing
  .LimitLevel = 100
  .Hysteresis = 1
  .ActionType = 2 'Bit 1 = Play audio file
  .SoundFile = "C:\Users\Public\Music\Sample Music\Sleep Away.mp3"
End With
EA_DAQ.AddEventDefinition ev
EA_DAQ.AssignEventToChannel ev.Name, "MY_MX840_CH_1", 0
```

次の例では、測定の実行中にキーボードイベントを監視し、定義されたキーが押された場合に*データストレージアクション*をトリガします:

```
Dim ev1 As catmanEvent, ev2 As catmanEvent, ev3 As catmanEvent, As As catmanEvent
With ev1
  .Name = "Alarm 1"
  .EventType = 4 '4 = Keyboard event
  .KeyCode = &h31 '1 key
  .ActionType = 32 'Bit 5 = Data storage actions
  .StorageAction = 2 '2 = Store single value (Snapshot)
End With
```

```
EA_DAQ.AddEventDefinition ev1
With ev2
  .Name = "Alarm 2"
  .EventType = 4 '4 = Keyboard event
  .KeyCode = &h32 '2 key
  .ActionType = 32 'Bit 5 = Data storage actions
  .StorageAction = 0 '0 = Continuous storage ON
EA_DAQ.AddEventDefinition ev2
End With
With ev3
  .Name = "Alarm 3"
  .EventType = 4 '4 = Keyboard event
  .KeyCode = &h33 '3 key
  .ActionType = 32 'Bit 5 = Data storage actions
  .StorageAction = 1 '1 = Continuous storage OFF
EA_DAQ.AddEventDefinition ev3
End With
With ev4
  .Name = "Alarm 4"
  .EventType = 4 '4 = Keyboard event
  .KeyCode = &h34 '4 key
  .ActionType = 32 'Bit 5 = Data storage actions
  .StorageAction = 3 '3 = Save data
End With
EA_DAQ.AddEventDefinition ev4
```

次の例では、目的の開始時間に達したときにActionTypeの*ビデオカメラアクション*をトリガし、指定された間隔で繰り返します:

```
Dim ev As catmanEvent
With ev
  .Name = "Alarm 1"
  .EventType = 2 '2 = Time interval in s
  .StartTime = #16:25:55# 'Event is triggered at 16:25:55
  .Interval = 3 'Event is repeated in each 3 s
  .ActionType = 128 'Bit 7 = Video camera action
  .VideoCameraMask = 1 'Video camera 1
  .VideoCameraAction = 2 '2 = Single frame
End With
EA_DAQ.AddEventDefinition ev
```

次の例では、チャネルのステータスを監視し、チャネルがオーバーフロー状態にある場合に*プッシュ通知*アクションをトリガします:

```
Dim ev1 As catmanEvent
Dim ConfigString As String
Dim sPushID As String, sPushSubject As String, sPushMessage As String
Dim nPushColor As Long
Const SEPARATOR As String = "|;|"
sPushID = "h1b2m3c4a5t6m7a8n9" 'Push ID
sPushSubject = "Push Test" 'Subject text
sPushMessage = "This is a test notification" 'Message body
nPushColor = vbBlue 'Color for push notification as RGB value
ConfigString = "@PUSH" & SEPARATOR
ConfigString = ConfigString & CStr(nPushColor) & SEPARATOR
ConfigString = ConfigString & "0" & SEPARATOR & SEPARATOR & Chr(1) & SEPARATOR & "0"
& SEPARATOR & SEPARATOR 'Reserved
ConfigString = ConfigString & sPushID & SEPARATOR
ConfigString = ConfigString & "1234567890" & SEPARATOR & SEPARATOR 'Reserved
ConfigString = ConfigString & sPushSubject & SEPARATOR
ConfigString = ConfigString & sPushMessage & SEPARATOR
```

```
  With ev
    .Name = "Alarm 1"
    .EventType = 14 '14 = Channel in Overflow
    .ActionType = 512 'Bit 9 = Send push notification
    .MailAction = ConfigString 'Complete parameterization for push notification
  End With
  EA_DAQ.AddEventDefinition ev
  EA_DAQ.AssignEventToChannel ev.Name, "MY_MX840_CH_1", 0
```

次の例では、イベントEasyScript EA_DAQ.TriggerEventが発生した場合に、*電子メールの送信*アクションをトリガします:

```
  Dim ev As catmanEvent
  Dim ConfigString As String
  Dim sServer As String, sUser As String, sPassword As String, sFrom As String, sTo
As String, sCc As String
  Dim sSubject As String, sMessage As String
  Dim nPort As Long
  Dim bUseAuthentication As Boolean, bUseSSL As Boolean
  Dim Attachments(1) As String 'Just an example for 2 attachments, could be more or
less
  Const SEPARATOR As String = "|;|"
  sServer = "mail.hbm.com" 'Name of Email server
  bUseSSL = True 'SSL option
  nPort = 465 'Port
  bUseAuthentication = True 'Authentication option
  sUser = "team.catman@hbm.com" 'Email address (Sender)
  sPassword = "1234567890" 'Password (Sender)
  sFrom = "team.catman@hbm.com" 'Email address (Sender)
  sTo = "teamQX.testing@hbm.com" 'Email address (Receiver)
  sCc = "teamMGC.testing@hbm.com" 'Email address (Cc)
  sSubject = "Email Test" 'Subject text
  sMessage = "This is a test Email" 'Message body
  Attachments(0) = "C:\Users\catman\Documents\HBM\catmanEasy\DATA\catman_MX840_
TestSig.TST" 'First attached file
  Attachments(1) = "C:\Users\catman\Documents\HBM\catmanEasy\DATA\catman_MX879_
TestSig.TST" 'Second attached file
  ConfigString = sServer & SEPARATOR
  ConfigString = ConfigString & CStr(nPort) & SEPARATOR
  If bUseAuthentication Then
    ConfigString = ConfigString & "1" & SEPARATOR
  Else
    ConfigString = ConfigString & "0" & SEPARATOR
  End If
  ConfigString = ConfigString & sUser & SEPARATOR
  ConfigString = ConfigString & sPassword & SEPARATOR
  If bUseSSL Then
    ConfigString = ConfigString & "1" & SEPARATOR
  Else
    ConfigString = ConfigString & "0" & SEPARATOR
  End If
  ConfigString = ConfigString & sFrom & SEPARATOR
  ConfigString = ConfigString & sTo & SEPARATOR
  ConfigString = ConfigString & sCc & SEPARATOR
  ConfigString = ConfigString & SEPARATOR 'Reserved
  ConfigString = ConfigString & sSubject & SEPARATOR
  ConfigString = ConfigString & sMessage & SEPARATOR
  ConfigString = ConfigString & Join(Attachments, ";") & ";"
```

```
  With ev
    .Name = "Alarm 1"
    .EventType = 5 '5 = EasyScript EA_DAQ.TriggerEvent
    .ActionType = 64 'Bit 6 = Send Email
    .MailAction = ConfigString 'Complete parameterization for Email
  End With
  EA_DAQ.AddEventDefinition ev
```

次の例では、DAQ 開始トリガ起動後に*EasyScriptコードを実行*アクションをトリガします:

```
  Dim ev As catmanEvent
  With ev
    .Name = "Alarm 1"
    .EventType = 12 '12 = DAQ start trigger fired
    .ActionType = 16 'Bit 4 = Execute EasyScript code
    .VBProc = "Sub Main" & vbcrlf & "MsgBox ""DAQ started at: "" & Now" & vbcrlf &
"End Sub"
  End With
  EA_DAQ.AddEventDefinition ev
```

次の例では、CX27のデジタル入力1の状態を監視し、入力1がLOWになった場合に*デジタル出力*アクションをトリガしてデジタル出力1をLOWに設定します:

```
  Dim ev As catmanEvent
  With ev
    .Name = "Alarm 1"
    .EventType = 3 '3 = Digital input
    .DInChanName = "MY_CX27"
    .DInBit = 1 'Input 1 LOW
    .ActionType = 1 'Bit 0 = Digital output
    .DOutChanName = "MY_CX27"
    .DoutBit = 1 'Output 1 LOW
  End With
  EA_DAQ.AddEventDefinition ev
```

次の例では、Live-FFT表示で指定された周波数範囲内でレベルが指定された振幅を超えた場合に、*ログメッセージ*アクションをトリガします:

```
  Dim ev As catmanEvent
  With ev
    .Name = "Alarm 1"
    .EventType = 6 '6 = Limit crossing in frequency range
    .FFT_LowFreq = 0.5
    .FFT_HighFreq = 5
    .FFT_Level = 100
    .FFT_Hyst = 1
    .FFT_Source = "RTGRAPH_1"
    .ActionType = 4 'Bit 2 = Log message
    .LogfileFormat = 0 '0 = Write to catman system log
    .LogMessage = "Limit crossing in frequency range"
  End With
  EA_DAQ.AddEventDefinition ev
```

EA_DAQ.AppendTempStore

データのブロックを一時ストレージ内のチャネルに追加します。

対話型モードには相当する機能がありません。これは、デバイスからデータを取得し、一時保管への書き込み時にcatmanEasy/APが使用します。

```
  EA_DAQ.AppendTempStore(ByVal Channel As Variant, ByVal Count As Long, Buffer() As
Double) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
Count	追加する値数。
Buffer	追加する値が含まれるバッファ。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- 一時計測保管は、DAQジョブを開始(または反復)する前に自動的にクリアされます。
- 一時保管(OPTIONS/DATA STORAGEを参照)に固定サイズを使用している場合は以下を考慮してください:
  新しいチャネル長がチャネルごとの最大サンプル数を超える場合、このメソッドは失敗します。
- Countがバッファ内の数よりも多い場合、バッファ内に存在する値のみが書き込まれ、戻り値にエラーコードが含まれます。
- 名前を使用してチャネルを指定すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

EA_DAQ.AssignEventToChannel

イベントを DAQ チャネルに割り当てます。

```
  Public Function AssignEventToChannel(ByVal EventName As String, ByVal Channel As
Variant, ByVal Index As Integer)
```

引数	説明
EventName	イベント名。
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
Index	制限値条件のインデックス。各チャネルは最大3つの制限値条件(LV1～LV3)を監視できるため、これは0が基準で、0～2の間である必要があります。 Index = 0: LV1 Index = 1: LV2 Index = 2: LV3

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

EA_DAQ.ClearTempStore

一時計測ストレージに存在する、チャネルのすべてのサンプルを削除します。

対話型モードには相当する機能がありません。ただし、一時計測ストレージは、DAQジョブを開始する前に自動的にクリアされます。

```
EA_DAQ.ClearTempStore(ByVal Channel As Variant) As Long
```

引数	説明
Channel	チャネル名または位置。-1を指定すると、すべてのチャネルの一時計測ストレージをクリアします。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- 一時計測ストレージは、DAQジョブを開始(または反復)する前に自動的にクリアされます。
- EA_DAQ.SetTempStoreを使用すると、独自の値で一時計測ストレージを上書きすることができます。
- 名前を使用してチャネルを指定すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

EA_DAQ.CloseStatJournal

この関数は、*Statistics Journal*を閉じて再開します。1 つの非常に長いDAQジョブ中に、複数のジャーナルファイルを作成することができます。例えば数週間DAQジョブを実行して一日毎の統計ジャーナルを取得したい場合、"Time interval" (リボンメニュー"Channel configuration/Limit monitoring and events)のイベントを作成し、このスクリプトコマンドを"Script"アクションでを使用します。

```
EA_DAQ.CloseStatJournal() As Long
```

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- 新規統計ジャーナルに一意の名前が取得できるよう、DAQジョブファイル名にプレースフォルダ%DateTime%を使用してください。

EA_DAQ.CurrentBlockSize

最後のデータ転送ブロック(読み取りブロック)に含まれているサンプル数を取得します。この値を使用して、EA_DAQ.GetCurrentBlockへの後続の呼び出しで使用されるバッファにメモリを割り当てたり、補助チャネルにこの数の値(EA_DAQ.SetCurrentBlock)を埋めたりすることができます。

対話型モードに相当する機能がありません。これは、リアルタイム表示器、制限値監視およびトリガの更新時にcatmanEasy/APが使用します。

```
EA_DAQ.CurrentBlockSize(ByVal Channel As Variant, BlockSize As Integer) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。 この引数はゼロにすることができます - この場合catmanは、現在処理中の補助チャネルの基準チャネルからサンプル数を返します。直接スクリプトコードの使用方法も参照してください。
BlockSize	ブロックに含まれているサンプル数。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- このメソッドはDAQジョブの実行中のみ行うことができます。手動でデータ転送(EA_DAQ.ExecuteDataTransferを参照)制御を行わない限り、通常はAutomatic procedure execution("After transfer DAQ data block"ステージ)に含まれるべきです。
- 名前を使用してチャネルを指定すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
Dim nSamples As Integer
Dim Buffer() As Double
EA_DAQ.CurrentBlockSize "Pressure", nSamples
ReDim Buffer(nSamples) As Double
EA_DAQ.GetCurrentBlock "Pressure", Buffer
For i=0 To nSamples - 1
  'Do something with data in Buffer
  ...
Next
```

EA_DAQ.CurrentReading

チャネルの最後のテータ転送ブロックの値、またはこのブロックの最後の統計データを返します。

対話型モードに相当する機能がありません。これは、リアルタイム表示器の更新時にcatmanEasy/APが使用します。

```
EA_DAQ.CurrentReading(ByVal Channel As Variant, Reading As Double, Optional ByVal
DataType As Integer, Optional ByVal StartIndex As Long, Optional ByVal Count As Long)
As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
Reading	DataTypeに応じた値。
DataType	0 = 最新の値(読み取りブロック最後の値) 1 = 読み取りブロックの最大値 2 = 読み取りブロックの最小値
StartIndex	開始位置。
Count	読み取り数。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

EA_DAQ.GetCurrentBlockも参照してください。

注記

- このメソッドはDAQジョブの実行中のみ行うことができます。手動でデータ転送(EA_DAQ.ExecuteDataTransferを参照)制御を行わない限り、通常はAutomatic procedure execution("After transfer DAQ data block"ステージ)に含まれるべきです。

- 名前を使用してチャネルを指定すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
Sub Monitor()
  Dim MV As Double
  EA_DAQ.CurrentReading "Valve pressure", MV
  If MV > 50 Then
    EA_Panel.ShowPanel "Panel 5"
  End If
End Sub
```

EA_DAQ.CycleTime (Read only)

これは有用な診断プロパティです。完全なDAQ周期および周期のサブパートの総処理時間(ms)を返します。
以下のDAQ周期が含まれます:

- 機器から新しいデータを取得
- オンライン計算の実行と補助チャネルの処理
- サンプルを一時的に保管
- すべての表示器およびグラフを更新
- イベント監視
- DAQ転送に関連した自動実行ステージに接続するスクリプト関数
- 周期保存モードが有効の場合にデータを保存
- DAQ周期中に自動実行するEasyScript(Automatic Execution stagesを参照)

総周期時間は、平均100 msを超えてはなりません。そうしないとDAQジョブが不安定になり、時間遅延が発生します。この時間が短いほど、DAQジョブのパフォーマンスは向上します。50 ms未満であれば問題ありません。時々、一時データ保管のキャッシュバッファの掃き出し周期による、保管時間のピークが発生することがありますが、これらのピーク時間は許容範囲内です。

```
Property EA_DAQ.CycleTime(ByVal ID As Integer) As Variant
```

引数	説明
ID	0 = 総周期時間 1 = データブロックを一時保管に格納するために必要な時間 2 = すべての視覚化を更新するために必要な時間(チャート、スコープ、表示器など) 3 = 統計表示の更新に必要な時間 4 = イベント監視に必要な時間 5 = 開始トリガの監視に必要な時間 6 = DAQのステータス情報の更新に必要な時間(約5秒間隔で更新します) 7 = 補助チャネルの処理に必要な時間 8 = データをディスクに保存するために必要な時間(周期保存モードの場合) 9 = 統計ジャーナルの更新に必要な時間 10 = ハードウェアからDAQバッファにデータを転送するのに必要な時間 11 = オンラインでの演算チャネルおよびデータのスケーリングに必要な時間 12 = 停止トリガの監視に必要な時間 13 = "After transfer DAQ data block"ステージ後に自動実行するEasyScriptコード 14 = "After online computations"ステージ後に自動実行するEasyScriptコード 15 = "After data storage"ステージ後に自動実行するEasyScriptコード 16 = "After complete DAQ cycle"ステージ後に自動実行するEasyScriptコード 個々の時間は合計すると、おおよそ総周期時間になります。

EA_DAQ.DataTransferControl

実行中のDAQジョブのすべてのデバイスから、データブロックの転送を制御します。この関数を使用して、スクリプトとデータ転送を同期します。新規データを一時保管に追加したくないときに有用です。

通常、この転送はcatmanのバックグラウンドプロセスにより、毎秒10～20回自動的に行われます。しかし、場合によっては、スクリプトが最後の転送のデータを処理できるように、このプロセスを停止することが望ましい場合があります。これはもちろん、自動実行ステージの"After transfer DAQ data block"にスクリプトプロシージャを含めることができますが、このプロセスをさらに詳細に制御したい場合があります。

⚠ データ転送を一時停止していると、デバイスからデータを取得しません! これは内部バッファのオバーフローにつながります。内部バッファの容量はデバイスの種類によりますが、数秒程度です。2～3秒以上のデータ転送の一時停止はさけてください。

```
EA_DAQ.DataTransferControl(ByVal Paused As Integer) As Long
```

引数	説明
Paused	0 = データ転送 1 = データ転送を一時停止

注記

- Automatic Procedure Eexecutionの*After transfer DAQ data block*も参照してください。
- EA_DAQ.ExecuteDataTransferと併用すると、自動データ転送を完全にオフにして、このプロセスをすべて自身で制御することもできます。

  💡 *catman Professional*に精通しているユーザの場合: 概ね*ACQRead*関数に似ています。

EA_DAQ.DeleteEvent

イベントを削除します。イベントがDAQチャネルに割り当てられている場合、その割り当ても自動的に削除されます。

```
Public Function DeleteEvent(ByVal EventName As String)
```

引数	説明
EventName	イベント名。

戻り値	説明
0	エラーなし。
<0	エラー。イベントまたはチャンネルが存在しないなど。

EA_DAQ.EnableLIV

制限値を有効/無効にします。

```
EA_DAQ.EnableLIV(ByVal Name As String, ByVal State As Integer) As Long
```

Nameは、制限値の作成時に定義した制限値条件の名前を指定します。

引数	説明
Name	制限値名。
State	状態(0 = 無効、1 = 有効)。

戻り値	説明
0	エラーなし。
<0	エラー。制限値が存在しないなど。

例

```
'Disables for example the limit value "Measuring range OVFL positive".
EA_DAQ.EnableLIV("Measuring range OVFL positive", 0)
```

EA_DAQ.EventCount

利用可能なDAQジョブイベント数を返します。

```
EA_DAQ.EventCount
```

戻り値	説明
>=0	利用可能なDAQジョブイベント数。

EA_DAQ.ExecuteDataTransfer

実行中のDAQジョブのすべてのデバイスから、データブロックの転送を制御します。この関数を使用して、スクリプトとデータ転送を同期します。新規データを一時保管に追加したくないときに有用です。

通常、この転送はcatmanのバックグラウンドプロセスにより、毎秒10～20回自動的に行われます。しかし、場合によっては、スクリプトが最後の転送のデータを処理できるように、このプロセスを停止することが望ましい場合があります。これはもちろん、自動実行ステージの"After transfer DAQ data block"にスクリプトプロシージャを含めることができますが、このプロセスをさらに詳細に制御したい場合があります。

*catman Professional*に精通しているユーザ向け: 概ね*ACQRead*関数に似ています。

EA_DAQ.DataTransferControlを使用して、自動転送が無効になっていることを確認してください!

データ転送を一時停止していると、デバイスからデータを取得しません! これは内部バッファのオバーフローにつながります。内部バッファの容量はデバイスの種類によりますが、数秒程度です。2～3秒以上のデータ転送の一時停止はさけてください。

```
EA_DAQ.ExecuteDataTransfer() As Long
```

注記

- Automatic Procedure Eexecutionの*After transfer DAQ data block*も参照してください。データ転送を手動で制御するか、catmanによって自動的に制御されるかに関係なく、この実行ステージに割り当てられたスクリプトプロシージャが実行されることを確認します。

EA_DAQ.ExportChannelData

このメソッドは、単一チャネルの数値データを一時データストレージからさまざまな形式にエクスポートします:

- ASCIIテキストファイル(.txt)
- 再生ファイル(.catwfm)
- 音声ファイル(.wav)

catmanは、上記ファイルの拡張子に基づいて形式を決定します。

結果として得られるファイルの構造は、形式とチャネルまたは計算の種類によって異なります。これについては、このトピックで後ほど説明します。

参照: 複数のチャネルをエクスポートするには、EA_DAQ.ExportMultipleChannelData

インタラクティブモード(解析モード)では、これは解析プロジェクトリストのコンテキストメニュー"Export data"に対応します。

```
EA_Test.ExportChannelData(ByVal Channel as Variant, ByVal FileName As String,
Optional ByVal NumFormat as String, Optional ByVal Delimiter As String)
```

引数	説明
Channel	チャネル名または位置。
FileName	ファイル拡張子によって形式が決定: .txt = ASCIIテキストファイル .catwfm = catman再生ファイル .wav = 音声ファイル
NumFormat	オプションです。数値のASCII変換で使用される数値精度(小数点以下の桁数)。例: "0.00" "0.0000" "0"(整数) "Auto"(小数点以下の桁数は値の大きさから決定します) "Full"(フル64 ビット精度、14桁) 省略した場合は小数点以下3桁を使用します。
Delimiter	オプションです。出力ファイルに複数の列 (クラス数行列、ジョイント周波数スペクトル行列、レベルでの時間、スパンペア)が含まれる場合に列を区切るために使用される区切り文字。省略した場合のデフォルトはTABです。

ASCII Text

チャネルや演算によるファイルの構造は以下の通りです:

データ型	**ファイル構造**
計測チャネル	CRLFで区切った数値の線形ベクトル。
オンライン演算チャネル(電卓、フィルタ、ロゼット)	CRLFで区切った数値の線形ベクトル。
演算: 電卓	CRLFで区切った数値の線形ベクトル。
演算: 周波数スペクトル	CRLFで区切った数値の線形ベクトル。
演算: ジョイント時間周波数スペクトル	複数列、列は*Delimiter*で区切り、行はCRLFで終了。 行1: 周波数 列1: 時間 セル(列、行)=振幅スペクトル
演算: フィルタ	CRLFで区切った数値の線形ベクトル。
演算: 外れ値の除去	CRLFで区切った数値の線形ベクトル。
演算: 区間(cut)	CRLFで区切った数値の線形ベクトル。

データ型	ファイル構造
演算: レインフロー(FromTo)	複数列、列はDelimiterで区切り、行はCRLFで終了。 行1: クラス制限 (最も低いクラス順) 列1: クラス制限 (最も高いクラス順) セル(列、行)=カウント
演算: レインフロー(RangeMean)	複数列、列はDelimiterで区切り、行はCRLFで終了。 行1: 範囲値 列1: 平均値 セル(列、行)=カウント
演算: スパンペア	複数列、列はDelimiterで区切り、行はCRLFで終了。 列1: スパン幅 列2: カウント
演算: レベルでの時間	複数列、列はDelimiterで区切り、行はCRLFで終了。 列1: 振幅区分 列2: 振幅間隔に費やした時間(秒)
演算: レベルでの共同時間	複数列、列はDelimiterで区切り、行はCRLFで終了。 行1: クラス制限 列1: クラス制限参照チャネル セル(列、行)=秒単位の時間

catman再生ファイル

最適な再生能力を確保する再生ファイルのの構造は、以下の通りです:

4 Byte DWORD	ファイルID (35014414)
4 Byte DWORD	最初のサンプルのバイト単位のオフセット
4 Byte DWORD	サンプル数
8 Byte DOUBLE	サンプルレート(Hz)
2 Byte WORD	チャネル名の文字数N
N Bytes CHAR	チャネル名
2 Byte WORD	チャネルの単位の文字数N
N Bytes CHAR	チャネルの単位
8 Byte DOUBLE	1. サンプル
8 Byte DOUBLE	2. サンプル
.....	.....

音声ファイル

このファイルの構造は、.WAV標準(RIFF)準拠です。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

EA_DAQ.ExportMultipleChannelData

このメソッドは、チャネルの数値データを一時データストレージから ASCIIテキストファイルにエクスポートします。

⚠ このメソッドは信号ベクトル(つまり時系列)にのみ適用できますが、クラス集計チャネルや行列には適用できません。

⚠ この方法は、catman再生ファイル(.catwvfm)または音声ファイル(.wav)の作成には使用できません。

クラス集計チャネルまたは行列のエクスポートについては、EA_DAQ.ExportChannelDataも参照してください。

インタラクティブモード(解析モード)では、これは解析プロジェクトリストのコンテキストメニュー"Export data"に対応します。

```
EA_DAQ.ExportMultipleChannelData(ByVal DataSource As String, ByVal FileName As String, Optional ByVal NumFormat as String, Optional ByVal Delimiter As String)
```

引数	説明
DataSource	以下のように"|"文字で区切ります。"Channel name|0|Channel name|0....." (例: "Pressur|1|Oil temperature|1|Wheel force|1.....")
FileName	結果のテキストファイル名(パスを含む)。
NumFormat	オプションです。数値のASCII変換で使用される数値精度(小数点以下の桁数)。例: "0.00" "0.0000" "0"(整数) "Auto"(小数点以下の桁数は値の大きさから決定します) "Full"(フル64 ビット精度、14桁) 省略した場合は小数点以下3桁を使用します。
Delimiter	任意です。出力ファイルが複数が含まれる場合に使用する列の区切り文字。省略した場合、Tabをデフォルトして使用。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルまたは演算チャネルが存在しないなど。

EA_DAQ.FireStartTrigger

DAQジョブの開始トリガを起動します。

```
EA_DAQ.FireStartTrigger() As Long
```

EA_DAQ.FireStopTrigger

DAQジョブの停止トリガを起動します。

```
EA_DAQ.FireStopTrigger() As Long
```

EA_DAQ.GetClassCountData

実行中のDAQジョブのクラスカウント型のオンライン計算の数値データを返します。例えば、計測中にレインフロー計数チャネルの行列データをASCIIファイルにに書き込む場合に有用です。戻り値のデータ構成は、以降に説明するカウントの種類によります。

便宜上、この関数は、クラス数、クラスの最小値と最大値など、カウントルゴリズムを説明するメタデータも返します。

```
  EA_DAQ.GetClassCountData(ByVal Channel As Variant, Buffer() As Double, NofClasses
As Integer, ClassMin As Double, ClassMax As Double, Optional ClassMin2 As Double,
Optional ClassMax2 As Double) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
Buffer	出力: 結果データを受け取る配列。必要な最小サイズ: Rainflow FromTo、Joint time at level: nClasses*nClasses Rainflow RangeMean: nClasses*nClasses+ 2*nClasses Time at level、SpanPairs: nClasses 可能なクラスの最大数は256のため、カウントメソッドに関係なく、次のバッファを割り当てるのが最も簡単です。 Dim Buf(256*256 + 2*256) As Double
nClasses	出力: クラス数。
ClassMin	出力: クラスの下限値。
ClassMax	出力: クラスの上限値。
ClassMin2	任意です(Joint time at levelのみ)。出力: Joint time at levelでの参照チャネルのクラスの下限値。
ClassMax2	任意です(Joint time at levelのみ)。出力: Joint time at levelでの参照チャネルのクラスの上限値。

返されるバッファ内のデータの構成は、カウントメソッドの型によって異なります。

メソッド	データ
Rainflow FromTo	以下のようにBufferは、行の後に行を構成した計数行列を含みます: バッファには、次のように行ごとに構成されたカウント行列が含まれています。 Buffer(0)=R1C1 Buffer(1)=R1C2 Buffer(2)=R1C3 ..... Buffer(nClasses)=R2C1 Buffer(nClasses+1)=R2C2 ..... Buffer(2*nClasses)=R3C1 .... 行列の横/縦軸の値は、以下から計算します: x(i) = ClassMin + (ClassMax-ClassMin)/nClasses*i y(i) = x(i)

メソッド	データ
Rainflow RangeMean	バッファには、まず範囲および平均点の二つのベクトルが含まれ、その後に次のように行ごとに構成されたカウント行列が続きます: Buffer(0)=Range abscissa point 1 Buffer(1)=Range abscissa point 2 ..... Buffer(nClasses) = Mean value ordinate point 1 Buffer(nClasses+1) = Mean value ordinate point 2 ..... Buffer(2*nClasses)=R1C1 Buffer(2*nClasses1+1)=R1C2 Buffer(2*nClasses2+2)=R1C3 ....
Time at level	バッファには、信号が各クラスで費やした時間(秒単位)を含む線形ベクトルが含まれています: Buffer(0)=Time in class 1 Buffer(1)=Time in class 2 ..... 行列の横軸/縦軸の値は次のように計算します: x(i) = ClassMin + (ClassMax-ClassMin)/nClasses*i
Joint time at level	バッファには、次のように行ごとに構成されたカウント行列が含まれています。 Buffer(0)=R1C1 Buffer(1)=R1C2 Buffer(2)=R1C3 ..... Buffer(nClasses)=R2C1 Buffer(nClasses+1)=R2C2 ..... Buffer(2*nClasses)=R3C1 .... 行列の横軸/縦軸の値は次のように計算できます。: x(i) = ClassMin + (ClassMax-ClassMin)/nClasses*i y(i) = ClassMin2 + (ClassMax2-ClassMin2)/nClasses*i

EA_DAQ.GetCurrentBlock

チャネルの最新のデータ転送ブロックのすべての値を取得します。

対話型モードに相当する機能がありません。これはリアルタイム表示器、制限値やトリガ監視の更新時にcatmanEasy/APが使用します。

```
EA_DAQ.GetCurrentBlock(ByVal Channel As Variant, Buffer() As Double) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。 この引数はゼロにすることができます- この場合、catmanは現在処理中の補助チャネルの基準チャネルからデータを返します。直接スクリプトコードの使用方法も参照してください。
Buffer	値を保持するバッファ。バッファの最初の値(インデックス = 0)は、ブロックの最も古い値が含まれています。

戻り値	説明
>0	ブロック内の値の個数。
0	利用可能な値なし。
<0	エラー。チャネルが存在しないなど。

注記

- このメソッドは実行中のDAQジョブのみ行うことができます。手動でデータ転送(EA_DAQ.ExecuteDataTransferを参照)制御を行わない限り、通常はAutomaticprocedureexecution("After transfer DAQ data block"ステージ)に含まれるべきです。

- バッファを呼び出す前に、適切にサイズを指定する必要があります。catmanEasy/APの一般的な読み取りブロックサイズは、サンプルレートの約1/10です。最大値は、ブロックごとに2048です。また、現在の読み出しブロック内のサンプル数を取得するには、EA_DAQ.CurrentBlockSizeメソッドを使用します。

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
Sub Monitor()
  Dim Buffer(1000) As Double
  Dim nSamples As Long
  nSamples = EA_DAQ.GetCurrentBlock("Valve pressure", Buffer)
  For i=0 To nSamples-1
    'Do something with data in Buffer
    ...
  Next
End Sub
```

EA_DAQ.GetEvent

EventIndexで指定したDAQジョブイベントを取得します。

```
  EA_DAQ.GetEvent(ByVal EventIndex As Integer, EventMessage As String, EventTime As
Double) As Long
```

引数	説明
EventIndex	DAQジョブイベントのインデックス。イベントは1から始まります。
EventMessage	イベントの詳細メッセージ。
EventTime	イベントが発生した時刻。

戻り値	説明
0	エラーなし。
<0	エラー。イベントのインデックスが範囲外など。

EA_DAQ.GetLIVState

DAQジョブの実行中に制限値スイッチの状態を取得します。

```
  EA_DAQ.GetLIVState(ByVal Channel As Variant, ByVal Index As Integer, State As
Integer) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
Index	制限値スイッチの番号(1..3)。
State	状態(0 = 無効、>0 = 有効)。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- この関数を利用するには、対話的に制限値を定義して割り当てる必要があります。
- このメソッドは実行中のDAQジョブのみ行うことができます。通常はAutomatic procedure execution("After transfer DAQ data block"ステージ)に含まれるべきです。

例

```
EA_DAQ.GetLIVstate("Ventildruck", 1, Status)
If Status = 1 Then
  .......
End If
```

EA_DAQ.GetTempStore

一時計測ストレージのチャネルのすべての計測値を、バッファに書き込みます。

対話型モードに相当する機能がありません。Post-process graphの更新時にcatmanEasy/APが使用します。

```
EA_DAQ.GetTempStore(ByVal Channel As Variant, Buffer() As Double, Optional ByVal
StartIndex As Long, Optional ByVal Count As Long ) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
Buffer	値を保持するバッファ。バッファの最初の値は、ブロックの最も古い計測値に対応します。
StartIndex	任意です。データを返す位置(最初のサンプルのインデックスは0)。省略した場合、インデックス0から返します。
Count	任意です。返すサンプル数。省略した場合、*StartIndex*から最後までのサンプルを返します。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- EA_DAQ.GetTotalSampleCount(一時計測保管の値数)を使用してバッファサイズを取得します。
- 一時計測ストレージは、DAQジョブを開始(または反復)する前に自動的にクリアされます。
- EA_DAQ.SetTempStoreを使用すると、独自の値で一時計測保管を上書きすることができます。
- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
Sub RemoveOutliers()
  Dim Buffer() As Double
  Dim nSamples As Long
  nSamples = EA_DAQ.GetTotalSampleCount("Valve pressure")
  ReDim Buffer(nSamples-1)
  EA_DAQ.GetTempStore("Valve pressure", Buffer)
  For i=0 To nSamples-1
    If Buffer(i) > 5000 Then Buffer(i) = Buffer(i-1)
  Next
  EA_DAQ.SetTempStore("Valve pressure", nSamples, Buffer)
End
```

EA_DAQ.GetTotalSampleCount

一時計測ストレージに保存したチャネルから計測値数を返します。

対話型モードに相当する機能がありません。Post-process graphの更新時にcatmanEasy/APが使用します。

```
EA_DAQ.GetTotalSampleCount(ByVal Channel As Variant) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。

戻り値	説明
>=0	チャネルの一時計測メモリ内の計測数。
<0	エラー。チャネルが存在しないなど。

注記

- 最小バッファサイズを見つけるにはEA_DAQ.GetTempStoreを使用します。
- 一時計測ストレージは、DAQジョブを開始(または反復)する前に自動的にクリアされます。
- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
Sub RemoveOutliers()
  Dim Buffer() As Double
  Dim nSamples As Long
  nSamples = EA_DAQ.GetTotalSampleCount("Valve pressure")
  ReDim Buffer(nSamples-1)
  EA_DAQ.GetTempStore("Valve pressure", Buffer)
  For i=0 To nSamples-1
    If Buffer(i) > 5000 Then Buffer(i) = Buffer(i-1)
  Next
  EA_DAQ.SetTempStore("Valve pressure", nSamples, Buffer)
End
```

EA_DAQ.LastTestExported (Read only)

最後に保存されたテストパラメータファイルのフルパスを返します。

```
Property EA_DAQ.LastTestExported
```

注記

- 最後のDAQジョブの計測データを保存していない場合、空の文字列を返します。

EA_DAQ.ReadEventDefinition

既存イベントの内容を読み出します。
EA_DAQ.AddEventDefinitionも参照してください。

```
  Public Function ReadEventDefinition(ByVal EventIndex As Integer, ev As catmanEvent,
Optional ByVal EventName As String) As Long
```

引数	説明
EventIndex	イベントのインデックス。DAQ プロジェクトのイベントは1から始まります。
ev	catmanのベント構造。catmanEventのメンバーの詳細については、EA_DAQ.AddEventDefinitionを参照してください。
EventName	オプションです。イベント名。指定した場合、イベントは名前で検索され、EventIndexは無視されます。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

例

```
'Assume that there is an event in the DAQ project with the index 1
Dim ev As catmanEvent
EA_DAQ.ReadEventDefinition 1, ev
If ev.Name = "Alarm 1" Then
' Your code here...
End If
```

EA_DAQ.RecorderControl

データレコーダアクションを実行します。

```
  EA_DAQ.RecorderControl(ByVal RecorderIndex As Integer, ByVal Action As Integer,
Param As Integer) As Long
```

引数	説明
RecorderIndex	データレコーダのインデックス。データレコーダのインデックスは1から始まります。
Action	実行するアクションのインデックス: 1 = 記録を開始 2 = 記録を停止 3 = 記録を一時停止 4 = 記録を再開 5 = レコーダーのステータスを取得
Param	データレコーダのステータス。RecorderIndexが5の場合のみ使用します: 0 = 記録中 1 = 待機中 2 = 一時停止 3 = 保存 4 = エラー 5 = 手動トリガ 6 = 手動停止 7 = ポストトリガ中 8 = 終了 9 = 再トリガ

戻り値	説明
0	エラーなし。
<0	エラー。データレコーダが存在しないなど。

注記

- Parallel Data Recorderアドインが実行されていない場合、この関数は効果がありません。
- Parallel Data Recorder機能を使用するには、EasyMonitoringモジュールまたはcatmanAPライセンスが必要です。

EA_DAQ.ResetAllStatistics

パネル上のすべての統計情報をリセットします。

```
EA_DAQ.ResetAllStatistics
```

注記

- 指定したパネルの統計情報を個別にリセットするには、EA_Panel.ResetStatisticsを使用します。

EA_DAQ.ResetCycleCount

演算チャネルの"cycle count"をゼロにリセットします。

```
EA_DAQ.ResetCycleCount(ByVal Channel As Variant) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

EA_DAQ.ResetEventCount

DAQジョブのイベント数をゼロにリセットします。

```
EA_DAQ.ResetEventCount
```

注記

- このメソッドは、DAQジョブのイベント数を0に設定します。新規DAQジョブイベントの指標は1から始まります。

EA_DAQ.SaveData

一時計測保管に格納しているデータをディスクに保存します。

対話型モードでのDAQ job settingsのSaveオプションに相当します。スクリプトから保存プロセスを制御したい場合、このオプションは"Do not save"に設定します。

```
  EA_DAQ.SaveData(ByVal FileName As String, ByVal FileFormat As Integer, ByVal
Precision As Integer, ByVal Comment As String, ByVal AmountToSave As Single, ByVal
ExportCustomData As Integer) As Long
```

引数	説明
FileName	パスを含むファイル名。空の文字列の場合、現在のジョブ設定で指定しているファイル名を使用します(また必要に応じて自動的に増分します)。
FileFormat	ファイル形式: 0 = 現在のジョブ設定 1 = catman 2 = ASCII 3 = ASCII + チャネル情報 4 = Excel (Office XP/2003) 5 = Diadem DAT 6 = nSoft DAC 7 = Vector MDF 3 8 = Excel (Office 2007 XLSX XML) 9 = Excel (Office 2007 XLSB Binary) 10 = Matlab 5.0 11 = RPC III 12 = nCode s3t 13 = ASAM MDF 4.1 14 = UFF58 15 = NI TDMS
Precision	精度: 0 = 現在のジョブ設定 1 = 8-Byte 浮動小数点 2 = 4-Byte 浮動小数点 3 = 2-Byte 整数 この引数はcatman標準形式(*Format* = 1)、MDF 3(*Format* = 7)およびMDF 4 format (*Format* = 13)にのみ影響があります。他のすべての形式の精度は、形式(MatlabやNI DiademのDOUBLE型)に応じて自動的に選択されます。
Comment	ファイルのコメント。空きの文字列の場合、現在のジョブ設定で指定しているファイルのコメントを使用します。
AmountToSave	保存(一時保管の最新値から始まる)する時間(秒)。ゼロの場合、現在のDAQジョブの設定を使用します。通常はデフォルトの0秒 = すべてを保存します! この引数と適切なスクリプトプログラミングにより、プレおよびポストトリガの"イベントベース"ストレージを実現することができます。これは一時保管と共に使用する周期モードで動作します。
ExportCustomData	通常はDAQジョブに参加する計測チャネルのみを保存します。スクリプトプログラマが、DAQジョブ実行中、複数チャネルの一時保管に計算データを追加格納(EA_DAQ.SetTempStoreを参照)する場合、この引数を1に設定します。この場合、追加チャネルも保存します。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- 一時保管内のデータ量によっては、非常に長い時間がかかることがあります。DAQジョブの実行中にこのメソッドを実行すると、保存プロセス中にデバイスからデータが転送されず、内部デバイスバッファがオーバーフローする可能性があることに注意してください。

- DAQジョブの実行中に何回も保存すると、生成されるファイルは大きくなっていきます。ただし、保存操作毎に一時保管をクリアすると(EA_DAQ.ClearTempStore)、複数の小さなファイルに分けることができます。

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。

EA_DAQ.SetCurrentBlock

このメソッドは、最後のデータ転送ブロックを上書きします。例えば、リアルタイムグラフに表示させたい補助チャネルのデータを提供します。カスタムデータのグラフ表示も参照してください。

対話型モードに相当する機能がありません。

```
EA_DAQ.SetCurrentBlock(ByVal Channel As Variant, ByVal StartIndex As Integer, ByVal Count As Long, Buffer() As Double) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。 この引数はゼロにすることができます - この場合catmanは、現在処理中の補助チャネルの基準チャネルからサンプル数を返します。Executing direct script codeも参照してください。
StartIndex	データ転送ブロックの指標(最初のサンプルは指標1で最も古い値)。
Count	書き込む値の数。転送ブロックの合計数を知るにはEA_DAQ.CurrentBlockSizeを使用します。
Buffer	書き込むデータを含むデータバッファ。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- このメソッドは実行中のDAQジョブのみ行うことができます。手動でデータ転送(EA_DAQ.ExecuteDataTransferを参照)制御を行わない限り、通常はAutomatic procedure execution("After transfer DAQ data block"ステージ)に含まれるべきです。
- このメソッドで補助チャネルのデータを使用する場合、サンプリングレートグループからサンプル数を取得するため、"lead channel"が必要です。これは通常のDAQチャネル(ハードウェアチャネル)です。このチャネル転送ブロックの合計サイズを取得するには、EA_DAQ.CurrentBlockSizeメソッドを使用します。
- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

次の例は、補助チャネルのデータを計算する方法を示しています(コンテキストメニューのComputations/Auxiliary channelまたは、EA_IO.CreateAuxChannelによるスクリプトで、Channel Configurationウィンドウから補助チャネルを作成):

```
Dim nSamples As Integer
Dim Buf1() As Double, Buf2() As Double, Buf3() As Double
EA_DAQ.CurrentBlockSize "RevsPerSecond", nSamples
ReDim Buf1(nSamples)
ReDim Buf2(nSamples)
ReDim Buf3(nSamples)
EA_DAQ.GetCurrentBlock "RevsPerSecond", Buf1
EA_DAQ.GetCurrentBlock "Speed", Buf2
For i=0 To nSamples-1
  Buf3(i) = 0.5*Buf1(i)/Buf2(i)+5.9
Next
EA_DAQ.SetCurrentBlock "Aux1", 1, nSamples, Buf3
```

動作を示す単純な例です。このような単純な計算はスクリプトを必要としませんが、オンラインの演算チャネルでは起こりえます。より複雑な計算(If...Then判定、リアルタイムで外れ値の除外を含む)やデータの取得はファイルから読み込まれるべきであり、これを行うための適切な手法です。

EA_DAQ.SetLIVLevels

実行中のDAQジョブ制限値レベルを変更します。

```
  EA_DAQ.SetLIVLevels(ByVal Name As String, ByVal AlarmLevel As Double, ByVal
WarnLevel As Double, Optional ByVal AlarmColor As Long, Optional ByVal WarnColor As
Long) As Long
```

ここで、Nameは、制限値の作成時に定義された制限値条件の名前を指定します。

引数	説明
Name	制限値名。
AlarmLevel	警報状態のレベル。
WarnLevel	警告状態のレベル。
AlarmColor	警報色。警報時にLEDや曲線に使用する色。例えばvbRedまたはRGB(255,0,0)。
WarnColor	警告色。警告時にLEDや曲線に使用する色。例えばvbYellowまたはRGB(255,255,0)。

戻り値	説明
0	エラーなし。
<0	エラー。制限値が存在しないなど。

例

```
  'Sets for example the alarm limit value to 1.75 volt, and warning limit value to
1.5 volt.
  EA_DAQ.SetLIVLevels("Measuring range OVFL positive", 1.75, 1.5, vbRed, vbYellow)
```

EA_DAQ.SetTempStore

現在のチャネルのすべての計測値を、一時計測ストレージに上書きします。この関数はまた、任意のチャネルを一時ストレージに書き込むことができます。

対話型モードに相当する機能がありません。Post-process graphの更新時にcatmanEasy/APが使用します。

```
  EA_DAQ.SetTempStore(ByVal Channel As Variant, ByVal Count As Long, Buffer() As
Double, Optional ByVal StartIndex As Long) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
Count	書き込むサンプル数。
Buffer	値を書き込むバッファ。バッファ内の最初の値は、最も古い計測値にあたります。
StartIndex	任意です。書き込み操作の開始位置(最初のサンプルはインデックス0です)。 省略した場合、インデックス0から開始します。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- VBAの"Option Base"ステートメンでは、モジュールレベルで配列の宣言に添字を指定できますが、catman EasyScriptでは使用できません! EasyScriptでは必ず0であり、"Option Base"ステートメントは必要とされません。
- 一時計測ストレージは、DAQジョブを開始(または反復)する前に自動的にクリアされます。
- 元の計測により生成した値より*Count*が少ない場合、後の値は失われます。書き込みには、*Count*で指定した値分だけが含まれています。
- CountがBuffer内の値より多い場合、バッファ内に存在する値のみを書き込み、戻り値にエラーコードを含みます。必ず、転送したいデータに対して十分な一時格納を作成します。一時格納がどのように構成されているかは、Check OPTIONS/DATA STORAGEを確認してください。"Fixed size"として構成されている場合、"Max. number of samples per channel"が十分かどうかを確認する必要があります! Fixed sizeとして構成されていない場合、データを保持するため、自動的に拡張します。
- 計測チャネルとは異なるチャネルにデータを追加する場合、最後の計測または演算チャネルよりも高いチャネル番号を使用してください。例えば一時保管のチャネル最大数が200の場合(OPTIONS/STORAGEを参照)、191、192、193 ....と使用します。
- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
Sub RemoveOutliers()
  Dim Buffer() As Double
  Dim nSamples As Long
  nSamples = EA_DAQ.GetTotalSampleCount("Valve pressure")
  ReDim Buffer(nSamples-1)
  EA_DAQ.GetTempStore("Valve pressure", Buffer)
  For i=0 To nSamples-1
    If Buffer(i) > 5000 Then Buffer(i) = Buffer(i-1)
  Next
  EA_DAQ.SetTempStore("Valve pressure", nSamples, Buffer)
End
```

EA_DAQ.StatJournalFile

統計ジャーナルファイル名を設定、または取得します。

```
Property EA_DAQ.StatJournalFile
```

例

```
Sub ChangeStatJournalFileName()
  EA_DAQ.StatJournalFile = "C:\MYDATA\%Job%_%DateTime%.stat" 'Change file name for
statistics journal.
End Sub

Sub RestoreStatJournalFileName()
  EA_DAQ.StatJournalFile = "" 'Restore the default file name for statistics journal.
End Sub
```

注記

- 統計ジャーナルのファイル名に%DateTime%などのプレースフォルダを使用することができます。
- 統計ジャーナルのファイル名をデフォルトに戻すには、空の文字列を割り当てます。

EA_DAQ.TempStoreControl

ストレージモード**Manual control**での一時計測保管内の値の扱いを決定します。

対話型モードでは、ツールバーの**Manual control**ボタンに相当します(*Manual control*が使用されている場合、ツールバーのみアクセス可能です)。

```
EA_DAQ.TempStoreControl(ByVal Mode as Integer) As Long
```

引数	説明
Mode	メモリモード: 0 = Snapshot (すべての有効チャネルの単一計測値を保存) 1 = Undo last snapshot (最後に保存した値をクリア) 2 = Activates continuous saving(継続的な保存開始) 3 = Deactivates continuous saving(継続的な保存を解除)

戻り値	説明
0	エラーなし。
<0	エラー。計測が実行していないなど。

注記

- この関数は、データ収集を実行していないと効果がありません。
- この関数はまた、現在のDAQジョブEA_Job.GetProperty(*PropIndex*, 24)のデータ保管モードがが0(すべてのデータを保存)の場合も効果はありません。
- 一時計測ストレージは、DAQジョブを開始(または反復)する前に自動的にクリアされます。

EA_DAQ.TriggerEvent

現在、プロジェクトで定義されているイベントや制限値トリガを実行します。現在のイベント条件にかかわらず、手動でイベントをトリガしたい場合に便利です。

EA_DAQ.SetLIVLevelsも参照してください:

```
  EA_DAQ.TriggerEvent(ByVal EventName As String, Optional ByVal TriggerImmediately
As Integer) As Long
```

引数	説明
EventName	イベント名。
TriggerImmediately	オプションです。1に設定すると、DAQ が実行されていない場合、指定されたイベントが直ちにトリガされます。これは、実行中の DAQから独立したイベントに役立ちます。

戻り値	説明
0	エラーなし。
<0	エラー。イベントが存在しないなど。

注記

- イベントおよび制限値構成ダイアログでは、*EA_DAQ.TriggerEventコマンド*を排他的に反応するイベント"Script (EA_DAQ.TriggerEvent)"を取得することができます。

例

```
Sub MonitorValue()
  Dim MV As Double
  EA_DAQ.CurrentReading("Valve pressure", MV, DAQ_MAX)

  If MV > 2.75 Then
    EA_DAQ.TriggerEvent "Measuring range OVFL positive"
  End If
End Sub

Sub MonitorCX22PowerLoss()
  Dim UPSState As Double
  UPSState = EA_IO.CX22GetUPSState

  If UPSState = 5 Or UPSState = 13 Then
    EA_DAQ.TriggerEvent "Send notification", 1
  End If
End Sub
```

EA_DAQ.UDPStart

UDP計測の出力を開始します。

対話型モードでは、オプション**DAQ Jobs** → **Advanced, Remote**タブでUDP出力の設定および有効化します。

```
  EA_DAQ.UDPStart(ByVal BroadcastType As Integer, ByVal Port As Long, Optional ByVal
Format As Integer, Optional ByVal IPAddress As String) As Long
```

引数	説明
`BroadcastType`	UDPブロードキャストの種類: `0` = UDP_SEGMENT_D `1` = UDP_SEGMENT_C `2` = UDP_SINGLE_ADDRESS
`Port`	ポート番号。受信側クライアントPCで指定しているポート。
`Format`	任意です。UDPパケットの形式。
`IPAddress`	任意です。受信側PCのIPアドレス。引数は*BroadcastType*でUDP_SINGLE_ADDRESSでのみ使用可。

戻り値	説明
`0`	エラーなし。
`<0`	エラー。

注記

- この関数は、任意のUDP出力を停止し、再起動します。
- 引数`BroadcastType`により、計測データはネットワーク全体のDセグメント(この場合、受信するクライアントPCだけでなくcatmanEasy/AP serverもサブネットマスクが255.255.255.0である必要があります)または、ネットワークのCセグメント(サブネットマスクは255.255.0.0が必要)に単一IPアドレスに送信します。
- `Port`は任意に選択することができます - ただし、すべてのクライアントがUDPパケットをListenするためにこのポート番号を使用する必要があります。10000以上のポート番号を使用することを推奨しています。

EA_DAQ.UDPStop

UDP計測の出力を停止します。

対話型モードでは、オプション**DAQ Jobs** → **Advanced, Remote**タブでUDP出力の設定および有効化します。

```
EA_DAQ.UDPStop() As Long
```

戻り値	説明
0	エラーなし。
<0	エラー。

EA_DataView

EA_DataViewオブジェクトはDataViewウィンドウのいくつかの基本的な関数にアクセスするために使用します。
このオブジェクトでは以下が行えます:

- DataViewウィンドウにチャネルを追加/削除: EA_DataView.Add、EA_DataView.Remove
- 単一またはすべてのDataViewウィンドウを更新: EA_DataView.Refresh、EA_DataView.RefreshAll

このオブジェクトでは以下のメソッドが公開されています:

EA_DataView.Add ・・・

EA_DataView.Add

読み込んだテストまたは演算をDataViewウィンドウのチャネルに割り当てます。

```
  EA_DataView.Add(ByVal ViewIndex As Integer, ByVal Test As Variant, ByVal
ChannelName as String) As Long
```

引数	説明
ViewIndex	チャネルを割り当てるDataViewウィンドウのインデックス。最初のウィンドウは1です。
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 Testに空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
ChannelName	割り当てるチャネル名。

戻り値	説明
0	エラーなし。
<0	エラー。DataViewウィンドウが存在しないなど。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。

例

```
Sub CustomizeDataView()
  EA_Test.Load "C:\MyTests\Test1.TST"
  EA_DataView.Add(1, 1, "Distance")
   ...
  EA_DataView.Remove(1, "Distance")
End Sub
```

EA_DataView.Refresh

指定したDataViewウィンドウのテーブルおよびプレビューグラフを更新します。

```
EA_DataView.Refresh(ByVal ViewIndex As Integer) As Long
```

引数	説明
ViewIndex	更新するDataViewウィンドウのインデックス。最初のウィンドウは1です。

戻り値	説明
0	エラーなし。
<0	エラー。DataViewウィンドウが存在しないなど。

注記

- 割り当てたチャネルのデータが更新された場合、すなわち新しいパラメータで再計算した場合に、更新が必要です。

例

```
Sub UpdateInterpolation()
  EA_Test.Interpolate 1, 0, 1, "Angle", 0, 200, 0, 0, 0, "Angle_200"
  EA_DataView.Refresh 1
End Sub
```

EA_DataView.RefreshAll

すべてのDataViewウィンドウのテーブルおよびプレビューグラフを更新します。

```
EA_DataView.RefreshAll
```

注記

- 割り当てたチャネルのデータが更新された場合、すなわち新しいパラメータで再計算した場合に、更新が必要です。

例

```
Sub UpdateInterpolation()
  EA_Test.Interpolate 1, 0, 1, "Angle", 0, 200, 0, 0, 0, "Angle_200"
  EA_DataView.RefreshAll
End Sub
```

EA_DataView.Remove

DataViewウィンドウからチャネルを削除します。

```
EA_DataView.Remove(ByVal ViewIndex As Integer, ByVal ChannelName as String) As Long
```

引数	説明
ViewIndex	チャネルを割り当てるDataViewウィンドウのインデックス。最初のウィンドウは1です。
ChannelName	削除するチャネル名。

戻り値	説明
0	エラーなし。
<0	エラー。DataViewウィンドウが存在しないなど。

例

```
Sub CustomizeDataView()
  EA_Test.Load "C:\MyTests\Test1.TST"
  EA_DataView.Add(1, 1, "Distance")
   ...
  EA_DataView.Remove(1, "Distance")
End Sub
```

EA_Graph

多数のオプションがグラフ(図表)で利用可能なため、この専用オブジェクトが存在します。

このオブジェクトでは、以下が行えます:

- タイトルや色など、グラフの一般的なプロパティを設定/取得: EA_Graph.GetProperty、EA_Graph.SetProperty
- 軸のプロパティを設定: EA_Graph.SetAxisProperty
- プロットのプロパティを設定/取得: EA_Graph.GetPlotProperty、EA_Graph.SetPlotProperty
- 単一/すべてのプロットを削除: EA_Graph.RemovePlot、EA_Graph.ClearPlots
- グラフを印刷、ファイルにエクスポート、クリップボードにコピーEA_Graph.PrintOut、EA_Graph.Export、EA_Graph.SetExportProperties
- 単一/すべてのグラフを再描画: EA_Graph.Refresh、EA_Graph.RefreshAll、EA_Graph.ImmediateRefresh
- すべてのグラフのカーソルモードをアクティブ/非アクティブ化: EA_Graph.CursorActivation
- カーソル位置の座標を設定/取得: EA_Graph.GetCursorPosEx、EA_Graph.SetCursorPosEx
- ビデオプレーヤーと表示器の時間をグラフ内のカーソルと同期: EA_Graph.SyncAllCursors
- Live-FFT graphのチャネルスペクトルから位置およびピークの振幅を取得: EA_Graph.GetFFTPeaks
- ヒストグラムグラフのビン数を取得: EA_Graph.GetHistogramCounts
- 多項式の最適曲線の係数を取得: EA_Graph.GetFitParameters
- 注釈を追加/削除: EA_Graph.AddAnnotation、EA_Graph.AddCursorGraphAnnotation、EA_Graph.ClearAnnotations
- マップ関数: EA_Graph.MapSetPosition、EA_Graph.MapSetCustomURL、EA_Graph.MapConvertGPXToTextFile

DAQモードではEA_Panelを使用し、解析モードではEA_Testを使用してプロットを割り当てます。

EA_Graphオブジェクトはまた、配列からグラフにデータをプロットするメソッドのセットを提供します:

- EA_Graph.PlotArrayXY
- EA_Graph.ChartArrayXY
- EA_Graph.PlotArrayY
- EA_Graph.PlotArrayMatrix

解析モードで、既存のテストからチャネルデータをプロットする場合、これらメソッドは必要ありません - *EA_Test* オブジェクトが適切なプロット関数を提供します。
カスタムデータのグラフ表示、ローレベルのグラフプログラミングも参照してください。

注記

上級プログラマは、下層のグラフオブジェクトに直接アクセスおよび制御することができます(例えばcatmanのチャネルに関連しない配列データをプロット)。EA.GetObjectメソッドの例を参照してください。

このオブジェクトでは以下のメソッドが公開されています:

EA_Graph.AddAnnotation ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 263
EA_Graph.AddCursorGraphAnnotation ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 265
EA_Graph.ChartArrayXY ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 267
EA_Graph.ClearAnnotations ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 270
EA_Graph.ClearPlots ・・・ 271
EA_Graph.CursorActivation ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 272
EA_Graph.Export ・・・ 273
EA_Graph.GetCursorPos ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 275
EA_Graph.GetCursorPosEx ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 276
EA_Graph.GetFFTPeaks ・・ 277

EA_Graph.GetFitParameters ・・・

EA_Graph.AddAnnotation

グラフに新規注釈を追加します。

カーソルグラフでは、より強力な注釈をサポートしている、EA_Graph.AddCursorGraphAnnotationを使用します。

また、例えば注釈を持つ演算データといった、カスタムプロットを作成することができます。この場合、引数の*Text*へは空の文字列を設定します。このアプローチは、大量でないデータセット(< 1000)に適しています。

```
EA_Graph.AddAnnotation(ByVal Panel As Variant, ByVal GraphName As String, ByVal
Axis As Integer, ByVal X As Double, ByVal Y As Double, ByVal AnnoType As Integer,
ByVal Text As String, ByVal TextSize As Single, ByVal SymbolSize As Single, ByVal
ForeColor As Long, ByVal BackColor As Long) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、*すべてのパネル*のグラフを検索することを意味します。
GraphName	グラフ名。
Axis	軸座標は以下を参照してください: 0 = y軸の最初のレイヤ 1 = y軸の2番目のレイヤ 2 = y軸の3番目のレイヤ 3 = 右y軸の最初のレイヤ 4 = 右y軸の2番目のレイヤ 5 = 右y軸の3番目のレイヤ
x,y	注釈の座標。注釈はこの点を中心とします。
AnnoType	注釈の種類: 0 = テキストのみ 37 = アンカー付きテキスト 15 = テキスト+丸 16 = テキスト+塗りつぶし丸 17 = テキスト+四角 18 = テキスト+塗りつぶし四角 19 = テキスト+菱形 20 = テキスト+塗りつぶし菱形 21 = テキスト+三角 22 = テキスト+塗りつぶし三角 92 = テキスト+矢印N 94 = テキスト+矢印E 96 = テキスト+矢印S 98 = テキスト+矢印W 93 = テキスト+矢印NE 97 = テキスト+矢印SW 99 = テキスト+矢印NW 95 = テキスト+矢印SE
Text	注釈のテキスト。
TextSize	6 ... 30 テキストサイズの設定は、すべてのグラフの注釈に適用されます。
SymbolSize	1 ... 10 シンボルサイズの設定は、すべてのグラフの注釈に適用されます。
ForeColor	テキストおよびシンボルの色。
BackColor	背景色。色設定は、すべてのグラフの注釈に適用されます。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

注記

- EA_Graph.ImmediateRefreshを*True*に設定すると、変更がすぐに表示されます。それ以外の場合はEA_Graph.Refreshでグラフを再描画します。

- 複数のグラフが同じ名前で、すべてのパネルを検索(*Panel* = 0)する場合、最初に見つかったグラフを使用します。

例

```
  Sub CustomizeGraph()
    EA_Graph.ImmediateRefresh=False
    EA_Graph.ClearAnnotations(1, "RTGRAPH_1")
    EA_Graph.AddAnnotation(1, "RTGRAPH_1", 0, 100, 125, ANNOT_TEXTONLY, "Watch point
1", 10, 4, vbBlack, vbWhite)
    EA_Graph.Refresh(1, "RTGRAPH_1")
  End Sub

  Sub CustomizeCursorGraph()
    Dim anno As CursorGraphAnnotation
    With anno
      .CoordinateType = CG_AXIS
      .X1 = 4
      .Y1 = 6
      .Style = CG_TEXT
      .Text = "Watch point 1"
      .FontSize = 10
      .ForeColor = vbBlack
      .BackColor = vbWhite
    End With
    EA_Graph.ImmediateRefresh=False
    EA_Graph.ClearAnnotations(1, "PPCURSORGRAPH_1")
    'Use AddCursorGraphAnnotation instead of AddAnnotation to add an annotation to
cursor graph.
    EA_Graph.AddCursorGraphAnnotation(1, "PPCURSORGRAPH_1", anno)
    EA_Graph.Refresh(1, "PPCURSORGRAPH_1")
  End Sub
```

EA_Graph.AddCursorGraphAnnotation

カーソルグラフに新規注釈を追加します。

⚠ このメソッドは標準グラフに適用されません。EA_Graph.AddAnnotationを使用してください。

```
  EA_Graph.AddCursorGraphAnnotation(ByVal Panel As Variant, ByVal GraphName As
String, Annotation As CursorGraphAnnotation) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	グラフ名。
Annotation	注釈を記述する*CursorGraphAnnotation*型の変数。 型のメンバーは、ほとんど説明の必要がありません。 .Style CG_TEXT = 0(テキスト) CG_RECTANGLE = 1(四角) CG_RECTANGLE_FULL = 2(塗りつぶし四角) CG_OVAL = 3(楕円) CG_OVAL_FULL = 4(塗りつぶし楕円) CG_LINE = 5(線) CG_TEXTPLUSRECTANGLE = 6(テキスト+四角) CG_ARROW = 7(矢印) .x1 = テキストのみ中心座標で、他の種類は左上隅 .y1 = テキストのみ中心座標で、他の種類は左上隅 .x2 = 右下隅 .y2 = 右下隅 すべての座標は選択した座標の種類により異なります。 .CoordinateType = 0: 座標は軸のスケーリングを参照 .CoordinateType = 1: 座標はグラフ領域(ピクセル単位)、原点の左上隅を参照 .CoordinateType = 2: 座標はプロット領域(ピクセル単位)、原点の左上隅を参照

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

注記

- EA_Graph.ImmediateRefreshを*True*に設定すると、変更がすぐに表示されます。それ以外の場合はEA_Graph.Refreshでグラフを再描画します。

- 複数のグラフが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったグラフを使用します。

例

```
Sub Main()
  Dim An As CursorGraphAnnotation
  Dim i As Integer

  An.FontName="Arial"
  An.FontSize=10
  An.CoordinateType=1
  An.Style=0
  An.ForeColor=vbBlue
  An.X1=100

  For i=1 To 10
    An.Y1=i*50
    An.Text = "Zeile " & i
    EA_Graph.AddCursorGraphAnnotation 1, "PPCURSORGRAPH_1", An
  Next
End Sub
```

EA_Graph.ChartArrayXY

グラフに新規データブロックを追加、すなわちスクロールするストリップチャートを実現します。

このメソッドは標準およびカーソルグラフに適用されます。

```
  EA_Graph.ChartArrayXY(ByVal Panel As Variant, ByVal GraphName As String, ByVal
PlotIndex As Integer, ByVal nPoints As Long, XData() As Double, YData() As Double) As
Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。*Panel* = 0の場合、*すべての*パネルのグラフを検索することを意味します。
GraphName	グラフ名。
PlotIndex	標準グラフ: 引数は使用しません。*ChartArrayXY*は**すべての**プロットのデータを設定します。詳細は引数**XData**を参照してください。
	カーソルグラフ: プロット(曲線)を必要に応じて作成します。つまり、*PlotIndex*が現在のプロット数より1つ大きい場合、追加のプロットを1つ作成します。*PlotIndex*で示すプロットがすでに存在する場合、このプロットのデータと交換します。最初のプロットのインデックスは1です。 2番目のy軸レイヤに描画することを示すには*PlotIndex*に100を追加し、3番目のレイヤに描画することを示すには1000を追加します。 グラフは適切なレイヤ数のために、対話的に設定する必要があります: EasyScriptは現在、グラフのレイヤ構成をサポートしていません。
nPoints	追加するデータポイント数。xおよびy配列は、少なくともこのサイズの設定が必要です! カーソルグラフの配列は*nPoints*-1になるよう、正確な要素数にする必要があります。そうしないと、原点と最後のデータポイントを結ぶ線が表示されます。 配列のプロットは常に非圧縮で行われます。チャートが保持できるポイントの最大数は、グラフに設定された非圧縮データポイントの最大数に依存します(追加する*nPoints*値は、通常これよりも小さくなります)。この値は、Graph Configurationタブの"Advanced"により対話的に、またはスクリプトのEA_Graph.SetPropertyで設定することができます。プロットの最大数および非圧縮の最大数によるメモリ消費量の説明は、後者のメソッドのヘルプを参照してください。
XData	xデータ(nPoints)を含む配列。この引数は、標準グラフとカーソルグラフでは大きく異なります。
	標準グラフ: *PlotArrayXY*メソッドとは対照的(専用のプロットまたは曲線にデータを入力)に*ChartArrayXY*は、常にプロットする**すべての**データを要求するよう構成されています(グラフ毎のプロットの最大数は*EA_Graph.SetProperty*で設定します)。例えばグラフが30プロット(デフォルト)の場合で、データポイントを10追加するには以下のように配列を編成する必要があります: XData(0)....XData(9) = 最初のプロットデータ(curve) XData(10)....XData(19) = 二番目のプロットデータ(curve) .... XData(290)....XData(299) = 最後のプロットデータ(curve) *XNullValue*(通常は0)で使用していないプロットを配列要素に取り込みます。これらプロットは描画されません。
	カーソルグラフ: カーソルグラフのデータの取り扱いは*EA_Graph.PlotArrayXY*と同じで容易です。XDataには単一のプロットデータのみが含まれます。

YData	yデータ(nPoints)を含む配列。
	標準グラフ: 配列の構成は*XData*で説明したものと同じです。*YNullValue*(通常は0)で使用していないプロットを配列要素を設定します。これらプロットは描画されません。
	カーソルグラフ: カーソルグラフのデータの取り扱いは*EA_Graph.PlotArrayXY*と同じで容易です。XDataには単一のプロットデータのみが含まれます。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

注記

- このメソッドを使用する場合、catman組み込みのグラフの再描画でプロットが駄目にならないよう、注意する必要があります。配列データのプロット時は、以下に注意してください:

  ⚠ 同一グラフ内に通常のデータチャネルと配列データを混在させないでください(つまり、同一グラフに*EA_Panel.SetRTDisplayChannel*と*EA_Graph.ChartArrayYX*は使用しないでください)。これは、ユーザが配列のチャートに使用するグラフに対して、チャネルをドラッグすべきでないことを意味します。

  ⚠ 配列のチャートは常に非圧縮で行われます。配列あたりのポイントの最大数は、非圧縮データポイントの最大数に依存します(Graph Configurationタブの"Advanced"を参照または、EA_Graph.SetPropertyで設定)。

- プロットの更なる属性の定義(凡例や線種など)にはEA_Graph.SetPlotPropertyを使用します。

例

```
Sub AppendDataToGraph4Plots()
  Dim RetVal As Long
  Dim xdata(200) As Double
  Dim ydata(200) As Double
  Dim SampleCount As Integer
  SampleCount = 200

  'Plot 1:
  For n=0 To (SampleCount/4)-1
    xdata(n) = n
    ydata(n) = n^2
  Next

  'Plot 2:
  For n=(SampleCount/4) To (SampleCount/2)-1
    xdata(n) = n-50
    ydata(n) = (n^2)-2500
  Next

  'Plot 3:
  For n=(SampleCount/2) To ((SampleCount/4)*3)-1
    xdata(n) = n-100
    ydata(n) = n*40
  Next
```

```
  'Plot 4:
  For n=(SampleCount/4)*3 To (SampleCount)-1
    xdata(n) = n-150
    ydata(n) = n*3
  Next

  RetVal = EA_Graph.SetProperty(1,"PPGRAPH_2",GRAPH_MAXPLOTS,4)
  RetVal = EA_Graph.SetProperty(1,"PPGRAPH_2",GRAPH_MAXPOINTSUNCOMPRESSED, 2000)
  RetVal = EA_Graph.SetProperty(1,"PPGRAPH_2",GRAPH_TIMEWINDOW, 60)
  RetVal = EA_Graph.ChartArrayXY(1,"PPGRAPH_2",1, 50, xdata, ydata)
End Sub
```

EA_Graph.ClearAnnotations

グラフからすべての注釈を削除します。

```
  EA_Graph.ClearAnnotations(ByVal Panel As Variant, ByVal GraphName As String) As
Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、*すべてのパネル*のグラフを検索することを意味します。
GraphName	グラフ名。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

注記

- EA_Graph.ImmediateRefreshを*True*に設定すると、変更がすぐに表示されます。それ以外の場合はEA_Graph.Refreshでグラフを再描画します。

- 複数のグラフが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったグラフを使用します。

例

```
  Sub CustomizeGraph()
    EA_Graph.ImmediateRefresh=False
    EA_Graph.ClearAnnotations(1, "RTGRAPH_1")
    EA_Graph.AddAnnotation(1, "RTGRAPH_1", 0, 100, 125, ANNOT_TEXTONLY, "Watch point
1", 10, 4, vbBlack, vbWhite)
    EA_Graph.Refresh(1, "RTGRAPH_1")
  End Sub
```

EA_Graph.ClearPlots

グラフからすべてのプロットを削除します。

```
EA_Graph.ClearPlots(ByVal Panel As Variant, ByVal GraphName As String) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	グラフ名。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

注記

- EA_Graph.ImmediateRefreshを*True*に設定すると、変更がすぐに表示されます。それ以外の場合はEA_Graph.Refreshでグラフを再描画します。

- 複数のグラフが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったグラフを使用します。

例

```
Sub ConfigureGraph()
  EA_Graph.ClearPlots 1, "GRAPH1"
  EA_Graph.ImmediateRefresh=False
  EA_Graph.SetProperty 1, "GRAPH1", 1, "Test run A-6" 'Main title.
  EA_Graph.SetProperty 1, "GRAPH1", 2, "Tester: John Smith" 'Subtitle.
  EA_Test.PlotYT 1, "Valve pressure", 1, "GRAPH1", 0, vbRed
  EA_Test.PlotYT 1, "Temperature", 1, "GRAPH1", 0, vbBlue
  EA_Graph.Refresh 1, "GRAPH1"
End Sub
```

EA_Graph.CursorActivation

プロジェクト内のすべてのグラフのカーソルモードを有効/無効にします。

```
EA_Graph.CursorActivation(ByVal Mode As Integer) As Long
```

引数	説明
Mode	0=カーソルモードは無効 1=カーソルモードは有効

EA_Graph.Export

グラフをファイルにエクスポートしたり、クリップボードにコピーします。可能な画像形式はメタファイル、ビットマップ、JPEGおよびPNGです。EMF(拡張メタファイル)形式を推奨しています。一般的に、MS Wordにグラフを挿入するときに最良の結果が得られます。
グラフの印刷はEA_Graph.PrintOutを使用します。

```
EA_Graph.Export(ByVal Panel As Variant, ByVal GraphName As String, ByVal Format As Integer, Optional ByVal FileName As String, Optional ByVal Width As Long, Optional ByVal Height As Long) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、*すべてのパネル*のグラフを検索することを意味します。 このメソッドは*スコープ*および*フローティングパネル*でも動作します - この場合にはLive-FFTを含むレイヤのレイヤ名を示します。
GraphName	グラフ名。
Format	作成する画像形式と書き出し先: 0 = メタファイル形式でファイルにエクスポート 1 = ビットマップ形式でファイルにエクスポート 2 = JPEG 形式でファイルにエクスポート 3 = メタファイル形式でクリップボードにコピー 4 = ビットマップ形式でクリップボードにコピー 5 = JPEG形式でクリップボードにコピー 6 = EMF(拡張メタファイル)をファイルに変換 7 = EMF(拡張メタファイル)をファイルに変換 8 = PNGをファイルに変換 9 = PNGをクリップボードに変換 10 = SVGをファイルに変換 11 = SVGをクリップボードに変換 カーソルグラフおよび3Dグラフは0、3、6および7のみの形式が有効です。
FileName	ファイルにエクスポートする場合のみ: 作成する画像ファイル名(およびパス)。パスを指定していない場合、現在のフォルダを使用します。特定のパスを設定する場合、ChDir関数を使用します。
Width	TWIPS(1 inch = 1440 TWIPS)単位の出力の幅。省略した場合、グラフの元の幅を使用します。
Height	TWIPS(1 inch = 1440 TWIPS)単位の出力の高さ。省略した場合、グラフの元の高さを使用します。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

注記

- 複数のグラフが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったグラフを使用します。
- エクスポートの詳細なプロパティ設定(EMFの種類や優先出力など)はEA_Graph.SetExportPropertiesメソッドを使用します。詳細なエクスポートスタイルは標準グラフでのみ使用できることに注意してください。カーソルグラフとスコープパネルまたはフローティングパネルのグラフは、これらをサポートしていません。

例

```
  'The following example exports a graph to the clipboard and subsequently inserts
it at a special bookmark in a MS Word document:
  Sub ExportGraph()
    Dim objWrd As Object
    Dim bkm As Variant
    Set objWrd = CreateObject("Word.Application")
    objWrd.Visible = True
    objWrd.Documents.Open FileName:="C:\MyFiles\MyDoc.doc"
    EA_Graph.Export 1, "RTGRAPH_1", EMF_TO_CLIPBOARD
    objWrd.Documents(1).Bookmarks("BK_1").Range.Paste 'In this demo a bookmark with
name "BK_1" must be inserted in Word document before.

    'Alternatively you may use the "Bookmark" Property of a graph To find the
bookmark:
    EA_Graph.GetProperty 1, "RTGRAPH_1", GRAPH_BOOKMARK, bkm
    objWrd.Documents(1).Bookmarks(bkm).Range.Paste
  End Sub
```

EA_Graph.GetCursorPos

現在のカーソル位置の座標を取得します。

この関数は廃止されました。バージョン3.3以降で新しく導入された、グラフ内の複数カーソルをサポートするメソッドEA_Graph.GetCursorPosExメソッドの使用を推奨しています。

```
  EA_Graph.GetCursorPos(ByVal GraphName As String, x As Double, y As Double,
CurveIndex As Integer, CurveName As String) As Long
```

引数	説明
GraphName	グラフ名。
x	カーソルのx座標。
y	カーソルのy座標。
CurveIndex	グラフに複数の曲線が含まれている場合、カーソルにスナップする曲線のインデックス。最初の曲線は1です。
CurveName	グラフに複数の曲線が含まれている場合、カーソルにスナップする曲線の名前。

EA_Graph.GetCursorPosEx

現在のカーソル位置の座標だけでなく、接続している曲線の情報も取得します。

```
  EA_Graph.GetCursorPosEx(ByVal GraphName As String, ByVal CursorIndex As Integer, x
As Double, y As Double, CurveIndex As Integer, CurveName As String) As Long
```

引数	説明
GraphName	グラフ名。
CursorIndex	カーソルの指定(最初のカーソルは1)。バージョン3.3より導入された新しいカーソルグラフオブジェクトは、複数のカーソルをサポートしています。標準グラフの場合、この引数は1でなければなりません。
x	カーソルのx座標。
y	カーソルのy座標。
CurveIndex	グラフに複数の曲線が含まれている場合、カーソルにスナップする曲線のインデックス。最初の曲線は1です。
CurveName	グラフに複数の曲線が含まれている場合、カーソルにスナップする曲線の名前。

例

```
  Sub GetCursorInfo()
    Dim xPos As Double, yPos As Double
    Dim indexCurve As Integer
    Dim nameCurve As String
    EA_Graph.GetCursorPosEx("PPGRAPH_1", 1, xPos, yPos, indexCurve, nameCurve)
  End Sub
```

EA_Graph.GetFFTPeaks

*Live-FFT*グラフのチャネルスペクトルからピークの位置と振幅を取得します。

注記: このメソッドはLive-FFT graphにのみ適用できます!

```
  EA_Graph.GetFFTPeaks(ByVal Panel As Variant, ByVal GraphName As String, ByVal
Channel As Variant, outNofPeaks As Integer, outX() As Single, outY() As Single) As
Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、*すべて*のパネルのグラフを検索することを意味します。 このメソッドは*スコープ*および*フローティングパネル*でも動作します - この場合にはLive-FFTを含むレイヤのレイヤ名を示します。
GraphName	グラフ名。
Channel	チャネル名。 Live-FFT表示がチャネルモードで*ベクトルの和*で動作する場合、名前はチャネルを構成する和、または空のままにすることができます。
outNofPeaks	見つかったピーク数。
outX	Hz単位のピークを受け取る位置(インデックス は0から開始)。配列のサイズはピーク数に基づいて自動的に決定されます。
outY	ピークの振幅を受け取る配列(インデックス は0から開始)。配列のサイズはピーク数に基づいて自動的に決定されます。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しない、グラフがLive-FFTグラフではないなど。

注記

- ピーク検出はLive-FFTグラフに定義された引数の"Threshold"および"Width"に強く影響されます(グラフのConfiguration windowの"General"タブ(標準パネル)または"FFT"タブ(フローティングやスコープパネル)を参照)。少なくとも"Width"データポイントが"Threshold"を超えるピークのみが有効なピークと見なされます。

- 出力配列は、最低周波数から最高出力の順に並べ替えます。

EA_Graph.GetFitParameters

多項式の最適曲線(最大4次)の係数を取得します。

```
  EA_Graph.GetFitParameters(ByVal Panel As Variant, ByVal GraphName As String, ByVal
Channel As Variant, Params() As Double) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	グラフ名。
Channel	チャンネル名またはインデックス。
Params	配列には、yデータに最もよく適合する次数nの多項式p(x)の係数が含まれています。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

EA_Graph.GetHistogramCounts

ヒストグラムグラフのビン数を取得します。

```
  EA_Graph.GetHistogramCounts(ByVal Panel As Variant, ByVal GraphName As String,
ByVal Channel As Variant, outNofBins As Integer, outX() As Single, outY() As Single)
As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	グラフ名。
Channel	チャンネル名。
outNofBins	ビン数。
outX	ビン値を受け取る位置(インデックス は0から開始)。配列のサイズはビン数に基づいて自動的に決定されます。
outY	各ビンの周波数を受け取る配列(インデックスは0から開始)。配列のサイズはビン数に基づいて自動的に決定されます。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはグラフが存在しないか、グラフがヒストグラムグラフではありません。

EA_Graph.GetPlotProperty

グラフ内のプロットのプロパティを取得します。

```
  EA_Graph.GetPlotProperty(ByVal Panel As Variant, ByVal GraphName As String, ByVal
Plot As Variant, ByVal PropIndex As Integer, PropVal As Variant) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、*すべてのパネル*のグラフを検索することを意味します。
GraphName	グラフ名。
Plot	プロパティを設定するプロット。グラフの最初のプロットの位置は1です。プロットは、名前または位置で指定できます。 ⚠ *EA_Graph.PlotArrayXY*または*EA_Graph.ChartArrayXY*を介して作成されたプロットは、位置インデックスを介してのみアドレス指定できます。
PropIndex	プロットのプロパティ。 1 = プロット名 この名前は凡例に表示され、プロットは他のメソッドでこの名前によってアドレス指定されます。デフォルトでは、データの生成元のチャネルまたは演算の名前が使用されます。 2 = プロット色 3 = プロットスタイル(*PropVal*: 1=線、2=線+点、3=点のみ(散布)、4=面積、5=棒、6=スプライン、7=最良適合曲線、8=2次最良適合曲線 、9=3次最良適合曲線、10=4次最良適合曲線) スタイル6～10は標準グラフにのみ適用され、カーソルグラフや極座標図には適用されません。 4 = 記号(*PropVal*: 0=プラス、1=十字、2=円、3=円(塗りつぶし)、4=四角、5=四角(塗りつぶし)、6=菱形、7=菱形(塗りつぶし)、8=三角、9=三角(塗りつぶし)) 5 = 線の種類(*PropVal*: 0=細線、1=破線、2=点線、3=破線+点線、4=破線+点線+点線、5=太線) 6 = プロットの凡例名 この名前は凡例にのみ表示されます。他のメソッドでプロットをアドレス指定のために使用する内部名は変更されません。 カーソルグラフの*EA_Graph.PlotArrayXY*によって作成されたプロットには適用されません。 7 = y(x)グラフの圧縮タイプ。1の場合、xデータは単調であると想定します。この場合、y(x)プロットには通常の圧縮が使用されます。 *EA_Graph.PlotArrayXY*によって作成されたプロットには適用されません。 8 = xオフセット *EA_Graph.PlotArrayXY*によって作成されたプロットには適用されません。 9 = yオフセット *EA_Graph.PlotArrayXY*によって作成されたプロットには適用されません。 10 = 1の場合、右のy軸に対するプロット カーソルグラフの*EA_Graph.PlotArrayXY*によって作成されたプロットには適用されません。
PropVal	プロパティ値。

戻り値	説明
>0	プロットのインデックス。
0	エラー。指定されたプロットが見つかりません。
<0	エラー。パネルやグラフが存在しないなど。

注記

- EA_Graph.ClearPlotsを使用して、グラフ内のすべてのプロットを一度に削除します。
- 複数のグラフが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったグラフを使用します。
- プロットスタイルに"最良適合曲線"を使用している場合、関数 EA_Graph.GetFitParametersを介してフィット係数を取得できます。

EA_Graph.GetProperty

タイトルや色など、グラフの一般的なプロパティを取得します。

```
  EA_Graph.GetProperty(ByVal Panel As Variant, ByVal GraphName As String, ByVal
PropIndex As Integer, PropVal As Variant) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	グラフ名。
PropIndex	取得するプロパティ: 1 = メインタイトル 2 = サブタイトル上 3 = サブタイトル下 4 = 秒単位の時間枠 リアルタイムグラフに使用できる値 : 0.5, 1, 2, 5, 10, 30, 60, 120, 300, 600, 1800, 3600, 7200, 18000, 36000, 54000, 86400, 172800, 259200, 345600, 432000, 518400, 604800 カーソル グラフに使用できる値: 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 2, 5, 10, 30, 60, 120, 300, 600, 1800, 3600, 7200, 18000, 36000, 54000, 86400, 172800, 259200, 345600, 432000, 518400, 604800 5 = 圧縮を開始するまでのデータポイント数。 6 = 等高線ステップ数(等高線グラフのみ) 7 = 等高線カラーテーブルのインデックス(0..7) 8 = MS Word にエクスポートするブックマーク(文字列) 9 = グラフでサポートされるプロット(曲線)の最大数 13 = グラフに割り当てられたプロット数 14 = y軸のスタック (0 = スタックなし、それ以外の場合はy軸がスタック)
PropVal	プロパティ値。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

例

```
Sub CustomizePlots()
  Dim i As Integer
  Dim nofPlots As Integer
  EA_Graph.GetProperty 1, "PPGRAPH_1", GRAPH_PLOTCOUNT, nofPlots
  For i = 1 To nofPlots
    EA_Graph.SetPlotProperty 1, "PPGRAPH_1", i, PL_STYLE , 3 'Plot type = Only points
    EA_Graph.SetPlotProperty 1, "PPGRAPH_1", i, PL_LINESTYLE, 2 'Line type = Dotted
line
    EA_Graph.SetPlotProperty 1, "PPGRAPH_1", i, PL_SYMBOL, i 'Symbol (Symbol list see
EA_Graph.SetPlotProperty)
  Next
  EA_Graph.Refresh 1, "PPGRAPH_1"
End Sub
```

EA_Graph.ImmediateRefresh

このプロパティをTrueに設定するとEA_Graph.SetProperty、EA_Graph.SetAxisPropertyやプロットの割り当て(EA_Test.PlotYT)の結果がすぐに表示されます。

同時に多くの設定変更やプロットを割り当てたい場合は*EA_Graph.ImmediateRefresh*をFalseに設定します。すべての設定を終えた後にEA_Graph.Refreshでグラフを更新します。

```
Property EA_Graph.ImmediateRefresh
```

引数	説明
True	グラフはプロパティの変更後または、プロットの割り当て後に再描画します(デフォルト)。
False	グラフは明示的にEA_Graph.Refreshで再描画する必要があります。

注記

- このプロパティは、すべてのグラフに適用されます。単一のグラフの動作を個別に設定することはできません。

例

```
Sub ConfigureGraph()
  EA_Graph.ImmediateRefresh = False
  EA_Graph.SetProperty 1, "GRAPH1", 1, "Test run A-6" 'Main title
  EA_Graph.SetProperty 1, "GRAPH1", 2, "Tester: John Smith" 'Subtitle
  EA_Graph.Refresh 1, "GRAPH1"
End Sub
```

EA_Graph.MapConvertGPXToTextFile

GPS Exchange Format(GPX)ファイルをタブ区切りのテキストファイルに変換します。

```
  EA_Graph.MapConvertGPXToTextFile(ByVal GPXFile As String, ByVal TextFile As String,
dt As Double) As Long
```

引数	説明
GPXFile	変換するGPXファイルのファイル名。
TextFile	結果のテキストファイル名。
dt	平均解決時間。

EA_Graph.MapQuery

中心座標、マップタイプ、ズームなどを含む完全なURLを指定して、マップサーバからマップを取得します。使用可能なURLの引数のリファレンスは、Google Static Maps API Developer Guideを参照してください。

⚠ この関数は廃止されました! 代わりにEA_Graph.MapSetCustomURLを使用してください。

```
  EA_Graph.MapQuery(ByVal Panel As Variant, ByVal GraphName As String, ByVal URL As
String) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	マップ名。
URL	URLクエリ。

戻り値	説明
0	エラーなし。
<0	エラー。パネルや地図が存在しないなど。

注記

- マップコンポーネントが正しく動作するために、インターネット接続が必要です。
- URLに引数"size"を含めることはできません。catmanはグラフオブジェクトのサイズに応じて、自動的に設定します。
- ユーザがグラフのサイズを変更するか、マップ構成ダイアログのアクセスすると、このメソッドで設定されたマップコンテンツがcatmanによって上書きされることに注意してください。

EA_Graph.MapSetCustomURL

catmanで生成したマップURLにリクエストパラメータを追加します。マップのロード用にcatmanで生成したマップURLには、中心座標、ズーム率、マップの幅と高さ、およびマップタイプが含まれています。カスタムURLを設定すると、このURLに追加のリクエストパラメータ(マーカ座標のリストなど)を追加できます。カスタムURLにパラメータを追加たびに、catmanはマップを更新する必要があります。
利用可能なURLの引数はGoogle Static Maps Developer Guideを参照してください。
参照: EA_Graph.MapSetPositionを使用すると、グラフ内の位置(ズームなどを含む)を自分で設定できます。

```
  EA_Graph.MapSetCustomURL(ByVal Panel As Variant, ByVal GraphName As String, ByVal
URL As String) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	マップ名。
URL	リクエストパラメータのカスタムURL。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやマップが存在しないなど。

注記

- マップコンポーネントが正しく動作するために、インターネット接続が必要です。
- カスタムURLは"&"(アンパサンド)から開始しないでください。この区切り文字は、マップサーバへのURL送信時にcatmanが自動的に挿入します。

EA_Graph.MapSetPosition

指定された中心位置に従って新しいマップ画像を読み込みます。

```
  EA_Graph.MapSetPosition(ByVal Panel As Variant, ByVal GraphName As String, ByVal
Lat As Double, ByVal Lon As Double, Optional ByVal Zoom As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	マップ名。
Lat	中心位置の緯度(-90....+90)
Lon	中心位置の経度(-180....+180)
Zoom	オプションです。グラフのズームレベルを指定します(1=最低、20=最高)。省略した場合は、グラフ設定ダイアログで現在設定されているズームレベルを使用します。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやマップが存在しないなど。

注記

- このメソッドが正しく動作するために、インターネット接続が必要です。オフラインマップモードでは利用できません。

EA_Graph.PlotArrayMatrix

行列データを滝グラフとしてプロットししします。

このメソッドは解析モードの3D/Waterfallダイアグラム にのみ、適用されます。

```
  EA_Graph.PlotArrayMatrix(ByVal Panel As Variant, ByVal GraphName As String, x() As
Single, y() As Single, z() As Single) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	グラフ名。
x	Xデータを含む配列(列)。
y	Yデータを含む配列(行)。
z	Zデータを含む配列。配列は次のように編成する必要があります: z(r1,c1)、z(r1,c2)、z(r1,c3)、....z(r2,c1)、z(r2,c2)、z(r2,c3),.... ここで*r*は行で*c*は列を示します。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

注記

- 配列**x**の次元が**m**で、配列**y**の次元が**n**である場合、配列zの次元は**m*n**でなければなりません。

EA_Graph.PlotArrayXY

データを含む配列をプロットします。二つの配列が必要です: xデータおよびyデータが一つづつ。極座標グラフに配列をプロットする場合、x配列には角度の値が含まれます。

このメソッドは標準およびカーソルグラフに適用されます。

```
  EA_Graph.PlotArrayXY(ByVal Panel As Variant, ByVal GraphName As String, ByVal
PlotIndex As Integer, ByVal nPoints As Long, XData() As Double, YData() As Double,
Optional ByVal UseRightYAxis As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、*すべての*パネルのグラフを検索することを意味します。
GraphName	グラフ名。
PlotIndex	標準グラフy(x)または極座標グラフ: グラフは常に指定したデータ数でプロット(曲線)を埋めます。*PlotIndex*は使用するデータのプロットを示します。最初のプロットの指標は1です。 グラフ毎のプロットの最大数は*EA_Graph.SetProperty*で設定します。 複数のy軸レイヤを持つグラフでは、*PlotIndex*がデータをプロットするレイヤも決定します。グラフのプロットの最大数が 30(デフォルト)で、レイヤ数が3の場合: PlotIndex = 1 -> 1番目のレイヤの最初のプロット PlotIndex = 11 -> 2番目のレイヤの最初のプロット PlotIndex = 21 -> 3番目のレイヤの最初のプロット 二つのレイヤを持つグラフの場合: PlotIndex = 1 -> 1番目のレイヤの最初のプロット PlotIndex = 16 -> 2番目のレイヤの最初のプロット グラフは適切なレイヤ数のために、対話的に設定する必要があります: EasyScriptは現在、グラフのレイヤ構成をサポートしていません。
	カーソルグラフ: プロット(曲線)を必要に応じて作成します。つまり、*PlotIndex*が現在のプロット数より1つ大きい場合、追加のプロットを1つ作成します。*PlotIndex*で示すプロットがすでに存在する場合、このプロットのデータと交換します。最初のプロットのインデックスは1です。 2番目のy軸レイヤに描画することを示すには*PlotIndex*に100を追加し、3番目のレイヤに描画することを示すには1000を追加します。 グラフは適切なレイヤ数のために、対話的に設定する必要があります: EasyScriptは現在、グラフのレイヤ構成をサポートしていません。
nPoints	プロットするデータポイント数。xおよびy配列は、少なくともこのサイズの設定が必要です! カーソルグラフの配列は*nPoints*-1になるよう、正確な要素数にする必要があります。そうしないと、原点と最後のデータポイントを結ぶ線が表示されます。 配列のプロットは常に非圧縮で行われます。チャートが保持できるポイントの最大数は、グラフに設定された非圧縮データポイントの最大数に依存します(追加する*nPoints*値は、通常これよりも小さくなります)。この値は、Graph Configurationタブの"Advanced"により対話的に、またはスクリプトのEA_Graph.SetPropertyで設定することができます。プロットの最大数および非圧縮の最大数によるメモリ消費量の説明は、後者のメソッドのヘルプを参照してください。

XData	xデータ(*nPoints*)を含む配列。
YData	yデータ(*nPoints*)を含む配列。
UseRightAxis	0または省略 = データを左のY軸に対してプロット 1 = データを右の Y 軸に対してプロット カーソルグラフではサポートしていません。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

注記

- このメソッドを使用する場合、catman組み込みのグラフの再描画でプロットが駄目にならないよう、注意する必要があります。配列データのプロット時は、以下に注意してください:

  ⚠ 同一グラフ内に通常のデータチャネルと配列データを混在させないでください(つまり、同一グラフに*EA_Test.PlotxxxI*と*EA_Graph.ChartArrayXY*は使用しないでください)。これは、ユーザが配列のチャートに使用するグラフに対して、チャネルをドラッグすべきでないことを意味します。

  ⚠ *EA_Panel.GetDirectControl*メソッドを使用すると管理権を得て、catman組み込みの描画ルーチンからの呼び出しを抑制します。*EA_Panel.ReleaseDirectControl*を呼び出すことで、catmanに制御を戻します。

- ⚠ 配列のプロットは常に非圧縮で行われます。大量のポイント(> 1000000)をプロットしようとすると、数秒かかる場合があります。

- プロットの更なる属性の定義(凡例や線種など)にはEA_Graph.SetPlotPropertyを使用します。

EA_Graph.PlotArrayY

yデータを含む配列と等距離間隔 dx をプロットします。

このメソッドは標準およびカーソルグラフに適用されます。

```
  EA_Graph.PlotArrayY(ByVal Panel As Variant, ByVal GraphName As String, ByVal
PlotIndex As Integer, ByVal nPoints As Long, ByVal xInc as Double, ByVal x0 As
Double, YData() As Double, Optional ByVal UseRightYAxis As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、*すべて*のパネルのグラフを検索することを意味します。
GraphName	グラフ名。
PlotIndex	**標準グラフy(x)**または**極座標グラフ**: グラフは常に指定したデータ数でプロット(曲線)を埋めます。*PlotIndex*は使用するデータのプロットを示します。最初のプロットの指標は1です。 グラフ毎のプロットの最大数は*EA_Graph.SetProperty*で設定します。 複数のy軸レイヤを持つグラフでは、*PlotIndex*がデータをプロットするレイヤも決定します。グラフのプロットの最大数が 30(デフォルト)で、レイヤ数が3の場合: PlotIndex = 1 -> 1番目のレイヤの最初のプロット PlotIndex = 11 -> 2番目のレイヤの最初のプロット PlotIndex = 21 -> 3番目のレイヤの最初のプロット 二つのレイヤを持つグラフの場合: PlotIndex = 1 -> 1番目のレイヤの最初のプロット PlotIndex = 16 -> 2番目のレイヤの最初のプロット グラフは適切なレイヤ数のために、対話的に設定する必要があります: EasyScriptは現在、グラフのレイヤ構成をサポートしていません。 **カーソルグラフ**: プロット(曲線)を必要に応じて作成します。つまり、*PlotIndex*が現在のプロット数より1つ大きい場合、追加のプロットを1つ作成します。*PlotIndex*で示すプロットがすでに存在する場合、このプロットのデータと交換します。最初のプロットのインデックスは1です。 2番目のy軸レイヤに描画することを示すには*PlotIndex*に100を追加し、3番目のレイヤに描画することを示すには1000を追加します。 グラフは適切なレイヤ数のために、対話的に設定する必要があります: EasyScriptは現在、グラフのレイヤ構成をサポートしていません。
nPoints	プロットするデータポイント数。xおよびy配列は、少なくともこのサイズの設定が必要です! カーソルグラフの配列は*nPoints*-1になるよう、正確な要素数にする必要があります。そうしないと、原点と最後のデータポイントを結ぶ線が表示されます。 配列のプロットは常に非圧縮で行われます。チャートが保持できるポイントの最大数は、グラフに設定された非圧縮データポイントの最大数に依存します(追加する*nPoints*値は、通常これよりも小さくなります)。この値は、Graph Configurationタブの"Advanced"により対話的に、またはスクリプトのEA_Graph.SetPropertyで設定することができます。プロットの最大数および非圧縮の最大数によるメモリ消費量の説明は、後者のメソッドのヘルプを参照してください。

xInc	x増分。
x0	x開始値。
YData	yデータを含む配列(*nPoints*)
UseRightAxis	0または省略 = データを左のY軸に対してプロット 1 = データを右の Y 軸に対してプロット カーソルグラフではサポートしていません。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

注記

- このメソッドを使用する場合、catman組み込みのグラフの再描画でプロットが駄目にならないよう、注意する必要があります。配列データのプロット時は、以下に注意してください:

  同一グラフ内に通常のデータチャネルと配列データを混在させないでください(つまり、同一グラフに*EA_Test.Plotxxx1*と*EA_Graph.ChartArrayY*は使用しないでください)。これは、ユーザが配列のチャートに使用するグラフに対して、チャネルをドラッグすべきでないことを意味します。

  *EA_Panel.GetDirectControl*メソッドを使用すると管理権を得て、catman組み込みの描画ルーチンからの呼び出しを抑制します。*EA_Panel.ReleaseDirectControl*を呼び出すことで、catmanに制御を戻します。

- 配列のプロットは常に非圧縮で行われます。大量のポイント(> 1000000)をプロットしようとすると、数秒かかる場合があります。

- プロットの更なる属性の定義(凡例や線種など)にはEA_Graph.SetPlotPropertyを使用します。

EA_Graph.PrintOut

グラフを印刷します。

```
  EA_Graph.PrintOut(ByVal Panel As Variant, ByVal GraphName As String, Optional ByVal
Format As Integer, Optional ByVal Width As Long, Optional ByVal Height As Long) As
Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	グラフ名。
Format	0 = 横向き 1 = 縦向き
Width	TWIPS(1 inch = 1440 TWIPS)単位の出力の幅。省略した場合、グラフの元の幅を使用します。 カーソルおよび3Dグラフはサポートしていません。これらは常にグラフの元の幅を使用します。
Height	TWIPS(1 inch = 1440 TWIPS)単位の出力の高さ。省略した場合、グラフの元の高さを使用します。 カーソルおよび3Dグラフはサポートしていません。これらは常にグラフの元の高さを使用します。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

注記

- 印刷はWindowsの標準プリンタで行われます。
- 複数のグラフが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったグラフを使用します。

EA_Graph.Refresh

Post-process(概要)グラフを強制的に再描画します。

```
EA_Graph.Refresh(ByVal Panel As Variant, ByVal GraphName As String) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、*すべてのパネル*のグラフを検索することを意味します。 このメソッドは*スコープ*および*フローティングパネル*でも動作します - この場合にはGraphNameはグラフを含むレイヤのレイヤ名を示します。
GraphName	グラフ名。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

EA_Graph.ImmediateRefreshも参照してください。

注記

- 複数のグラフが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったグラフを使用します。

例

```
Sub ConfigureGraph()
  EA_Graph.ImmediateRefresh = False
  EA_Graph.SetProperty 1, "GRAPH1", 1, "Test run A-6" 'Main title
  EA_Graph.SetProperty 1, "GRAPH1", 2, "Tester: John Smith" 'Subtitle
  EA_Graph.Refresh 1, "GRAPH1"
End Sub
```

EA_Graph.RefreshAll

すべてのパネル/ページ(スコープおよびフローティングパネルも含む)上のPost-process(概要)グラフを更新します。特に演算の作成や修正後に必要です。

```
EA_Graph.RefreshAll()
```

戻り値	なし

注記

- 単一のグラフのみの更新には、EA_Graph.Refreshを使用します。

EA_Graph.RemovePlot

グラフから単一のプロットを削除します。

```
  EA_Graph.RemovePlot(ByVal Panel As Variant, ByVal GraphName As String, ByVal Plot
As Variant) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	グラフ名。
Plot	プロット名または位置。グラフに最初に割り当てられているプロットは1です。

戻り値	説明
0	エラーなし。
<0	エラー。パネル、グラフやプロットが存在しないなど。

注記

- グラフ内のすべてのプロットを削除するにはEA_Graph.ClearPlotsを使用します。
- 複数のグラフが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったグラフを使用します。

例

```
Sub ConfigureGraph()
  EA_Graph.RemovePlot 1, "GRAPH1", "Temperature"
  EA_Graph.ImmediateRefresh = False
  EA_Graph.SetProperty 1, "GRAPH1", 1, "Test run A-6" 'Main title
  EA_Graph.SetProperty 1, "GRAPH1", 2, "Tester: John Smith" 'Subtitle

  If EA.OperationMode=0 Then 'DAQ mode
    EA_Panel.SetRTDisplayChannel 1,"GRAPH1",5
  Else
    EA_Test.PlotYT 1, "Valve pressure", 1, "GRAPH1", 0, vbBlue
  END If
  EA_Graph.Refresh 1, "GRAPH1"
End Sub
```

EA_Graph.SetAxisProperty

グラフの、指定した軸のプロパティを設定します。

```
  EA_Graph.SetAxisProperty(ByVal Panel As Variant, ByVal GraphName As String, ByVal
Axis As Integer, ByVal PropIndex As Integer, ByVal PropVal As Variant) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	グラフ名。
Axis	設定する軸(標準グラフおよびカーソルグラフ): -2 = 第二(上) x軸 -1 = x軸 0 = 1番目のレイヤy軸 1 = 2番目のレイヤy軸 2 = 3番目のレイヤy軸 3 = 1番目のレイヤ右y軸 4 = 2番目のレイヤ右y軸 5 = 3番目のレイヤ右y軸
	標準グラフ: 6から14 = 4番目のから12番目のレイヤy軸 15から23 = 4番目のから12番目のレイヤ右y軸
	極座標グラフ: -1 = 角度軸 0 = 半径軸
PropIndex	設定するプロパティ: 1 = 自動スケーリング (PropVal: 0 = 自動スケーリングなし, 1 = 自動スケーリング) 2 = 最小値(自動スケーリングはオフにする必要があります) 3 = 最大値(自動スケーリングはオフにする必要があります) 4 = 対数表現(PropVal: 0 = 対数表現なし, 1 = 対数表現) 5 = 軸名 6 = 現在はサポートしてません(V4.0) 7 = すべての軸の単位を表示または非表示 8 = スケールの形式 9 = テキストの色 10 = 背景色(カーソルグラフではサポートしていません。)
PropVal	プロパティ値。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

注記

- 設定をすぐに表示するよう、EA_Graph.ImmediateRefreshを*True*に設定します。あるいは、例えば多くの設定を変更する場合、EA_Graph.Refreshでグラフを再描画します。

- 複数のグラフが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったグラフを使用します。

例

```
Sub Plot()
  EA_Test.Load "C:\MYDATA\Test1.TST"
  EA_Graph.ClearPlots 1, "PPGRAPH_1"
  EA_Test.PlotYX 1, 1, "pressure_1", "force_1", 1, "PPGRAPH_1", 0, vbRed
  EA_Graph.SetAxisProperty 1, "PPGRAPH_1", -1, AX_CAPTION, "Force" 'Axis caption.
End Sub
```

EA_Graph.SetCursorPosEx

現在のカーソル位置の座標だけでなく、接続している曲線の情報も設定します。

```
  EA_Graph.SetCursorPosEx(ByVal GraphName As String, ByVal CursorIndex As Integer,
ByVal CurveIndex As Integer, ByVal x As Double, ByVal y As Double, Optional ByVal
CurveName As String) As Long
```

引数	説明
GraphName	グラフ名。
CursorIndex	カーソルの指定(最初のカーソルは1)。バージョン3.3より導入された新しいカーソルグラフオブジェクトは、複数のカーソルをサポートしています。標準グラフの場合、この引数は1でなければなりません。
CurveIndex	グラフに複数の曲線が含まれている場合、カーソルにスナップする曲線の指標。最初の曲線は1です。
x	カーソルのx座標。
y	カーソルのy座標。
CurveName	グラフに複数の曲線が含まれている場合、カーソルにスナップする曲線の名前。

注記

- カーソルグラフの場合、この機能が動作するにはカーソルモードを"フローティング"に切り替える必要があります。これは、この関数を呼び出すと自動的に行われます。

EA_Graph.SetExportProperties

グラフをクリップボード、ファイルやプリンタにコピーするときに必要な、いくつかのプロパティを設定します。

```
  EA_Graph.SetExportProperties(ByVal EMFType As eEMFTypes, ByVal EMFOutput As
eEMFDCs, ByVal EMFSupportGradient As Integer, ByVal ImageResolutionDPI As Long, ByVal
UseLargeFonts As Integer) As Long
```

引数	説明
EMFType	0 = 従来のGDI拡張メタファイル。高く支持されています。 1 = GDI+による従来のGDIの拡張メタファイル。GDI内でGDI+をシミュレート高く支持されています。 2 = デュアル形式: 利用可能な場合はGDI+でレンダリングし、利用できない場合はGDIでレンダリングします。ファイルサイズは大きくなります。MS OfficeおよびAdobeをサポートしていますが、大きいサイズの書体は変形する傾向にあります。 3 = GDI+のレンダリングのみ。ファイルサイズは非常に小さくなります。GDI+が提供されている必要があり、サポートは限定されます。MS Office/Adobeはすべてをサポートしていない可能性があります。この設定では、テストを十分に行ってください。うまくいけば、最適なファイルサイズおよび画質にたどりつきます。
EMFOutput	0 = 現在のデオ表示デバイスコンテキストを使用します。通常のDPIは約96 DPIで、CreateDC経由で変更は行えません。この設定は、様々なDPI設定をシミュレートするためにメタファイルのヘッダを解決します。 1 = 現在のビデオ表示デバイス コンテキストを使用します。メタファイルのヘッダは変更せず、メタファイルとビデオディスプレイのDPIは一致します。転送はモニタ表示のみで、印刷を行わない場合に最適です。 2 = 現在のデフォルトのプリンタドライバを使用します。*ImageResolution*でDPIを300に設定している場合に最適です。
EMFSupportGradient	0 = カラーグラデーションやグラフのビットマップを考慮しません。 1 = カラーグラデーションやグラフのビットマップを考慮します(EMFのみ)。
ImageResolutionDPI	対象デバイスのDPI。
UseLargeFonts	1 = メタファイルおよびビットマップのエクスポート時に大きいフォントを使用します。 0 = メタファイルおよびビットマップのエクスポート時に小さいフォントを使用します。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- これらのプロパティは、プリンタ出力に影響を与える可能性があることに注意してください(EA_Graph.PrintOutを参照)。

EA_Graph.SetPlotProperty

グラフ内のプロットのプロパティを設定します。

```
  EA_Graph.SetPlotProperty(ByVal Panel As Variant, ByVal GraphName As String, ByVal
Plot As Variant, ByVal PropIndex As Integer, ByVal PropVal As Variant) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、*すべてのパネル*のグラフを検索することを意味します。
GraphName	グラフ名。
Plot	設定するプロット。グラフの最初のプロットの位置は1です。プロットは、名前または位置で指定できます。 ⚠ *EA_Graph.PlotArrayXY*または*EA_Graph.ChartArrayXY*を介して作成されたプロットは、位置インデックスを介してのみアドレス指定できます。
PropIndex	設定するプロパティ: 1 = プロット名 この名前は凡例に表示され、プロットは他のメソッドでこの名前によってアドレス指定されます。デフォルトでは、データの生成元のチャネルまたは演算の名前が使用されます。 2 = RGBの色相: 色合いはVBAで定義された定数のいずれか(vbBlack、vbWhite、vbBlue、vbRed、vbGreen、vbCyan、vbMagenta、vbYellow)、または赤、緑および青の組み合わせRGB(R,G,B)になります。 3 = プロットスタイル(PropVal: 1=線、2=線+点、3=点のみ(散布)、4=面積、5=棒、6=スプライン、7=最良適合曲線、8=2次最良適合曲線 、9=3次最良適合曲線、10=4次最良適合曲線) スタイル6～10は標準グラフにのみ適用され、カーソルグラフや極座標図には適用されません。 4 = 記号(PropVal: 0=プラス、1=十字、2=円、3=円(塗りつぶし)、4=四角、5=四角(塗りつぶし)、6=菱形、7=菱形(塗りつぶし)、8=三角、9=三角(塗りつぶし)) 5 = 線の種類(PropVal: 0=細線、1=破線、2=点線、3=破線+点線、4=破線+点線+点線、5=太線) 6 = プロットの凡例名 この名前は凡例にのみ表示されます。他のメソッドでプロットをアドレス指定のために使用する内部名は変更されません。 カーソルグラフの*EA_Graph.PlotArrayXY*によって作成されたプロットには適用されません。 7 = y(x)グラフの圧縮タイプ。1の場合、xデータは単調であると想定します。この場合、y(x)プロットには通常の圧縮が使用されます。 *EA_Graph.PlotArrayXY*によって作成されたプロットには適用されません。 8 = xオフセット *EA_Graph.PlotArrayXY*によって作成されたプロットには適用されません。 9 = yオフセット *EA_Graph.PlotArrayXY*によって作成されたプロットには適用されません。 10 = 1の場合、右のy軸に対するプロット カーソルグラフの*EA_Graph.PlotArrayXY*によって作成されたプロットには適用されません。
PropVal	プロパティ値。

戻り値	説明
>0	プロットのインデックス。
0	エラー。指定されたプロットが見つかりません。
<0	エラー。パネルやグラフが存在しないなど。

注記

- グラフ内のすべてのプロットを削除するにはEA_Graph.ClearPlotsを使用します。
- 複数のグラフが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったグラフを使用します。
- プロットスタイルに"最良適合曲線"を使用している場合、関数 EA_Graph.GetFitParametersを介してフィット係数を取得できます。

例

```
  Sub Plot()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Graph.ClearPlots 1, "PPGRAPH_1"
    EA_Test.PlotYX 1, 1, "pressure_1", "force_1", 1, "PPGRAPH_1", 0, vbRed
    EA_Graph.ImmediateRefresh=False
    EA_Graph.SetPlotProperty 1, "PPGRAPH_1", 1, 1, "Main pressure" 'Rename plot.
    EA_Graph.SetPlotProperty 1, "PPGRAPH_1", "Main pressure", 2, RGB(120,200,250)
'Change plot Color.
    EA_Graph.SetPlotProperty 1, "PPGRAPH_1", "Main pressure", 3, 2 'Plot type=Area.
    EA_Graph.Refresh 1, "PPGRAPH_1"
  End Sub
```

EA_Graph.SetProperty

タイトルや色など、グラフの一般的な設定を行います。

```
  EA_Graph.SetProperty(ByVal Panel As Variant, ByVal GraphName As String, ByVal
PropIndex As Integer, ByVal PropVal As Variant) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネル位置は1です。Panel = 0の場合、*すべてのパネル*のグラフを検索することを意味します。
GraphName	グラフ名。
PropIndex	設定するプロパティ: 1 = メインタイトル 2 = サブタイトル上 3 = サブタイトル下 4 = 秒単位の時間枠 リアルタイムグラフに使用できる値 : 0.5, 1, 2, 5, 10, 30, 60, 120, 300, 600, 1800, 3600, 7200, 18000, 36000, 54000, 86400, 172800, 259200, 345600, 432000, 518400, 604800 カーソル グラフに使用できる値: 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 2, 5, 10, 30, 60, 120, 300, 600, 1800, 3600, 7200, 18000, 36000, 54000, 86400, 172800, 259200, 345600, 432000, 518400, 604800 5 = 圧縮を開始するまでのデータポイント数。 ⚠ このサイズのメモリは、グラフがサポートするすべてのプロット(曲線)に割り当てられることに注意してください(PropIndex 9を参照)。グラフがサポートする30プロットまでのメモリ消費量(デフォルト): 30 * (2 * データポイント数) * 8 係数2は、xおよびy軸両方のデータポイントを意味します。 すなわち、プロットしたい曲線数に関係なく、10000のデータ数では4.8MBのメモリを消費します。 非圧縮で大量のデータをプロットしたい場合、プロットの最大数を減らすことを推奨します(例えば圧縮をサポートしていないPlotArrayXYおよびPlotArrayY)。 6 = 等高線ステップ数(等高線グラフのみ) 7 = 等高線カラーテーブルのインデックス(0..7) 8 = MS Word にエクスポートするブックマーク(文字列) 9 = グラフでサポートされるプロット(曲線)の最大数 10 = プロットエリアの背景色 11 = グラフ枠の背景色 12 = グリッド線と目盛りの色
PropVal	プロパティ値。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやグラフが存在しないなど。

注記

- 設定をすぐに表示するよう、EA_Graph.ImmediateRefreshを*True*に設定します。あるいは、例えば多くの設定を変更する場合、EA_Graph.Refreshでグラフを再描画します。

- 複数のグラフが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったグラフを使用します。

例

```
  Sub CustomizeGraph()
    EA_Graph.ImmediateRefresh=False
    EA_Graph.SetProperty(1, "RTGRAPH_1", GRAPH_MAINTITLE, "Force_Displacement_
Measure")
    EA_Graph.SetProperty(1, "RTGRAPH_1", GRAPH_SUBTITLE, "HBM Script Demo")
    EA_Graph.SetProperty(1, "RTGRAPH_1", GRAPH_BOTTOMTITLE, Now)
    EA_Graph.SetAxisProperty(1, "RTGRAPH_1", 0, AX_CAPTION, "Force")
    EA_Graph.SetAxisProperty(1, "RTGRAPH_1", 0, AX_UNIT, "N")
    EA_Graph.SetAxisProperty(1, "RTGRAPH_1", -1, AX_CAPTION, "Displacement")
    EA_Graph.SetAxisProperty(1, "RTGRAPH_1", -1, AX_UNIT, "m")
    EA_Graph.Refresh(1, "RTGRAPH_1")
  End Sub
```

EA_Graph.SyncAllCursors

グラフ、ビデオプレーヤー、表示器のカーソルを、T0で指定した特定の時間に同期させます。

```
EA_Graph.SyncAllCursors(ByVal T0_Seconds As Double) As Long
```

引数	説明
T0_Seconds	グラフではカーソルがこの位置に移動し、ビデオではその時点のフレームが表示され、表示器ではその時点に属する数値が表示されます。T0は秒単位で指定する必要があります。

- この関数は解析モードでのみ使用できます。

EA_IO

EA_IOを使用すると、*計測チャネル*や*デバイス*にアクセスすることができます。このオブジェクトを使用すると、対話型モードでは使用できない、いくつかのアクションを実行することができます。

このオブジェクトによって公開される最も重要なメソッドのいくつかは次のとおりです:

- デバイスに直接コマンド(ローレベル)を送受信: EA_IO.Command、EA_IO.ChanSelect、EA_IO.GetSlotInfo
- 接続されたデバイスをスキャン: EA_IO.Scan
- 現在の DAQ プロジェクト内のデバイスまたはチャネルの位置を取得: EA_IO.DevicePos、EA_IO.ChanPos
- すべてのDAQ計測および演算チャネルの最大数を取得: EA_IO.MaxTempStoreChannels
- 現在の DAQ プロジェクト内のすべてのデバイスまたはチャネルの数を取得: EA_IO.DeviceCount、EA_IO.ChanCount
- DAQプロジェクト内の位置からデバイス名を取得: EA_IO.GetDeviceNameFromPos
- QuantumXモジュール情報とXML構成設定を取得: EA_IO.QuantumX.GetModuleInfo、EA_IO.QuantumX.GetXMLView
- DAQプロジェクトのチャネルの情報を取得: EA_IO.ChannelAttribute
- チャネル名の変更: EA_IO.SetName
- チャンネルを目的のサンプルレートグループに割り当て: EA_IO.AssignToSamplerateGroup
- センサの割り当ての有無: EA_IO.AdaptSensor
- 既存のオンライン演算チャネルの式を取得/設定: EA_IO.GetFormula、EA_IO.SetFormula
- チャネルリストの更新: EA_IO.RefreshChanView
- チャネルステータスのテキストおよびシンボルを設定: EA_IO.SetStatusMsg
- チャネルリストをテーブルとしてファイルに保存: EA_IO.SaveChanTable
- TEDS変換器をスキャン: EA_IO.Sensors.ScanTEDS
- センサデータベースからセンサをチャネルに割り当て: EA_IO.Sensors.SelectSensor
- 名前からセンサのIDを取得: EA_IO.Sensors.GetSensorIDFromName
- チャネルに割り当てられているセンサのIDおよび名前を取得: EA_IO.Sensors.GetSensorIDFromChannel
- 現在のセンサデータベースからチャネルのセンサを更新: EA_IO.Sensors.UpdateSensorFromDatabase
- センサデータベース内のセンサを操作: EA_IO.Sensors.CreateSensor、EA_IO.Sensors.WriteSensor、EA_IO.Sensors.ReadSensor、EA_IO.Sensors.ReadLastModified
- ゼロバランスの設定: EA_IO.ZeroBalanceControl、EA_IO.ZeroBalanceMethod、EA_IO.ZeroBalanceAvgTime、EA_IO.ZeroBalanceSampleRate
- チャネルの初期化、初期化の除外: EA_IO.InitializeChannel、EA_IO.NoAutoInit
- DAQジョブを使用せずに計測値を読み取り: EA_IO.Measure、EA_IO.Snapshot
- LED(QuantumX、PMX、CANHEAD/ML74)を点滅させてチャネルまたはモジュールを物理的に識別: EA_IO.FlashLED
- サンプルレートドメインの設定: EA_IO.QuantumX.SetSamplerateDomain
- "代数"タイプの演算チャネルの作成や修正: EA_IO.CreateCalculatorChannel、EA_IO.ModifyCalculatorChannel
- 補助チャネルの作成: EA_IO.CreateAuxChannel
- "クラス集計"や"ひずみゲージロゼット解析"の演算チャネルを作成: EA_IO.CreateClasscountChannel、EA_IO.CreateRosetteChannel
- MGCplus ML71、QuantumX MX840、またはMX471でCANメッセージの送信: EA_IO.SendCANMessage
- QuantumX MX471のECUのCCP/XCPトラフィックのオン/オフの切り替え: EA_IO.QuantumX.ECUControl
- *EA_IO.QuantumX*クラスによるQuantumXの拡張機能: XMLの読み取り/書き込み、ピーク値の制御、シャントの適用、モジュールの再接続、単一データの読み取り
- QuantumXモジュールのデジタル入出力: EA_IO.QuantumX.DigitalIOControl
- QuantumXモジュールのテスト信号の設定: EA_IO.QuantumX.SetTestSignalMode、EA_IO.QuantumX.SetTestSignalValue
- QuantumX MX840/MX840Aの一番目のコネクタをCANまたはアナログモードに設定: EA_IO.QuantumX.SetToCAN、EA_IO.QuantumX.SetToAnalog
- QuantumX MX410/MX411/MX403のコネクタの高速モードのオン/オフを切り替え: EA_IO.QuantumX.SetHighspeedMode
- QuantumXの内部制御関数を実行: EA_IO.QuantumX.ExecuteControl
- デジタル入出力の読み取り/書き込み: EA_IO.SetDigitalOut、EA_IO.ReadDigitalIn
- アナログ出力の設定: EA_IO.SetAnalogOut

- catmanがCX22で実行しているかどうかを確認: EA_IO.IsCX22
- CX22のUPS状態を取得: EA_IO.CX22GetUPSState
- CX22モジュールのデジタル入出力の読み取り/書き込み: EA_IO.CX22SetOutputBit、EA_IO.CX22ReadInputBit
- CX27モジュールのデジタル入出力の読み取り/書き込み: EA_IO.CX27DIOControl
- QuantumX MX878/879のアナログおよびデジタル出力を有効/無効に設定: EA_IO.EnableControllers
- QuantumX MX878/M879のアナログ出力の任意の信号発生器を設定 EA_IO.QuantumX.SetArbitrarySignalGenerator

以下のアクションは、現在のバージョンでサポートされていません:

- ハードウェアを接続せずにチャネルを定義
- ML71のCANデータベース読み込み
- ML71サブチャネルの構成

このオブジェクトでは以下のメソッドが公開されています:

EA_IO.AdaptSensor ・・・

EA_IO.QuantumX.GetModuleInfo ・・・

EA_IO.AdaptSensor

これは、スクリプトからハードウェアチャネルを完全に構成できる非常に強力な機能です。
*EA_IO.AdaptSensor*で行った変更を適用するには、常にEA_IO.InitializeChannelを呼び出す必要があります。

この関数は、次の2つの異なるシナリオで使用します:

- センサデータベースのセンサがすでに割り当てられているチャネル上
  センサが割り当てられた後にセンサのプロパティを変更します。これは"Sensor adaptation"ダイアログに非常に似ていますが、さらに多くの設定を構成できます。
- センサが割り当てられていないチャンネルの場合
  これは、ハードウェアの直接構成に対応します。センサデータベースを関与させる必要はありません。センサのタイプを変更することもできますが、対話型ダイアログでは変更できません。このメソッドを使用するには、チャネルを初期化する前に、関連するプロパティをすべて設定する(つまり、AdaptSensorを繰り返し呼び出す)必要があります。
  *EA_IO.InitializeChannel*は、*Mode*パラメータを`INIT_ADAPTSENSOR(2)`に設定して呼び出す必要があります。

```
  EA_IO.AdaptSensor(ByVal Channel As Variant, ByVal PropType As Integer, PropVal As
Variant) As Long
```

引数	説明
`Channel`	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
`PropType`	設定するセンサのプロパティ: `1` = 一般的なセンサのタイプ `2` = 特定のセンサのタイプ `3` = 電気ユニットの最初の特徴点 `4` = 電気ユニットの2番目の特徴点 `5` = 物理ユニットの最初の特徴点 `6` = 物理ユニットの2番目の特徴点 `7` = 最大物理範囲 `8` = ゲージ係数(kファクタ) `9` = 励起電圧(ボルト) `10` = 供給電圧のキャリア周波数(Hz)、`0`=DC、`-1`=自動 `11` = 工学単位(物理単位) `12` = ブリッジ係数
`PropVal`	プロパティ値。

戻り値	説明
`0`	エラーなし。
`<0`	エラー。チャネルが存在しないなど。

注記

次のセクションでは、チャネルを最初から完全に構成するワークフローについて説明します:

1) 汎用センサのタイプを設定

センサのタイプ	値
ブリッジ変換器	2
ゲージ係数付S/G ブリッジ	1
直流電圧	4
直流電流	5
熱電対	8
ptxxxx	7
周波数	6
カウンタ	12
誘導ブリッジ変換器	11

センサのタイプ	値
LVDT	3
ピエゾ(アクティブ)	9
ピエゾ(パッシブ)	13
抵抗器	10
ポテンショメータ	14
光ファイバ式ひずみセンサ	20
光ファイバ式温度センサ	21
SSI	30
IEPE	31
CANバス	100

2) 必要に応じて、特定の変換器のタイプを設定

SSIなどの一部のセンサの場合は、汎用タイプで十分です。さまざまなタイプについては、次のコードを参照してください。

コード	センサ
350	フルブリッジひずみゲージセンサ
351	ハーフブリッジひずみゲージセンサ
353	フルブリッジひずみゲージ変換器
354	ハーフブリッジひずみゲージ変換器
356	誘導フルブリッジ
357	誘導ハーフブリッジ
362	フルブリッジひずみゲージ 120オーム(kファクタ付)
363	フルブリッジひずみゲージ 350オーム(kファクタ付)
364	フルブリッジひずみゲージ 700オーム(kファクタ付)
365	ハーフブリッジひずみゲージ 120オーム(kファクタ付)
366	ハーフブリッジひずみゲージ 350オーム(kファクタ付)
367	ハーフブリッジひずみゲージ 700オーム(kファクタ付)
368	1/4ブリッジひずみゲージ 4線120オーム(kファクタ付)
369	1/4ブリッジひずみゲージ 4線350オーム(kファクタ付)
370	1/4ブリッジひずみゲージ 4線700オーム(kファクタ付)
371	1/4ブリッジひずみゲージ 3線120オーム(kファクタ付)
372	1/4ブリッジひずみゲージ 3線350オーム(kファクタ付)
373	1/4ブリッジひずみゲージ 3線700オーム(kファクタ付)
376	1/4ブリッジひずみゲージ 4線1000オーム(kファクタ付)
380	LVDT
385	ポテンショメータ
420	DC電圧
421	DC電流
450	熱電対J
451	熱電対K
452	熱電対T
453	熱電対S
454	熱電対B
455	熱電対E
456	熱電対R
457	熱電対N
458	熱電対C
475	抵抗
500	Pt10
501	Pt100
502	Pt500
503	Pt1000
520	周波数
522	周波数 + F2信号
523	周波数 4x (4倍)
525	Counter
581	IEPE

3) 入力特性(スケーリング)、励起電圧などのその他のパラメータを設定。

4) *EA_IO.InitializeChannel*のMode = 2(INIT_ADAPTSENSOR)を使用してチャネルを初期化

例

例1: 割り当て済みセンサのゲージ率を変更

```
EA_IO.AdaptSensor "Pressure", SEPROP_GAGEFAC, 2.0756
EA_IO.InitializeChannel "Pressure", INIT_FULL
```

例2: センサを割り当てずにチャネルをフルブリッジとして設定する方法

```
EA_IO.AdaptSensor "Pressure", SEPROP_GENERICTYPE, 2
EA_IO.AdaptSensor "Pressure", SEPROP_TRANSDUCERTYPE, 353
EA_IO.AdaptSensor "Pressure", SEPROP_ENGUNIT, "bar"
EA_IO.AdaptSensor "Pressure", SEPROP_P1X, 0.0034 'mV/V
EA_IO.AdaptSensor "Pressure", SEPROP_P2X, 2.1443 'mV/V
EA_IO.AdaptSensor "Pressure", SEPROP_P1Y, 0 'bar
EA_IO.AdaptSensor "Pressure", SEPROP_P2Y, 500 'bar
EA_IO.AdaptSensor "Pressure", SEPROP_MAXRANGE, 1000 'bar
EA_IO.AdaptSensor "Pressure", SEPROP_UEX , 2.5 'V
EA_IO.AdaptSensor "Pressure", SEPROP_CARRIERFREQ, 4800 'Hz

EA_IO.InitializeChannel "Pressure", INIT_ADAPTSENSOR
```

EA_IO.AssignToSamplerateGroup

チャンネルをデフォルト、低速または高速のサンプルレートグループに割り当てます。この設定は、DAQプロジェクト内のすべてのジョブに適用されます。

```
  EA_IO.AssignToSamplerateGroup(ByVal Channel As Varaiant, ByVal SRGroup As Integer)
As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
SRGroup	0 = デフォルトのサンプルレートグループ 1 = 低速サンプルレートグループ 2 = 高速サンプルレートグループ

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- MGCplusマルチチャネル増幅器または光インテロゲータのサブチャネルは、同一のサンプルレートグループに属している必要があります。サブチャネルを適切なサンプルレートグループに割り当てると、同じアンプの他のすべてのサブチャネルも同じサンプルレートグループに割り当てられます。

例

```
Dim i As Integer
Dim chanCount As Integer
Dim chanType As Integer
Dim chanAttr As Integer
chanCount = EA_IO.ChanCount
chanType = 0 'Channel type
For i = 1 To chanCount
  EA_IO.ChannelAttribute i, chanType, chanAttr
  If chanAttr = 1 Then 'Measurement channel
    EA_IO.AssignToSampleRateGroup i, 1 'Assign channel to slow sample rate group
  End If
Next
```

EA_IO.ChanCount (Read only)

現在のDAQプロジェクト内のすべてのチャネル数を返します。これには計測および、演算の両方を含みます。

```
Property EA_IO.ChanCount
```

注記

- 計測なのか演算チャネルなのかを確認するにはEA_IO.ChannelAttributeを使用します。

例

```
Sub CheckChannels()
  Dim i As Integer, CType As Integer
  Dim MV As Double, Status As Long
  Const DAQCHAN = 1
  Const MATHCHAN = 2
  For i = 1 To EA_IO.ChanCount
    EA_IO.ChannelAttribute i, 0, CType
    If CType = DAQCHAN Then
      EA_IO.Measure i, MV, Status
      If Status <> 0 Then
            ...
      End If
    End If
  Next
End Sub
```

EA_IO.ChannelAttribute

DAQプロジェクトのチャネルから、様々な情報を取得します。

対話型モードでは、拡張チャネル情報に相当します。

```
  EA_IO.ChannelAttribute(ByVal Channel As Variant, ByVal AttribType As Integer,
Attrib As Variant) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
AttribType	属性の種類: 0 = Channel type Attrib = 1: 計測チャネル Attrib = 2: 演算チャネル Attrib = 3: catman時間チャネル Attrib = 4: ハードウェア時間チャネル Attrib = 5: CAN Rawチャネル 1 = 名前 2 = 単位 3 = コメント 4 = サンプルレート 5 = スロット 6 = サブチャンネル 7 = センサ名 8 = センサID 9 = TEDS: モデル、メーカ、チップIDを含む文字列 10 = アンプのタイプ 11 = 接続プレートのタイプ 12 = 変換器回路 13 = フィルタ特性 14 = フィルタ周波数 15 = サンプルレートからのdt(msec単位) 16 = チャネルがオンライン演算の場合の数学式 17 = センサ校正日(OLE時間形式) 18 = センサ校正の有効期限(OLE時間形式) 19 = チャネルが属するデバイスのインデックス 20 = 親デバイスのタイプ Attrib = 8: Spider8 Attrib = 157: NMEA 互換デバイス(GPS、Vaisala 気象観測所) Attrib = 2000: MGCplus Attrib = 2400: PMX Attrib = 2500: Kistler RoaDyn/KiRoad Attrib = 2600: DMP41 Attrib = 3000: QuantumX Attrib = 4000: 光インテロゲータ 21 = センサのシリアル番号 32 = 拡張エラーステータス(テキスト)

AttribType	33 = チャンネルのサンプルレート Attrib = 0: デフォルトのサンプルレート Attrib = 1: 低速サンプルレート Attrib = 2: 高速サンプルレート 34 = ゼロバランスのステータス (0 = ロックされていない、それ以外の場合はロックされている) 35 = 補助チャンネル Attrib = 0: 補助チャネルなし Attrib = 1: 補助チャネルのデフォルトのサンプルレートグループ Attrib = 2: 補助チャネルの低速サンプルレートグループ Attrib = 3: 補助チャネルの高速サンプル レートグループ Attrib = 4: 手動タイムベースを備えた補助チャネル Attrib = 5: 基準チャネルからのタイムベースを持つ補助チャネル 36 = QXモジュールのファームウェアバージョン。他のデバイスタイプではサポートしていません
Attrib	属性値。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- これは、現在のDAQプロジェクトの計測チャネルの情報を読み出すメソッドのため、DAQモードでのみ有効です。テストに保存したチャネルからのチャネル情報を検索するには、EA_Test.ChannelAttributeを使用します。

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
Sub GetChannelInfo()
  Const C_NAME = 1
  Const C_AMPTYPE = 10
  Const C_TRANSDUCER = 12
  Dim Attrib
  Dim Name As String, AmpType As String, TransducerType As String
  EA_IO.ChannelAttribute "Valve pressure", C_NAME, Attrib
  Name = Attrib
  EA_IO.ChannelAttribute "Valve pressure", C_AMPTYPE, Attrib
  AmpType = Attrib
  EA_IO.ChannelAttribute "Valve pressure", C_TRANSDUCER, Attrib
  TransducerType = Attrib
End Sub
```

EA_IO.ChanPos

名前に基づいて、DAQプロジェクトのリスト(**Channel settings**ウィンドウ)からチャネルの位置を返します。リスト内の計測チャネルは1(通常は時間チャネル)から始まります。

```
EA_IO.ChanPos(ByVal ChannelName As String) As Long
```

引数	説明
ChannelName	チャネル名。

戻り値	説明
>0	チャネル位置。
<0	エラー。存在しないチャネルなど。

注記

- チャネル位置は、デバイスのスキャン後か、DAQプロジェクト読み込み後に確立されます。
- パラメータとしてのチャネルやチャネル名として、すべての関数に転送することができます。DAQプロジェクトに毎に異なるデバイスの配置でも、実際のチャネル位置に依存しないため、プログラムが読みやすくなります。欠点は、名前にチャネルの詳細を使用すると、実行に時間がかかることです。

EA_IO.ChanSelect

以降のコマンドのためにチャネルを選択します。例えばEA_IO.Command。

```
EA_IO.ChanSelect(ByVal Channel As Variant) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。

戻り値	説明
0	エラーなし。
<0	エラー。存在しないチャネルなど。

注記

- MGCplusを使用では、EA_IO.Commandと、PCS(スロット)およびSPS(サブチャネル)コマンドでチャネルを選択することができます。このように、チャネルは同時に選択することができます。チャネルのスロットおよび、サブチャネル数は、EA_IO.GetSlotInfoより、読み出すことができます。

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
Sub SetZero()
  'Zero balancing a channel
  Dim Answer
  EA_IO.ChanSelect "Valve pressure"
  Answer = EA_IO.Command(1, "CDV")
End Sub
```

EA_IO.Command

PMX、MGCplus、Spider8または光インテロゲータデバイスに直接コマンドを送信し、デバイスの応答を返します。このメソッドで、デバイスの内部関数にアクセスすることができます。

⚠ このメソッドは、すべてのデバイスに適用されるわけではありません。
QuantumXでは代わりに、EA_IO.QuantumX.WriteXMLおよびEA_IO.QuantumX.ReadXMLメソッドを使用します。

```
EA_IO.Command(ByVal Device As Variant, Cmd As String) As Variant
```

引数	説明
Device	コマンドを送信すべきデバイスのインデックス。DAQプロジェクトのデバイスは、1から始まります。デバイスの名前、IPドレス、UIUIDからインデックスを取得するには、EA_IO.DevicePosメソッドを使用します。
Cmd	コマンド(二重引用符で囲んだ文字列)。

戻り値
文字列としてのデバイスの応答。アクションを実行するコマンドは0(成功)または?(エラー)を返します。クエリコマンド(すなわちコマンドの後に?)はクエリに対する応答を返します。コマンドが送信されないか、応答がなかった場合、戻り値は-32766になります。詳細は、EA.LastErrorおよびEA.LastErrorTextから取得することができます。

ℹ チャネルの初期化およびトレーサビリティデータも参照してください。

注記

- このメソッドを使用するにはHBMのコマンドセットの純分な知識が必要です。コマンドの不適切に使用すると、catman/APの動作に支障をきたす可能性があります。コマンドのリファレンスは、System-CDやHBMのWebサイトから、検索してください。
- DAQジョブの実行中は、デバイスでのデータ収集に影響を与えるコマンドを送信しないでください(例えばPMXやMGCplusでTSV、STP、RMB?。Spider8でMSV、OMB?)。光インテロゲータは、DAQジョブ実行中にコマンドを送信した場合、何の応答も受信しません。
- 終端文字(";"やLFのような)を指定しないでください。これはメソッドにより自動的に行われます。
- PMXおよびMGCplusの多くのコマンドは、現在選択されているチャネルに従います。EA_IO.ChanSelectでチャネルを選択してください! 別の方法として、チャネルに対応する"PCS<Slot>"および"SPS<Subchannel>"コマンドを使用することができます。このメソッドは、一度に複数のチャネルを選択することができます。EA_IO.GetSlotInfoからDAQプロジェクト内のチャネルの、スロットおよびサブチャネル番号を取得することができます。
- Spider 8では、チャネルをコマンドで指定します。例えば、"ASA?0"は、最初のチャネルの変換器の設定を問い合わせます。EA_IO.GetSlotInfoからDAQプロジェクト内のチャネルのスロットを取得することができます。
- catmanのチャネル初期化プロセスの一部を変更する場合(例: 変換器、励起電圧、フィルタ周波数)、catmanがチャネルの初期化プロセスを自動実行して設定を上書きしないようにする必要があります。このために、EA_IO.NoAutoInit = Trueを使用します。
- DAQジョブの自動実行開始前に特別な設定/構成コマンドを適用させたい場合、"自動実行ステージ"**After channel init**を使用します。
- 設定/構成コマンドを適用した場合、EA_IO.InitializeChannelからチャネルのトレーサビリティデータを更新することを推奨しています。

例

```
Sub SetZeroMGC()
  'Zerobalance a channel (MGCplus)
  Dim Answer
  EA_IO.ChanSelect "Pressure"
  Answer = EA_IO.Command(1, "CDV")
End Sub

Sub SetZeroSpider()
  'Zerobalance a channel (Spider 8)
  Dim Answer
  Dim Slot As Integer, SubChan As Integer
  EA_IO.GetSlotInfo "Pressure", Slot, SubChan
  Answer = EA_IO.Command(1, "TAV" & Slot)
End Sub
```

EA_IO.CPDIOCONTROL

MGCplus CP42/52デジタル入力/出力の1つを設定または読み取ります。

```
  EA_IO.CPDIOControl(ByVal Device As Variant, ByVal Action As eDIOAction, ByVal Pin
As Integer, State As Integer) As Long
```

引数	説明
Device	コマンドの送信先となるデバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。
Action	0 = 入力の読み取り 1 = 出力の設定
Pin	0 = デジタル入出力1 1 = デジタル入出力2
State	Action=0の場合、入力の状態を返します。それ以外の場合は、出力状態に設定します。0=LOW、1=HIGH。

戻り値	説明
0	エラーなし。
<0	エラー。デバイスが存在しないなど。

- 単一チャネル(MGCplus)のデジタルI/Oにアクセスするには、EA_IO.ReadDigitalInおよびEA_IO.SetDigitalOutを使用します。

EA_IO.CreateAuxChannel

プログラマがデータを入力できる、新規の補助チャネルを作成します。新規チャネルは、演算チャネルリストの後に追加されます。EA_DAQ.SetCurrentBlockメソッドを使用して補助チャネルを設定します。

```
  EA_IO.CreateAuxChannel(Optional ByVal Name As String, Optional ByVal Unit As
String, Optional ByVal TMode As Integer, Optional ByVal dt As Single) As Long
```

引数	説明
Name	新規のチャネル名。
Unit	チャネルの単位。
TMode	タイムスタンプモード: 0=デフォルトサンプルレート 1=低速サンプルレート 2=高速サンプルレート 3=dt指定
dt	dt(ms)。TMode=3の場合。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- DAQジョブの実行中は、このメソッドは呼び出せません。
- 1ms未満のdt(TMode=3)の指定はサポートされていません。

EA_IO.CreateCalculatorChannel

calculator型(代数計算)の演算チャネルを作成します。新規チャネルは、演算チャネルリストの後に追加されます。
式に現れるすべての引数チャネルの代数計算を*行ごとに*行います。

```
  EA_IO.CreateCalculatorChannel(ByVal ChanName As String, ByVal Expression As String,
ByVal Unit As String, Optional ByVal RefreshChannelView As Integer) As Long
```

引数	説明
ChanName	新規のチャネル名。
Expression	式。チャネル名を式内に使用することができます: `1/3*(pressure_1 + pressure_2)`
Unit	チャネルの単位。
RefreshChannelView	チャネルリストの再描画。新規の演算を追加後、自動的にチャネルリストを再描画させる場合はtrue(1)。一度に多くのチャネルの追加を予定している場合、この引数をfalse(0)に設定し、追加後にEA_IO.RefreshChanViewでチャネルリストの表示を更新することを推奨しています。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- DAQジョブの実行中は、このメソッドを呼び出すことはできません。DAQジョブ実行中に演算を変更するにはEA_IO.Modify CalculatorChannelを参照してください。

- 次の演算子および関数式を数式に使用することができます:

  `+, -, /, *, ^`
  `AND, OR, >=, <=, =`
  `sqrt, log, ln, exp`
  `sin, cos, tan, arctan`
  `ceil, floor, min, max`

  特殊関数は微分および積分です:

  ```
  deriv(y channel, x channel)
  integrate(y channel, x channel)
  ```

- 演算に*ResultName*が含まれているとエラーが発生します。

- 式では、有理数(小数点以下の桁がある数値)をカンマではなく、常に小数点を使用して入力する必要があります。

EA_IO.CreateClasscountChannel

クラスカウント型の演算チャネルを作成します。新規チャネルは、演算チャネルリストの後に追加されます。

```
  EA_IO.CreateClasscountChannel(ByVal ResultName As String, ByVal InputChannel
As Variant, ByVal CountType As Integer, ByVal nClasses As Integer, ByVal ClassMin
As Double, ByVal ClassMax As Double, ByVal MinAmplitudePercent As Double, ByVal
CloseResid As Integer, Optional ByVal RefChannel As Variant, Optional ByVal ClassMin2
As Double, Optional ByVal ClassMax2 As Double, Optional ByVal RefreshChannelView As
Integer) As Long
```

引数	説明
ResultName	結果チャネル名。
InputChannel	分類するチャネル名。
CountType	クラスカウントの方法: 0 = レインフロー(FromTo) 1 = スパンペア 2 = レベルにおける時間 3 = レベルにおける共同時間 4 = レインフロー(RangeMean)
nClasses	クラス数(4...255)
ClassMin	クラスの下限値
ClassMax	クラスの上限値
MinAmplitudePercent	peak-valley検出のために考慮される最小振幅スパン(ゲート)(全クラス範囲の%)。
CloseResid	1に設定した場合、不完全な残差も集計します。 CountTypeが*レインフロー(FromTo)*、*スパンペア*および*レインフロー(RangeMean)*でのみ必要です。
RefChannel	任意です。二番目の信号チャネルまたは演算(例えばRPM)。"reference"信号として表します。 CountTypeが*レベルにおける共同時間*でのみ必要です。
ClassMin2	任意です。CountTypeが*レベルにおける共同時間*でのみ必要です。
ClassMax2	任意です。CountTypeが*レベルにおける共同時間*でのみ必要です。
RefreshChannelView	任意です。演算チャネルを作成後、直ちにチャネルリストを更新します。デフォルト = 0。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- DAQ ジョブの実行中は、このメソッドを呼び出すことはできません。
- 演算に*ResultName*が含まれているとエラーが発生します。

例

```
  Sub Rainflow_FT_Online()
    EA_IO.CreateClasscountChannel "Pressure Rainflow_FT", "pressure_1", CLSC_
RAINFLOWFROMTO, 64, -200, 200, 1, 1
  End Sub

  Sub SpanPairs_Online()
    EA_IO.CreateClasscountChannel "Pressure SpanPairs", "pressure_1", CLSC_SPANPAIRS,
64, -200, 200, 1, 1
  End Sub

  Sub JointTimeAtLevel_Online()
  EA_IO.CreateClasscountChannel "PresDisp JointTimeAtLevel", "pressure_1", CLSC_
JOINTTIMEATLEVEL, 64, -200, 200, 1, 0, "displacement_1", -5, 5
  End Sub
```

EA_IO.CreateRosetteChannel

ロゼットタイプのひずみゲージ解析の演算チャネルを作成します。新規チャネルは、演算チャネルリストの後に追加されます。

```
  EA_IO.CreateRosetteChannel(ByVal ResultName As String, ByVal Ca As Variant,
ByVal Cb As Variant, ByVal Cc As Variant, ByVal RosType As Integer, ByVal EMod As
Double, ByVal PoisRatio As Double, ByVal ResultType As Integer, Optional ByVal
RefreshChannelView As Integer) As Long
```

引数	説明
ResultName	結果チャネル名。
Ca	計測グリッドaのひずみチャネルの名前。
Cb	計測グリッドbのひずみチャネルの名前。
Cc	計測グリッドcのひずみチャネルの名前。
RosType	ロゼットひずみゲージの種類: 45 = 45°/90°ロゼット 60 = 60°/120°ロゼット 90 = 90° 2軸ロゼット 1 = 単一ひずみゲージ
EMod	弾性係数。
PoisRatio	ポアソン比
ResultType	解析結果の種類: 0 = 角度 1 = 主公称応力 1 2 = 主公称応力 2 3 = せん断応力 4 = 基準応力 (ミーゼス応力) 5 = 応力X 6 = 応力Y 7 = 主ひずみ1 8 = 主ひずみ2 9 = ひずみX 10 = ひずみY 11 = せん断ひずみ
RefreshChannelView	任意です。演算チャネルを作成後、直ちにチャネルリストを更新します。デフォルト = 0。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- DAQジョブの実行中は、このメソッドが呼び出されない可能性があります。
- 計算に*ResultName*が含まれているとエラーが発生します。
- RosTypeが*2 軸ロゼット*の演算では、計測グリッドcのひずみチャネルは無視されます。*単一ひずみゲージ*の演算では、グリッドbおよびcのチャネルは無視されます。ただし、演算が機能するには、これらの場所で有効なチャネルを指定する必要があります。
- *単一のひずみゲージ*の演算では、結果として*主公称応力1*のみを提供します。引数*ResultType*は無視されます。

EA_IO.CX22GetUPSState

CX22のUPS状態を取得します。

```
EA_IO.CX22GetUPSState() As Long
```

戻り値	説明
-1	利用可能なUPSがありません。
5	利用可能な電源がなく、危機的な状態です。
13	利用可能な電源はありませんが、状態は良好です。
7	利用可能な電源があり、充電中です。
15	利用可能な電源があり、フル充電済みです。

例

```
Sub MonitorCX22PowerLoss()
  Dim UPSState As Double
  UPSState = EA_IO.CX22GetUPSState

  If UPSState = 5 Or UPSState = 13 Then
    EA_DAQ.TriggerEvent "Send notification", 1
  End If
End Sub
```

EA_IO.CX22ReadInputBit

CX22の3点デジタル入力の、指定したピンを読み取ります。

```
EA_IO.CX22ReadInputBit(ByVal BitPos As Integer, State As Integer) As Long
```

<table>
<tr><th>引数</th><th>説明</th></tr>
<tr><td>BitPos</td><td>指定した入力ピン番号。
⚠ CX22の前面に印刷されたピンやLEDのラベルは、ビット位置を示すものではありません。ビット位置とピンラベルの関係は以下の表を参照してください:
<table>
<tr><th>ピン番号</th><th>BitPos</th></tr>
<tr><td>1</td><td>5</td></tr>
<tr><td>2</td><td>4</td></tr>
<tr><td>3 and Start/Stop switch</td><td>6</td></tr>
</table>
</td></tr>
<tr><td>State</td><td>1 = HIGH
0 = LOW</td></tr>
</table>

EA_IO.CX22SetOutputBit

CX22のデジタル出力の1つを設定します。各出力には、それに応じて設定されるLEDも搭載されています。

```
EA_IO.CX22SetOutputBit(ByVal BitPos As Integer, ByVal State As Integer) As Long
```

<table>
<tr><th>引数</th><th>説明</th></tr>
<tr><td>BitPos</td><td>指定する出力ピン番号。
⚠ CX22の前面に印刷されたピンやLEDのラベルは、ビット位置を示すものではありません。ビット位置とピンラベルの関係は以下の表を参照してください:
<table>
<tr><th>ピン番号</th><th>BitPos</th></tr>
<tr><td>4 (LED + pin)</td><td>0</td></tr>
<tr><td>5 (LED + pin)</td><td>1</td></tr>
<tr><td>6 (LED + pin)</td><td>2</td></tr>
<tr><td>Error (LED)</td><td>3</td></tr>
<tr><td>2 (LED)</td><td>4</td></tr>
<tr><td>Recording (LED)</td><td>5</td></tr>
<tr><td>1 (LED)</td><td>6</td></tr>
</table>
</td></tr>
<tr><td>State</td><td>1 = HIGH
0 = LOW</td></tr>
</table>

EA_IO.CX27DIOControl

QuantumX CX27のデジタル入力/出力の、指定したピンを設定または読み取りを行います。

```
  EA_IO.CX27DIOControl(ByVal Device As Variant, ByVal Action As eDIOAction, ByVal
Pin As eCX27Pin, State As Integer) As Long
```

引数	説明
Device	コマンドの送信先となるデバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。
Action	0 = 入力を読み取り 1 = 出力を設定
Pin	1 = ピン1 (入力) 2 = ピン2 (入力) 3 = ピン3 (出力) 4 = ピン4 (出力)
State	Action=0の場合、入力の状態を返します。それ以外の場合、出力状態を設定します。0=LOW、1=HIGH

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しない、デジタル入力を持っていないなど。

注記

- 単一チャネル(MGCplus、Spider8)のデジタル入力/出力にアクセスするには、EA_IO.ReadDigitalInおよびEA_IO.SetDigitalOutを使用します。

EA_IO.DeviceCount(Read only)

現在の DAQプロジェクト内のすべてのデバイスの数を返します。

```
Property EA_IO.DeviceCount
```

注記

- DAQプロジェクトのチャネルは、1から始まります。

EA_IO.DevicePos

名前、IPアドレスやUUIDに基づいて、DAQプロジェクトのリスト(**Channel settings**ウィンドウ)からデバイスの位置を返します。リストのデバイスは1から始まります。
デバイスの位置(またはインデックス)は、EA_IO.Commandのメソッドが必要です。

```
EA_IO.DevicePos(ByVal DeviceName As String) As Long
```

引数	説明
DeviceName	デバイス名。名前、IPアドレス、デバイスのUUID。

戻り値	説明
>0	チャネル位置。
<0	エラー。デバイスが存在しないなど。

注記

- デバイス位置は、デバイスのスキャン後か、DAQプロジェクト読み込み後に確立されます。

EA_IO.EnableControllers

QuantumX MX878/879モジュールのアナログ出力とデジタル出力を有効/無効に設定します。

```
EA_IO.EnableControllers(ByVal Mode As Integer)
```

引数	説明
Mode	新しい状態 (0 = オフ、1 = オン)

EA_IO.FlashLED

QuantumX、PMX、espressoDAQ、CANHEAD/ML74、CANHEADdirect、またはeDAQスマート モジュール上のチャネルLEDを点滅させます。

```
EA_IO.FlashLED(ByVal Channel As Variant, ByVal Mode As Integer) As Long
```

引数	説明
Channel	チャネル名、または位置。DAQプロジェクトのチャネルは、1から始まります。
Mode	0 = LEDオフ 1 = LEDオン

戻り値	説明
0	エラーなし。
<0	エラー。サポートしていないモジュールなど。

EA_IO.GetDeviceNameFromPos

名前、IPアドレスやUUIDに基づいて、DAQプロジェクトのリスト(**Channel settings**ウィンドウ)からデバイスの名前を返します。リストのデバイスは1から始まります。

```
EA_IO.GetDeviceNameFromPos(ByVal Index As Integer, DeviceName As String)
```

引数	説明
Index	デバイス位置。
DeviceName	デバイス名。

戻り値	説明
>0	チャネル位置。
<0	エラー。デバイスが存在しないなど。

注記

- デバイス位置は、デバイスのスキャン後か、DAQプロジェクト読み込み後に確立されます。

EA_IO.GetFormula

既存のオンライン演算チャネルの数式文字列を取得します。
EA_IO.SetFormulaも参照してください。

このメソッドを使用すると、オンライン演算チャネルの式に直接アクセスし、それを自身で完全に変更できます。ただし、catman演算チャネルの仕組みに精通していることが前提です。

```
EA_IO.GetFormula(ByVal Channel As Variant, strFormula As String)
```

引数	説明
Channel	チャネル名、または位置。DAQプロジェクトのチャネルは、1から始まります。
strFormula	数式。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- このメソッドは、既存のオンライン演算チャネルにのみ適用できます。
- 演算チャネルの種類によっては、数式が"~"などの区切り文字を含む複雑な文字列になる場合があります。代数式の場合、それは単なる式ですが(例: "PressureLeft + PressureRight/2")。ロゼットなどの他の場合は、"ROSETTE~Strain_a~Strain_b~Strain_c~45~235000~0.3~0.14~0.14~0.14~11"のようになります。SENSOR/FUNCTION列の上にマウスを置いていくつかの数式文字列を調べると、ほとんどの式の構造を簡単に理解できます。

EA_IO.GetSlotInfo

デバイス内のチャネルのスロットを検索します。MGCplusマルチチャネルプラグインモジュールでは、サブチャネル数も返します。このメソッドは、EA_IO.Commandを使用してハードウェアチャネルを選択し、ハードウェア チャネルを選択できるようにするために必要な情報を取得します。

```
EA_IO.GetSlotInfo(ByVal Channel As Variant, Slot As Integer, SubChan As Integer) As
Long
```

引数	説明
Channel	チャネル名、または位置。DAQプロジェクトのチャネルは、1から始まります。
Slot	スロット。
SubChan	サブチャネル(単一チャネルのプラグインモジュールまたはSpider8の場合、0)

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
Sub SetZeroMGC()
  'Zero balancing a channel
  Dim Answer
  Dim Slot As Integer, SubChan As Integer
  EA_IO.GetSlotInfo "Valve pressure", Slot, SubChan
  Answer = EA_IO.Command(1, "PCS" & Slot)
  Answer = EA_IO.Command(1, "SPS" & SubChan)
  Answer = EA_IO.Command(1, "CDV")
End Sub
```

EA_IO.InitializeChannel

DAQプロジェクトのチャネルを初期化したり、チャネルのトレーサビリティデータを更新します。通常catmanEasy/APは、DAQジョブの起動時に自動的にチャネルの初期化を行います。しかしEA_IO.AdaptSensorを使用してセンサのプロパティを構成したり、EA_IO.Commandを使用してデバイスに直接コマンドを送信する場合、EA_IO.NoAutoInitで自動初期化を防ぎ、*EA_IO.InitializeChannel*を使用して手動で実行することができます。

```
EA_IO.InitializeChannel(ByVal Channel As Variant, ByVal Mode As Integer) As Long
```

引数	説明
Channel	チャネル名、または位置。DAQプロジェクトのチャネルは、1から始まります。
Mode	0 =完全初期化。センサデータベースのセンサが適用される場合、すべてのパラメータがハードウェアに転送されます。 1 =設定を行わずトレーサビリティデータの更新のみ。 2 =0と同じですが、センサを割り当てずにハードウェア構成を許可します。このモードはEA_IO.AdaptSensorと一緒に使用します。

戻り値	説明
0	エラーなし。
<0	エラー。

チャネルの初期化およびトレーサビリティデータも参照してください。

注記

- チャネルの初期化には、デバイスの種類や選択したセンサにより、500ms程度かかる場合があります。
- DAQジョブ実行中は、このコマンドのMode = 0またはMode = 2(完全初期化)は行えません。

例

```
  Sub SetSpecialFilter()
    'Sets a special filter frequency in an MGCplus channel
    Dim Answer As String
    EA_IO.InitializeChannel "Valve pressure", 0 'Carry out once-only initialization -
needed at least once!
    EA_IO.ChanSelect "Valve pressure" 'Set filter
    Answer = EA_IO.Command(1, "SFC 142,967")
    EA_IO.InitializeChannel "Valve pressure", 1 'Update traceability data
    EA_IO.NoAutoInit("Valve pressure") = False 'Exclude channel from automat.
initialization
    'Start measurement
    EA_Job.Start 1
  End Sub
```

EA_IO.IsCX22

catmanがCX22上で実行されているかどうかを判断します。

```
Property EA_IO.IsCX22
```

値	説明
0	catmanEasyはCX22上で実行していません。
1	catmanEasyはCX22上で実行しています。

EA_IO.IsDeviceLost

デバイスの接続ステータスを取得します。

```
EA_IO.IsDeviceLost(ByVal Device As Variant) As Long
```

引数	説明
Device	デバイス名またはインデックス。DAQプロジェクトのデバイスは1から始まります。

戻り値	説明
0	デバイスと接続。
1	デバイスの消失。
<0	エラー。デバイスが存在しないなど。

例

```
Sub ResetDevice()
  Dim retVal As Long
  Dim status As Long
  retVal = EA_IO.QuantumX.RebootModule(1)
  status = EA_IO.IsDeviceLost(1)
  While status = 1
    retVal = EA_IO.QuantumX.ReconnectAllModules()
    DoEvents
    EA.Wait 3000
    status = EA_IO.IsDeviceLost(1)
  Wend
  EA_IO.RefreshChanView 0
End Sub
```

EA_IO.IsInOverflow

チャネルのオーバーフローステータスを取得します。

```
EA_IO.IsInOverflow(ByVal Channel As Variant) As Long
```

戻り値	説明
0	オーバーフロー状態ではありません。
-1	オーバーフロー状態です。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

EA_IO.MaxTempStoreChannels (Read only)

すべてのDAQ計測および演算チャネルの最大数を返します。

```
Property EA_IO.MaxTempStoreChannels
```

注記

- これは、実際に使用している数ではなく、使用可能なチャネル数を示します。

EA_IO.Measure

実行中のDAQジョブを必要とせずにチャネルから単一の計測値を取得します。このコマンドはすべてのデバイスタイプ(QuantumX、PMX、MGCplus、Optical Interrogators)に適用することができます。

複数チャンネルのサンプル1つを一度に取得する場合は、EA_IO.Snapshotを参照してください。

対話型モードでは、DAQプロジェクトの**Channel settings**ウィンドウでチャネルを選択(複数可)し、**Signal reading / Update**ボタンが同様の機能です。

```
EA_IO.Measure(ByVal Channel As Variant, Reading As Double, Status As Long) As Long
```

引数	説明
Channel	チャネル名、または位置。DAQプロジェクトのチャネルは、1から始まります。
Reading	計測値。
Status	計測ステータス(ビットマスク): Bit 0 = LIV1 set (MGCplus) Bit 1 = LIV2 set (MGCplus) Bit 2 = LIV3 set (MGCplus) Bit 3 = LIV4 set (MGCplus) Bit 4 = 総オーバーフロー Bit 5 = 正味オーバーフロー Bit 6 = 計算エラー Bit 7 = ビットセットの変更(アンプの設定が変更)

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- このメソッドは、低速計測(毎秒最大10回の計測)にのみ、適しています。計測を高速するにはDAQジョブを実行する必要があります。EA_DAQ.CurrentReadingまたはEA_DAQ.CurrentBlockから、実行中のDAQジョブの計測値にアクセスすることができます。

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
  Sub DataToExcel()
    'Reads a measurement cyclically every 10 seconds and transfers it to an Excel
table.
    Dim xls As Object
    Dim MV As Double, Row As Integer
    Dim T0 As Single
    Set xls = CreateObject("Excel.Application")
    xls.Visible = True
    xls.Workbooks.Add
    T0 = Timer

    Do
      If Timer-T0 > 10 Then
        EA_IO.Measure "Valve pressure", MV, 1
        T0 = Timer
        Row = Row + 1
        xls.Cells(Row,1).Value = MV
      End If
    Loop
  End Sub
```

EA_IO.ModifyCalculatorChannel

計算機 型(代数計算)の演算チャネルを変更します。DAQジョブの実行中でも行うことができます。
EA_IO.CreateCalculatorChannelも参照してください。

```
  EA_IO.ModifyCalculatorChannel(ByVal Channel As Variant, ByVal Expression As String,
ByVal Unit As String) As Long
```

引数	説明
Channel	チャネル名、または位置。DAQプロジェクトのチャネルは、1から始まります。
Expression	式。チャネル名を式内に使用することができます: `1/3*(pressure_1 + pressure_2)`
Unit	チャネルの単位。*Unit*が空きの文字列の場合、チャネルの単位は変更されません。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- 次の演算子および関数式を数式に使用することができます:

  +, -, /, *, ^

  AND, OR, >=, <=, =

  sqrt, log, ln, exp

  sin, cos, tan, arctan

  ceil, floor, min, max

  特殊関数は微分および積分です:

  ```
  deriv(y channel, x channel)
  integrate(y channel, x channel)
  ```

- 式では、有理数(小数点以下の桁がある数値)をカンマではなく、常に小数点を使用して入力する必要があります。

EA_IO.NoAutoInit

チャネルの自動初期化を防ぎます。通常catmanEasy/APは、DAQジョブの起動時に自動的にチャネルの初期化を行います。しかしEA_IO.Commandを直接デバイスに送信する場合、EA_IO.NoAutoInit = Trueで自動初期化を防ぎ、自身でEA_IO.InitializeChannelを実行します。

```
Property EA_IO.NoAutoInit(ByVal Channel As Variant)
```

引数	説明
Channel	チャネル名、または位置。DAQプロジェクトのチャネルは、1から始まります。

戻り値	説明
0	エラーなし。
<0	エラー。

チャネルの初期化およびトレーサビリティデータも参照してください。

例

```
Sub SetSpecialFilter()
  'Sets a special filter frequency in an MGCplus channel
  Dim Answer
  EA_IO.InitializeChannel "Valve pressure", 0 'Execute once-only initialization
  EA_IO.ChanSelect "Valve pressure" 'Set filter
  Answer = EA_IO.Command(1, "SFC 142,967")
  EA_IO.InitializeChannel "Valve pressure", 1 'Update traceability data
  EA_IO.NoAutoInit("Valve pressure") = False 'Exclude channel from automat.
initialization
  'Start measurement
  EA_Job.Start 1
End Sub
```

EA_IO.QuantumX.CANOutCyclicControl

QuantumX MX471 CANモジュールのコネクタまたはQuantumX MX840/MX840Aモジュールの1番目のコネクタでの自動サイクリックCAN送信をアクティブ/非アクティブ化します。

```
  EA_IO.QuantumX.CANOutCyclicControl(ByVal Device As Variant, ByVal Connector As
Integer, ByVal Mode As Integer) As Long
```

引数	説明
Device	MX471またはMX840/MX840Aモジュールの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。
Connector	MX471のCAN コネクタ(1 ～ 4)、またはMX840/MX840Aの1番目のコネクタを指定します。
Mode	0 = サイクリックCAN送信を無効に設定 1 = サイクリックCAN送信を有効に設定

戻り値	説明
0	エラーなし。
<0	エラー。

EA_IO.QuantumX.DigitalIOControl

デジタル出力の設定または、デジタル入力の状態を読み取ります。

⚠ このメソッドは、QuantumX MX879およびCX27のみに適用されます。他のデバイスではサポートしていません。

```
  EA_IO.QuantumX.DigitalIOControl(ByVal Device As Variant, ByVal Action As
eDIOAction, ByVal Pin As Integer, State As Integer) As Long
```

引数	説明
Device	デバイス名または位置。DAQプロジェクトのデバイスは、1から始まります。
Action	0 = DIO_READ 1 = DIO_WRITE
Pin	MX879: 最初のピン番号は1で、32ピンまで利用可能です。 Cx27: デジタル入力ピン = 1または2 デジタル出力ピン = 3または4
State	これは、入力の"論理"状態を示していることを注目してください。電気的状態(High/Low)は、構成された*極性*によっては異なる場合があります。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しない、モジュールでサポートしていないなど。

EA_IO.QuantumX.ECUControl

ECUのCCP/XCPトラフィックをオン/オフに切り替えます。QuantumX MX471 CANモジュールにのみ適用されます。

```
  EA_IO.QuantumX.ECUControl(ByVal Device As Variant, ByVal Connector As Integer,
ByVal Action As Integer)
```

引数	説明
Device	MX471モジュールの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。
Connector	QuantumX MX471のCANコネクタ(1 ～ 4) を指定します。
Action	0 = CCP/XCPトラフィックを無効にする 1 = CCP/XCPトラフィックを有効にする
State	これは、入力の"論理"状態を示していることを注目してください。電気的状態(High/Low)は、構成された*極性*によっては異なる場合があります。

戻り値	説明
0	エラーなし。
<0	エラー。

- この機能が動作するには、MX471がCCP/XCPを使用するように構成されており、データベースファイルが読み込まれている必要があります。これらの機能は、MX Assistantソフトウェアで利用できます。

EA_IO.QuantumX.ExecuteControl

QuantumXの内部制御関数を実行することができます。

この関数はHBM内での使用を目的としており、QuantumXの内部制御に関する十分な知識が必要です。有用な制御のいくつかの例を以下に記載します。

```
  EA_IO.QuantumX.ExecuteControl(ByVal Device As Variant, ByVal CtrlName As String,
ByVal ParamCount As Integer, Params() As Long) As Long
```

引数	説明
Device	コマンドの送信先となるデバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。
CtrlName	制御名(例えばRESETZERO)。
ParamCount	制御が要求する*Params*の要素数。
Params	インデックス0から始まる長整数型の配列。

戻り値	説明
0	エラーなし。
<0	エラー。サポートしていないモジュールなど。

利用可能な制御

名前	CONNECTORRECONNECT
可能なモジュール	MX410, MX840, MX840A, MX440A, MX1609, MX1601, MX590, MX460
引数	*ParamCount* = 1 *Params*(0) = コネクタ番号、0 = すべてのコネクタ
説明	強制的にすべてのコネクタのチェックステップを実行します(センサが接続されているか検出、TEDSなど)

名前	UPDATETEDS
可能なモジュール	MX410, MX840, MX840A, MX440A, MX1609, MX1601, MX590, MX460
引数	*ParamCount* = 2 *Params*(0) コネクタ番号、0 = すべてのコネクタ *Params*(1) 0 = TEDSモードが"Use if available"または"TEDS required"の場合のみ、TEDSを強制 1 = 常にTEDSを強制
説明	コネクタからTEDSを強制的に読み取り、それに応じてアンプを設定します。

名前	INHIBITAUTOADJUSTAUTOCAL
可能なモジュール	MX840、MX840A、MX440A
引数	*ParamCount* = 2 *Params*(0) コネクタ番号、0 = すべてのコネクタ *Params*(1) 0 = 自動校正/自動調整を有効 1 = 自動校正/自動調整を無効
説明	内部自動構成/調整の有効/無効を実行します。

EA_IO.QuantumX.GetLastErrMsg

QuantumX システムによって送信された最後のエラーメッセージを取得します。

```
EA_IO.QuantumX.GetLastErrMsg() As String
```

戻り値
エラーメッセージのテキスト。

EA_IO.QuantumX.GetModuleInfo

QuantumXシステムのモジュール情報を取得します。

```
  EA_IO.QuantumX.GetModuleInfo(ByVal Device As Variant, ByVal PropertyID As Integer,
XMLBuf As String) As Long
```

引数	説明
Device	デバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。
PropertyID	プロパティのID: 4 = NTP品質 15 = PTP品質 24 = 稼働時間、システム時間 26 = EtherCATバスのステータス 33 = ストリーミングログ
XMLBuf	必要な情報が含まれるXMLバッファ。

戻り値	説明
0	エラーなし。
<0	エラー。サポートしていないモジュールなど。

EA_IO.QuantumX.GetXMLView

QuantumXモジュールのXML構成設定を取得します。

⚠ このメソッドは、QuantumXモジュールにのみに適用されます。他のデバイスではサポートしていません。

```
  EA_IO.QuantumX.GetXMLView(ByVal SourceName As String, ByVal View As Integer, strXML
As String) As Long
```

引数	説明
SourceName	デバイス名。
View	利用可能なビューのインデックス値: 1: 通信設定 3: システム情報(ファームウェアのバージョンなど) 5: アナログコネクタ構成 6: オンボード演算チャネル構成 8: 特別なコネクタ構成(MX471のCAN、MX878のデジタルI/OおよびアナログMX410のアナログ出力)
strXML	必要な情報が含まれるXMLバッファ。

戻り値	説明
0	エラーなし。
<0	エラー。モジュールが見つからないなど。

EA_IO.QuantumX.PeakValueControl

MX410、MX411、MX403B、MX460、MX878、およびMX879のQuantumXモジュールのピーク値監視にアクションを実行します(リセット、実行/ホールド、ピーク/フォロー)。

```
  EA_IO.QuantumX.PeakValueControl(ByVal Channel As Variant, ByVal Action As Integer)
As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
Action	0 = ピーク値のリセット。 1 = *実行*に切り替え。ピーク値を監視します。 2 = *ホールド*に切り替え。ピーク値の監視を停止し、最後のピーク値を保持します。 3 = *ピーク*モードに切り替え。ユニットはピーク値を監視します。 4 = *フォロー*モードに切り替え。ユニットは元の信号を保持し、ピークの監視を行いません。

戻り値	説明
0	エラーなし。
<0	エラー。モジュールでサポートしていないなど。

EA_IO.QuantumX.ReadXML

QuantumXモジュールからXMLデータブロックを読み取ります。

QuantumXモジュールとのXMLデータ交換の詳細については、QuantumXモジュールと直接通信を参照してください。

EA_IO.QuantumX.WriteXMLも参照してください。

このメソッドは、QuantumXモジュールにのみに適用されます。他のデバイスではサポートしていません。

```
  EA_IO.QuantumX.ReadXML(ByVal Device As Variant, ByVal XMLSection As String, ByVal
RcvBuf As String, ByVal MaxCount As Long) As Long
```

引数	説明
Device	デバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。
XMLSection	XMLデータブロックを読み取るターゲット(XPath)。 catmanの"Channel info"ダイアログで完全なXML構造を見つけてパスを表示することもできます。
RcvBuf	XMLデータブロックのバッファ。
MaxCount	最大読み込み文字数。

戻り値	説明
0	成功。
-32766	エラー。追加のエラー情報を取得するには、EA.LastErrorおよびEA.LastErrorTextを使用してください。

例

```
  Sub ChangeOutputRate
    'Use predefined commands to specify the XPath
    EA_IO.QuantumX.WriteXML 1, "", "SELECT_VIEW 5"
    EA_IO.QuantumX.WriteXML 1, "", "SELECT_CONNECTOR 1"
    EA_IO.QuantumX.WriteXML 1, "", "SELECT_SIGNAL 1"
    'Change the output rate of the 1st connector
    EA_IO.QuantumX.WriteXML 1, "", "<SyncSignal><OutputRate>1200</OutputRate></
SyncSignal>"
  End Sub

  Sub GetAdditionalConnectorXMLView
    'Get the whole XML view of the additional connectors of a MX410
    Dim strBuf As String
    EA_IO.QuantumX.GetXMLView "MX410_Test", 8, strBuf
    MsgBox strBuf
  End Sub

  Sub Edit_MX410AnalogOut_Scale
    Dim X1 As String, Y1 As String, X2 As String, Y2 As String
    Dim XPath As String

    XPath = "/AdditionalConnector/Connector[@address=""1""]/Mode/AnalogOut/Scaling/
Scalingtype/LinTable"

    'Read out the current scaling values of the 1st analog output of MX410
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""1""]/X", X1, 1024
    MsgBox X1
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""1""]/Y", Y1, 1024
    MsgBox Y1
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""2""]/X", X2, 1024
    MsgBox X2
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""2""]/Y", Y2, 1024
  MsgBox Y2
```

```
    'Change scaling values
    EA_IO.QuantumX.WriteXML 1, XPath & "/Point[@address=""1""]", "<X>0.11</X>"
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""1""]/X", X1, 1024
    MsgBox X1
    EA_IO.QuantumX.WriteXML 1, XPath & "/Point[@address=""1""]", "<Y>0.12</Y>"
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""1""]/Y", Y1, 1024
    MsgBox Y1
    EA_IO.QuantumX.WriteXML 1, XPath & "/Point[@address=""2""]", "<X>9</X>"
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""2""]/X", X2, 1024
    MsgBox X2
    EA_IO.QuantumX.WriteXML 1, XPath & "/Point[@address=""2""]", "<Y>10</Y>"
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""2""]/Y", Y2, 1024
    MsgBox Y2
End Sub
```

EA_IO.QuantumX.RebootModule

QuantumXモジュールを再起動します。

```
EA_IO.QuantumX.RebootModule(ByVal Device As Variant) As Long
```

引数	説明
Device	デバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。

戻り値	説明
0	成功。
<0	エラー。モジュールでサポートしていないなど。

例

```
Sub ResetDevice()
  Dim retVal As Long
  Dim status As Long
  retVal = EA_IO.QuantumX.RebootModule(1)
  EA.Wait 30000
  status = EA_IO.IsDeviceLost(1)
  While status = 1
    retVal = EA_IO.QuantumX.ReconnectAllModules()
    DoEvents
    EA.Wait 5000
    status = EA_IO.IsDeviceLost(1)
  Wend
  EA_IO.RefreshChanView 0
End Sub
```

EA_IO.QuantumX.ReconnectAllModules

すべてのデバイスに再接続します。

```
EA_IO.QuantumX.ReconnectAllModules() As Long
```

戻り値	説明
0	エラーなし。
<0	エラー。

例

```
Sub ResetDevice()
  Dim retVal As Long
  Dim status As Long
  status = EA_IO.IsDeviceLost(1)
  While status = 1
    retVal = EA_IO.QuantumX.ReconnectAllModules()
    DoEvents
    EA.Wait 3000
    status = EA_IO.IsDeviceLost(1)
  Wend
  EA_IO.RefreshChanView 0
End Sub
```

EA_IO.QuantumX.SetArbitrarySignalGenerator

このメソッドは、MX878/M879モジュールのアナログ出力の任意信号発生器の波形データを転送します。

```
  Public Function EA_IO.QuantumX.SetArbitrarySignalGenerator(ByVal Device As Variant,
ByVal Connector As Integer, ByVal SampleList As String, ByVal SampleReps As Long,
ByVal LoopMode As Integer, ByVal StartMode As Integer, Optional ByVal SampleFile As
String) As Long
```

引数	説明
Device	MX878/879モジュールの名前または位置。。DAQプロジェクトのデバイスは、1から始まります。
Connector	アナログ出力の番号(1...8)
SampleList	空白で区切られたデータポイントのリスト(例: "10.1 10.5 11.3 ....)。データポイントは必ずしも電圧(-10...10)単位である必要はなく、電圧に変換される任意の値にすることができることに注意してください。スケーリングの設定は、catmanの"Configure Analog-Out"(リボンバーまたはチャネルリストのデバイスのコンテキストメニューの"OnBoard"機能を参照)、またはHBM MX Assistantから行うことができます。 *SampleFile*が指定されている場合、*SampleList*パラメータは無視されます。
SampleReps	単一のデータポイントが出力される頻度を示します(デフォルト = 1)。MX878/879信号発生器は常に96kHzのレートで電圧を出力するため、この繰り返し係数は、より低い出力レートを可能にする効果的な方法となります。たとえば、*SampleReps*を96に設定すると、アナログ出力は1ミリ秒ごとにのみ新しい電圧値を供給します。
LoopMode	0 = 波形データの終わりに達すると、出力は再び最初のサンプルから始まります。 1 = 出力には1つのサンプルのみが含まれます。
StartMode	サンプルバッファを変更する条件を指定します。 0 = できるだけ早くサンプル データに変更 1 = 現在のサイクルで"古い"サンプル データの終了後に変更
SampleFile	オプションです。波形データを含むテキストファイルを指定します。データは1つの列にあり、各行(CRLFで終了)には1つの値が含まれている必要があります。 *SampleFile*が指定されている場合、*SampleList*パラメータは無視されます。

注記

- このメソッドが機能するには、信号発生器がすでに"Arbitrary"および"Buffered mode"に正しく設定されている必要があります。この目的にはHBM MX Assistantを使用する必要があります。信号発生器の構成はcatmanから行うことはできません。

- *Connector*で指定するアナログ出力は"Source=Signal Generator"に設定する必要があります。これは、catmanの"Configure Analog-Out"(リボンバーまたはチャネルリストのデバイスのコンテキストメニューの"OnBoard"機能を参照)、またはHBM MX Assistantから行うことができます。

- ","または"."を使用できます。波形データの小数点区切り文字として使用します。

- 波形出力を一時停止および再開するには、EA_IO.QuantumX.ExecuteControlメソッドを使用します。

- 組み込みの任意信号発生器を使用する代わりに、EA_IO.SetAnalogOutメソッドも考慮します。リアルタイム動作と高速応答時間が重要でない場合、この方法は信号発生器を使用するよりも汎用性が高くなります。

EA_IO.QuantumX.SetChargeAmpSig

この関数を使用すると、QuantumXモジュールに接続されたチャージアンプ(HBM CMD600など)の信号モードを制御することができます。

```
  EA_IO.QuantumX.SetChargeAmpSig(ByVal Channel As Variant, ByVal SigMode As Integer)
As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
SigMode	0 = 計測モード 1 = 放電モード: このモードでは、アンプは連続放電を行います。 アンプの放電中は、DAQジョブを開始しないでください!

戻り値	説明
0	エラーなし。
<0	エラー。モジュールでサポートしていないなど。

注記

- この関数は、DC電圧センサが割り当てられたチャネルのにみ適用されます。
- この関数は、MX410、MX411、MX440AまたはMX840Aモジュールに属するチャネルのみに適用されます。他のすべてのモジュールタイプには適用されません。

EA_IO.QuantumX.SetHighspeedMode

QuantumX MX410/MX411/MX403のコネクタを高速モードに切り替えます(192kHz)。

```
  EA_IO.QuantumX.SetHighspeedMode(ByVal Device As Variant, ByVal Mode As Integer) As
Long
```

引数	説明
DeviceID	コマンドの送信先となるデバイスの名前またはインデックス。DAQプロジェクトのデバイス指標は、1から始まります。
Mode	0 = 高速モード OFF 1 = 高速モード ON

戻り値	説明
0	エラーなし。
<0	エラー。サポートしていないモジュールなど。

EA_IO.QuantumX.SetSamplerateDomain

サンプルレートドメインを設定します。

```
  EA_IO.QuantumX.SetSamplerateDomain(ByVal Device As Variant, ByVal Domain As
eSamplerateDomains) As Long
```

引数	説明
Device	コマンドの送信先となるデバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。
Domain	サンプルレートドメインの種類。 0 = SRD_STANDARD(クラシックレートドメイン) 1 = SRD_DECIMAL(10進数レートドメイン) 2 = SRD_BINARY(バイナリレートドメイン)

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- 変更は、プロジェクトに含まれるすべてのモジュールに適用されます。プロジェクト内で、サンプルレートドメインを混合して使用することはできません!
- モジュールのすべてのチャネルは、同一のサンプルレートドメインを使用します。すなわち、レートのドメイン設定はグローバル設定です。
- Aタイプのハードウェアプラットフォームではサンプルレートドメインの変更はできません - クラシックレートドメインのみをサポートしています。
- 変更してもすぐにモジュールで有効になりません。ドメインに固有のサンプルレートが割り当てられるよう、モジュールを再起動する必要があります。
- 10進数レートではシリアルレートと同様のフィルタ周波数をサポートしていますが、カットオフ周波数の減衰はクラシックレートでは-1dbはなく、-3dbになります。

例

```
Sub SetDecimalrate()
  Dim retVal As Long
  Dim status As Long
  retVal = EA_IO.QuantumX.SetSampleRateDomain(1,SRD_DECIMAL)
  retVal = EA_IO.QuantumX.RebootModule(1)
  EA.Wait 30000
  status = EA_IO.IsDeviceLost(1)
  While status = 1
    retVal = EA_IO.QuantumX.ReconnectAllModules()
    DoEvents
    EA.Wait 5000
    status = EA_IO.IsDeviceLost(1)
  Wend
  EA_IO.RefreshChanView 0
End Sub
```

EA_IO.QuantumX.SetShunt

この関数を使用すると、QuantumXのMX410、MX411またはMX1615モジュールのチャネルにシャント抵抗を適用することができます。

```
  EA_IO.QuantumX.SetShunt(ByVal Channel As Variant, ByVal ShuntMode As Integer) As
Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
SigMode	0 = シャントOFF 1 = シャントON チャネルがシャントオン中は、DAQジョブを開始しないでください!

戻り値	説明
0	エラーなし。
<0	エラー。モジュールでサポートしていないなど。

注記

- この関数は、ブリッジセンサが割り当てられたチャネルにのみ適用されます。
- この関数は、MX410、MX411、またはMX1615モジュールのみに適用されます。他のすべてのモジュールタイプには適用されません。

EA_IO.QuantumX.SetTestSignalMode

希望する仮想テスト信号タイプにチャネルを切り替えます。

```
  EA_IO.QuantumX.SetTestSignalMode(ByVal Channel As Variant, ByVal Mode As
eTestSignalMode, ByVal MinLevel As Double, ByVal MaxLevel As Double, ByVal
FrequencyHz As Double, ByVal DutyCyclePercent As Double) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
Mode	テスト信号の種類 0 = TESTSIGNAL_OFF(無効) 1 = TESTSIGNAL_MANUAL(定数) 2 = TESTSIGNAL_RECT(周波数、アンプおよびデューティサイクルを選択可能な矩形波) 3 = TESTSIGNAL_SINE(周波数、アンプおよびデューティサイクルを選択可能な正弦波) 4 = TESTSIGNAL_TRI(周波数、アンプおよびデューティサイクルを選択可能な三角波)
MinLevel	信号範囲の最小値。
MaxLevel	信号範囲の最大値。
FrequencyHz	周波数(Hz)。
DutyCyclePercent	テスト信号がアクティブになる1周期の割合(%)。

戻り値	説明
0	成功。
<0	エラー。追加のエラー情報を取得するには、EA.LastErrorおよびEA.LastErrorTextを使用します。

注記

- テスト信号を有効にすると、センサ入力がアンプから切断され、実際の計測信号の代わりに仮想信号が配信されます。
- EA_IO.QuantumX.SetTestSignalValueメソッドを使用して*定数*タイプのテスト信号値を変更します。

EA_IO.QuantumX.SetTestSignalValue

*定数*タイプのテスト信号値を設定します。

```
  EA_IO.QuantumX.SetTestSignalValue(ByVal Channel As Variant, ByVal Value As Double)
As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
Value	テスト信号の定数値。

戻り値	説明
0	成功。
<0	エラー。追加のエラー情報を取得するには、EA.LastErrorおよびEA.LastErrorTextを使用します。

注記

- このメソッドは、テスト信号タイプが*定数*タイプにのみ適用されます。
- テスト信号を有効にすると、センサ入力がアンプから切断され、実際の計測信号の代わりに仮想信号が配信されます。
- 他の信号タイプはEA_IO.QuantumX.SetTestSignalModeメソッドを使用してください。

例

```
Sub SetConstSignal()
  EA_IO.QuantumX.SetTestSignalMode "MX840_CH_1", TESTSIGNAL_MANUAL, 0, 0, 0, 0
  EA_IO.QuantumX.SetTestSignalValue "MX840_CH_1", 3.4
End Sub
```

EA_IO.QuantumX.SetToAnalog

QuantumX MX840またはMX840Aの一番目のコネクタを、アナログモードに設定します。

⚠ このメソッドはQuantumX MX840およびMX840Aにのみに適用されます。他のデバイスではサポートしていません。

```
EA_IO.QuantumX.SetToAnalog(ByVal Device As Variant) As Long
```

引数	説明
Device	コマンドの送信先となるデバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。

戻り値	説明
0	成功。
<0	エラー。追加のエラー情報を取得するには、EA.LastErrorおよびEA.LastErrorTextを使用します。

注記

- CAN-busに切り替えるにはEA_IO.QuantumX.SetToCANメソッドを使用してください。
- QuantumX MX840またはMX840Aの最初のコネクタがすでにアナログ モードになっている場合、関数は何もせず、エラーなしですぐに戻ります。

EA_IO.QuantumX.SetToCAN

MX840またはMX840Aの一番目のコネクタをCAN信号に割り当て、アナログからCAN-busモードにコネクタを切り替えます。

⚠ このメソッドはQuantumX MX840およびMX840Aにのみに適用されます。他のデバイスではサポートしていません。

```
  EA_IO.QuantumX.SetToCAN(ByVal Device As Variant, ByVal CANSignalCount As Integer,
ByVal SensorName As String, ByVal TID As String) As Long
```

引数	説明
Device	コマンドの送信先となるデバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。
CANSignalCount	CAN信号のチャネル数。8、16、32、64および128
SensorName	CANのセンサ名。
TID	CANセンサのID。

戻り値	説明
0	成功。
<0	エラー。追加のエラー情報を取得するには、EA.LastErrorおよびEA.LastErrorTextを使用します。

注記

- このメソッドは、MX840またはMX840Aの一番目のコネクタのみに利用可能です。
- QuantumX MX840およびMX840AのCAN信号の最大数は128です。
- アナログモードに切り替えるにはEA_IO.QuantumX.SetToAnalogメソッドを使用してください。
- QuantumX MX840またはMX840Aの最初のコネクタがすでにCANバスモードになっている場合、関数は何もせず、エラーなしですぐに戻ります。

EA_IO.QuantumX.SinglePointReading

この関数を使用すると、DAQジョブを開始することなく、任意のチャネルから計測値を取得することができます。またこのメソッドは、信号2から値を取得するだけでなく、電気単位で値を取得することもできます(catmanのDAQジョブは信号1のみをサポート)。

```
EA_IO.QuantumX.SinglePointReading(ByVal Channel As Variant, MV As Double, Optional
ByVal SignalNum As Integer, Optional ByVal ScaleType As Integer) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
MV	読み取り値。
SignalNum	任意です。省略した場合、信号1を渡します。
ScaleType	1 = 電気単位 2 = 工学単位(省略した場合のデフォルト)

戻り値	説明
0	エラーなし。
<0	エラー。モジュールでサポートしていないなど。

注記

- この関数の応答時間は、2から5msの範囲です。

EA_IO.QuantumX.WriteXML

QuantumXモジュールへXMLデータブロックを書き込みます。

QuantumXモジュールとのXMLデータ交換の詳細については、QuantumXモジュールと直接通信を参照してください。

EA_IO.QuantumX.ReadXMLも参照してください。

このメソッドは、QuantumXモジュールにのみに適用されます。他のデバイスではサポートしていません。

```
  EA_IO.QuantumX.WriteXML(ByVal Device As Variant, ByVal XMLSection As String, ByVal
Data As String) As Long
```

引数	説明
Device	デバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。
XMLSection	XMLデータブロックが移動する場所へのターゲット(XPath)。 catmanの"Channel info"ダイアログで完全なXML構造を見つけてパスを表示することもできます。 XPathが特別なWriteXMLコマンドを使用して事前に定義されている場合、このパラメータは空になる可能性があります(注記を参照)。
Data	XMLデータブロック。例えば"<ChannelName>Pressure_1<ChannelName>"

戻り値	説明
0	成功。
-32766	エラー。追加のエラー情報を取得するには、EA.LastErrorおよびEA.LastErrorTextを使用してください。

注記

- XPathのターゲット選択は、*Data*引数に特別なコマンドを渡すことにより、事前に定義することができます。ここで指定したXPathは、以降のWriteXML/ReadXMLの呼び出しで有効です。*XMLSection*引数は空にしておく必要があります。

```
SELECT_VIEW Number (View 5 for instance contains all parameters related to a channel:
sensor settings, filters etc.)
  EA_IO.QuantumX.WriteXML(1, "", "SELECT_VIEW 5")

SELECT_CONNECTOR Number
  EA_IO.QuantumX.WriteXML(1, "", "SELECT_CONNECTOR 1")

SELECT_CHANNEL Number (only with CAN Bus mode on connector 1 of MX840)
  EA_IO.QuantumX.WriteXML(1, "", "SELECT_CHANNEL 50")

SELECT_SIGNAL Number
  EA_IO.QuantumX.WriteXML(1, "", "SELECT_SIGNAL 1")
```

例

```
  Sub ChangeOutputRate
    'Use predefined commands to specify the XPath
    EA_IO.QuantumX.WriteXML 1, "", "SELECT_VIEW 5"
    EA_IO.QuantumX.WriteXML 1, "", "SELECT_CONNECTOR 1"
    EA_IO.QuantumX.WriteXML 1, "", "SELECT_SIGNAL 1"
    'Change the output rate of the 1st connector
    EA_IO.QuantumX.WriteXML 1, "", "<SyncSignal><OutputRate>1200</OutputRate></
SyncSignal>"
  End Sub

  Sub GetAdditionalConnectorXMLView
    'Get the whole XML view of the additional connectors of a MX410
    Dim strBuf As String
    EA_IO.QuantumX.GetXMLView "MX410_Test", 8, strBuf
    MsgBox strBuf
  End Sub

  Sub Edit_MX410AnalogOut_Scale
    Dim X1 As String, Y1 As String, X2 As String, Y2 As String
    Dim XPath As String

    XPath = "/AdditionalConnector/Connector[@address=""1""]/Mode/AnalogOut/Scaling/
Scalingtype/LinTable"

    'Read out the current scaling values of the 1st analog output of MX410
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""1""]/X", X1, 1024
    MsgBox X1
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""1""]/Y", Y1, 1024
    MsgBox Y1
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""2""]/X", X2, 1024
    MsgBox X2
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""2""]/Y", Y2, 1024
  MsgBox Y2

    'Change scaling values
    EA_IO.QuantumX.WriteXML 1, XPath & "/Point[@address=""1""]", "<X>0.11</X>"
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""1""]/X", X1, 1024
    MsgBox X1
    EA_IO.QuantumX.WriteXML 1, XPath & "/Point[@address=""1""]", "<Y>0.12</Y>"
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""1""]/Y", Y1, 1024
    MsgBox Y1
    EA_IO.QuantumX.WriteXML 1, XPath & "/Point[@address=""2""]", "<X>9</X>"
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""2""]/X", X2, 1024
    MsgBox X2
    EA_IO.QuantumX.WriteXML 1, XPath & "/Point[@address=""2""]", "<Y>10</Y>"
    EA_IO.QuantumX.ReadXML 1, XPath & "/Point[@address=""2""]/Y", Y2, 1024
    MsgBox Y2
  End Sub
```

EA_IO.ReadDigitalIn

デジタル入力の状態を返します。
EA_IO.SetDigitalOutも参照してください。

MGCplus

1または2個のAP75を備えたML78、または単一チャネルボードのすべてのデジタル入力を1つのビットマスクに結合します(最大16ビット)。デジタル入力ごとに個別のcatmanEasy/APチャネルが作成されることはありません。MGCplusの単一チャネルボードでは、アナログ入力(センサ)と同じように、catmanEasy/APのチャネルを指定します。ML78はデジタルのビットマスクを含むcatmanEasy/APのチャネルを指定します。

Spider8

デジタル入力はSpider8のチャネル8で、デジタル出力と組み合わせてビットマスクとして扱います。従って*Channel*引数で、このチャネルを使用します。

QuantumX MX879

*Channel*引数内のモジュール名を指定します。*BitPos*の範囲は1..32で、モジュールのピンに対応しています。

PMX

PMXのすべてのデジタル入力はビットマスク(8ビット)で統合し、スロット10のサブチャネル1の単一チャネルとして扱います。従って*Channel*引数で、このチャネルを使用します。

```
  EA_IO.ReadDigitalIn(ByVal Channel As Variant, ByVal BitPos As Integer, Status As
Long) As Long
```

引数	説明
`Channel`	チャネル名、または位置。DAQプロジェクトのチャネルは、1から始まります。 MX879はデバイス名。
`BitPos`	ビット位置。すなわち、入力ピン番号。
	Spider8: 入力位置は0から始まり、出力としても使用することができます(入力/出力は0から7)。8から15は入力のみです。
	MGCplus: ML78およびAP75では最大16点入力が利用可能で、単一チャネルプラグインでは8入力が利用可能です。入力位置は0(単一チャネルボード: Remote 1)から始まります。
	MX879: 最初のピン番号は1で、32ピンまで利用可能です。ピンが"出力"に切り替わると、関数は出力の状態を返します。
	PMX: 8点の入力が利用可能です。入力位置は0から始まります。
`Status`	現在の状態(`0` = オフ、`1` = オン)。 これは入力の"論理"状態を示していることに注目してください。電気的状態(High/Low)は、構成された*極性*によっては異なる場合があります。

戻り値	説明
`0`	エラーなし。
`<0`	エラー。チャネルが存在しない、デジタル入力を持たないなど。

注記

- MGCplusの単一チャネルボードは、アナログ入力(センサ)と同じチャネル名や位置を使用します。catmanEasy/APでは、ML78およびSpider8を使用するMGCplusのビットマスクに対してのみ個別のチャネルが作成されます。

- 名前を使用してチャネルを指定すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

EA_IO.RefreshChanView

DAQおよび演算チャネルの編集後(例えばチャネルの名前の変更、移動や削除など)、チャネルリストを更新します。

```
EA_IO.RefreshChanView(ByVal ParentWindow As Integer)
```

引数	説明
ParentWindow	チャネルリストウィンドウの種類: 0 = DAQ チャンネルリストウィンドウ 1 = チャンネル選択リストウィンドウ

EA_IO.RemoveMathChan

DAQプロジェクトの既存の演算チャネルを削除します。

```
  EA_IO.RemoveMathChan(ByVal Channel As Variant, Optional ByVal RefreshChannelView
As Integer) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
RefreshChannelView	演算の削除後にチャネルリストを自動的に再描画する場合は、TRUEに設定します。一度に多くのチャネルを削除する予定がある場合は、このパラメータをFALSEに設定し、完了後にEA_IO.RefreshChanViewでチャネルリストの表示を更新することをお勧めします。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- DAQジョブの実行中は、このメソッドを呼び出すことはできません。

EA_IO.SaveChanTable

チャネルリストをテーブルとしてファイルに保存します。この関数はセルの内容だけでなく、アクティブシート上のセルの書式設定を保存します。

```
EA_IO.SaveChanTable(ByVal FileName As String, ByVal Format As Integer) As Long
```

引数	説明
FileName	ファイル名。
Format	ファイル形式: 0 = HTML 1 = Excel

戻り値	説明
0	エラーなし。
<0	エラー。

例

```
Sub ExportChanList()
  EA_IO.SaveChanTable "C:\Users\catman\Documents\HBM\catmanEasy\DATA\HtmlChanList",
0
  EA_IO.SaveChanTable "C:\Users\catman\Documents\HBM\catmanEasy\DATA\XlsChanLIst",
1
End Sub
```

EA_IO.Scan

新たにデバイスのスキャンを行います。

```
  EA_IO.Scan(ByVal PortMask As Integer, ByVal IPScanMask As String, Optional ByVal
DeviceTypeMask as Integer) As Long
```

引数	説明
PortMask	検索するインタフェースを指定するビットマスク: Bit 0: 予約 Bit 4: TCP/IP 例: PortMask = 16の場合、TCP/IPを検索
IPScanMask	*PortMask*にTCP/IPが含まれれている場合に検索するアドレス範囲。 例: 192.168.169.20-150 192.168.169.2,3,10-15 192.168.169.2,3,10-15;192.168.240.3,4
DeviceTypeMask	任意です。検索するデバイスタイプを指定するビットマスク: Bit 0: デバイスタイプを指定しない Bit 2: MGCplus Bit 3: QuantumX/SomatXR Bit 4: 光インテロゲータ Bit 8: PMX Bit 10: DMP41 指定しない場合、すべてのデバイスタイプを検索します。 例: *DeviceTypeMask* = 20(00010100b)の場合、MGCplusおよび光インテロゲータを検索します。

戻り値	説明
>0	見つかったデバイス数。
<0	エラー。

- デバイスのスキャンを行うと現在のDAQプロジェクトのすべてのチャネルおよび、設定は失われます(センサの割り当て、サンプリングレートグループなど)。

例

```
Sub ScanDevices()
  EA_IO.Scan 16, "192.168.169.100-130"
End Sub
```

EA_IO.SendCANMessage

MGCplusのML71、QuantumXのMX840またはMX471のCANコネクタから、CANメッセージを送信します。

```
  EA_IO.SendCANMessage(ByVal Device As Variant, ByVal Channel As Variant, ByVal MsgID
As Long, ByVal ByteCount As Integer, Bytes() As Byte, Optional ByVal Port As Integer)
As Long
```

引数	説明
Device	コマンドの送信先となるデバイスの名前またはインデックス。DAQプロジェクトのデバイスは、1から始まります。
Channel	MGCplusのML71のみ必要。チャネルは、ML71アンプボードのスロット位置を決定するために使用します。したがって、どのサブチャネル(CAN信号)を指定したかは問題ではありません - catmanは自動的に決定します。 チャネルは名前または、位置で指定することができます。DAQプロジェクトのチャネルは、1から始まります。 MX840またはMX471を使用する場合、0または空きの文字列を指定します。
MsgID	CANメッセージのID。IDが2048より大きい場合、catmanは拡張フレームフォーマットビットIDを自動的に設定します。IDを16進形式で指定するには、VBAの&Hプレフィックスを使用します。例えば255 = &HFF。
ByteCount	メッセージを構成するバイト数。
Bytes()	メッセージを構成するバイトを含む配列(0...7)。
Port	コマンドを送信するCANポート(コネクタ)。 ML71: 1または2 MX840: 常に1 MX471: 1、2、3または4

戻り値	説明
0	エラーなし。
<0	エラー。モジュールでサポートしていないなど。

EA_IO.Sensors.CreateSensor

センサデータベース内に新規のセンサを作成します。

対話型モードでは、Sensor database managerの**New sensor**や**New CAN signal**機能に相当します。

```
  EA_IO.Sensors.CreateSensor(ByVal SensorDatabase As String, ByVal SensorName As
String, SensorID As String, Optional ByVal GroupName As String, Optional ByVal
SyncSensorDB As Integer) As Long
```

引数	説明
SensorDatabase	空の文字列の場合、現在のセンサデータベースを使用します。それ以外の場合、任意のセンサデータベースファイルを指定します。
SensorName	新規センサの名前。
SensorID	センサID。IDをcatmanで自動的に作成させるには、空の文字列を指定します。IDを自身で指定する場合、TIDは16文字を超えず、データベース内で一意である必要があります。 関数が戻った後、引数に新規のセンサIDが含まれます。
GroupName	センサが所属するグループの名前。空の文字列の場合、catmanの"My sensors"グループにセンサを挿入します。
SyncSensorDB	任意(デフォルト = FALSE)です。TRUEの場合、センサデータの変更は、catmanとセンサデータベース間で同期されます。

戻り値	説明
0	エラーなし。
<0	エラー。グループが存在しない、IDが一意でないなど。

EA_IO.Sensors.DisconnectSensor

割り当てられたセンサをDAQプロジェクトのチャネルから切断します。

```
  EA_IO.Sensors.DisconnectSensor(ByVal Channel As Variant, Optional ByVal
NoInitialization As Integer) As Long
```

引数	説明
Channel	チャネル名またはチャネルの位置。DAQプロジェクトのチャネルは、1から始まります。
NoInitialization	オプションです。TRUEに設定すると、センサが切断された後、初期化は実行されません。 EA_IO.InitializeChannelを介してチャネルを初期化するかどうかはプログラマに委ねられます。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

EA_IO.Sensors.SelectSensorも参照してください。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

EA_IO.Sensors.GetSensorIDFromChannel

チャネルに割り当てられているIDおよびセンサ名を取得します。EA_IO.Sensors.WriteSensorおよびEA_IO.Sensors.ReadSensorメソッドの呼び出しにはセンサIDが必要です。

```
  EA_IO.Sensors.GetSensorIDFromChannel(ByVal Channel As Variant, SensorID As String,
SensorName As String) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
SensorID	センサID。
SensorName	センサの名前。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しない、チャネルにセンサが割り当てられていないなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

EA_IO.Sensors.GetSensorIDFromName

センサ名からIDおよびセンサIDを取得します。EA_IO.Sensors.WriteSensorおよびEA_IO.Sensors.ReadSensorメソッドの呼び出しにはセンサIDが必要です。

```
  EA_IO.Sensors.GetSensorIDFromName(ByVal SensorDatabase As String, ByVal SensorName
As String, SensorID As String)
```

引数	説明
SensorDatabase	空の文字列の場合、現在のセンサデータベースを使用します。任意のセンサデータベースファイルを指定することもできます。データベースを指定した場合、以降の*ReadSensor*および*WriteSensor*メソッドでは同じデータベースを使用してください!
SensorName	センサの名前。
SensorID	センサID。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しない、チャネルにセンサが割り当てられていないなど。

EA_IO.Sensors.ReadLastModified

データベースからセンサの最終変更日を読み取ります。

```
  EA_IO.Sensors.ReadLastModified(ByVal SensorDatabase As String, ByVal SensorID As
String, LastModified As Double) As Long
```

引数	説明
SensorDatabase	空の文字列の場合、現在のセンサデータベースを使用します。任意のセンサデータベースファイルを指定することもできます。
SensorID	読み取るセンサのセンサID。GetSensorIDFromChannelまたはGetSensorIDFromNameを使用してIDを取得します。
LastModified	OLEオートメーション日付形式でのセンサの最終変更日。OLEオートメーションの日付は、1899年12月30日の午前0時からの日数を値とする浮動小数点数として実装されます。

戻り値	説明
0	エラーなし。
<0	エラー。グループが存在しない、IDが一意でないなど。

EA_IO.Sensors.ReadSensor

データベースからセンサを読み取ります。通常はセンサの作成や取得、プロパティの一部を変更して書き戻します。このメソッドは、*Sensor data type*(構造体)を使用しています。詳細はSensorsメンバのSensorデータ型および列挙を参照してください。

```
  EA_IO.Sensors.ReadSensor(ByVal SensorDatabase As String, ByVal SensorID As String,
oSensor As Sensor) As Long
```

引数	説明
SensorDatabase	空の文字列の場合、現在のセンサデータベースを使用します。任意のセンサデータベースファイルを指定することもできます。
SensorID	読み取るセンサのセンサID。GetSensorIDFromChannelまたはGetSensorIDFromNameを使用してIDを取得します。
oSensor	Sensor型の変数。

戻り値	説明
0	エラーなし。
<0	エラー。グループが存在しない、IDが一意でないなど。

例

```
  'The following example demonstrates how to change an existing sensor named "C4
20kN":
  Dim MySensor As Sensor
  Dim SensorID As String

  With EA_IO.Sensors
  .GetSensorIDFromName "", "C4 20kN", SensorID
  .ReadSensor "", SensorID, MySensor
  MySensor.P2X = 2.0355
  MySensor.P2Y = 500
  .WriteSensor "", SensorID, MySensor
  End With
```

EA_IO.Sensors.ScanTEDS

*EA_IO.ScanTEDS*は互換性のために残していますが、将来は廃止する予定です。*EA_IO.Sensors*の使用を推奨しています。

チャネルのセンサスキャンを実行します。

対話型モードでは、**Channel settings**ウィンドウのコンテキストメニューや、ツールバーの**Sensor scan**に相当します。

```
EA_IO.Sensors.ScanTEDS(ByVal Channel As Variant, TEDSModel As String, TEDSName As String) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
TEDSModel	変換器モデルを指定。TEDSが見つからない場合、空の文字列。
TEDSName	HBMテンプレートChannel NameをTEDSで使用している変換器名。TEDSが見つからない、またはChannel Nameを設定していない場合は、空の文字列。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しない。

注記

- TEDSクエリは、約2秒かかることがあります。TEDSセンサのアクティベーションが完了するまで、次のアクションを実行しないでください。

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
Sub ScanForTEDS()
  Dim Model As String
  Dim Name As String
  EA_IO.Sensors.ScanTEDS "Valve pressure", Model, Name
End Sub
```

EA_IO.Sensors.SelectSensor

*EA_IO.SelectSensor*は互換性のために残していますが、将来は廃止する予定です。*EA_IO.Sensors*の使用を推奨しています。

DAQプロジェクトのチャネルにセンサデータベースからのセンサを割り当てて、チャネルを初期化します。

対話型モードでの、センサリストからチャネルにセンサをドラッグ&ドロップすることに相当します。

```
  EA_IO.Sensors.SelectSensor(ByVal Channel As Variant, ByVal SensorName As String,
Optional ByVal SensorID As String, Optional ByVal NoInitialization As Integer) As
Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
SensorName	センサの指定。空の文字列が渡された場合、センサーデータベースダイアログで選択したセンサがチャネルに割り当てられます。
SensorID	任意です。指定した場合、センサはこのIDで検索します(センサデータベースの"Sensor-ID"で検索する固有の識別)。*SensorName*は無視されます。
NoInitialization	任意です。TRUEに設定した場合、センサの割り当て後に初期化を行いません。 EA_IO.InitializeChannelを介してチャネルを初期化するかどうかはプログラマに委ねられます。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しない、チャネルでサポートしていないセンサなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
Sub AssignSensor()
  EA_IO.Sensors.SelectSensor "Valve pressure", "P3MB 5bar"
End Sub

Sub AssignSensorWithID()
  EA_IO.Sensors.SelectSensor "Valve pressure", "", "HBM_P3MB_5bar"
End Sub
```

EA_IO.Sensors.UpdateSensorFromDatabase

チャネルがセンサーデータベースから割り当てられたセンサを持っている場合、この関数はデータベースにアクセスし、現在のデータベース設定に従ってチャンネル設定を更新します。チャネルにセンサを割り当てた後、誰かがデータベース内のセンサデータを変更した可能性がある場合に便利です。

対話型モードでは、チャネルリストのコンテキストメニュー**"Update sensor"**に相当します。

```
EA_IO.Sensors.UpdateSensorFromDatabase(ByVal Channel As Variant) As Long
```

引数	説明
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しない、チャネルにセンサが割り当てられていないなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

EA_IO.Sensors.WriteSensor

センサをデータベースへ書き戻します。通常はセンサの作成や取得、プロパティの一部を変更して書き戻します。このメソッドは、*Sensor data type*(構造体)を使用しています。詳細はSensors.メンバのSensorデータ型および列挙を参照してください。

```
  EA_IO.Sensors.WriteSensor(ByVal SensorDatabase As String, ByVal SensorID As String,
oSensor As Sensor, Optional ByVal SyncSensorDB As Integer) As Long
```

引数	説明
SensorDatabase	空の文字列の場合、現在のセンサデータベースを使用します。任意のセンサデータベースファイルを指定することもできます。
SensorID	書き込むセンサのセンサID。GetSensorIDFromChannelまたはGetSensorIDFromNameを使用してIDを取得します。
oSensor	Sensor型の変数。
SyncSensorDB	任意(デフォルト = FALSE)です。TRUEの場合、センサデータの変更は、catmanとセンサデータベース間で同期されます。

戻り値	説明
0	エラーなし。
<0	エラー。グループが存在しない、IDが一意でないなど。

例

```
  'The following example demonstrates how to create a new fullbridge sensor with a
minimal set of parameters:
  Dim MySensor As Sensor
  Dim SensorID As String
  Dim SensorName As String

  SensorID = ""
  SensorName = "MyFullbridge"
  EA_IO.Sensors.CreateSensor "", SensorName, SensorID, ""

  MySensor.Description = SensorName
  MySensor.GenericType = GENTYPE_BRIDGE
  MySensor.SensorType = SETYPE_FULLBRIDGE
  MySensor.EngUnit = "bar"
  MySensor.Excitation = 2.5
  MySensor.ScaleType = SESCALING_TWOPOINTS
  MySensor.P1X = 0
  MySensor.P1Y = 0
  MySensor.P2X = 2.0355
  MySensor.P2Y = 500
  MySensor.RangeMax = 1000
  MySensor.RangeMin = -1000

  EA_IO.Sensors.WriteSensor "", SensorID, MySensor, 1

  'And assign it to a channel.
  EA_IO.Sensors.SelectSensor "PressureChan_1", "MyFullbridge"
```

EA_IO.SetAnalogOut

アナログ出力チャネルに、電圧を印加します。

```
  EA_IO.SetAnalogOut(ByVal Channel As Variant, ByVal Connector As Integer, ByVal
Voltage As Double, ByVal OperMode As Long, Optional ByVal OutputNumber as Integer) As
Long
```

⚠ この関数はMGCplusの単一チャネルボード、PMXおよびQuantumX MX878/MX879およびMX410に適用されます(しかし、将来変更される予定です)。

引数	説明
Channel	MGCplus、MX410: チャネル名 QuantumX MX878/879およびPMX: デバイス名 この理由は、現在のMX878/MX879およびPMXのアナログコネクタは、catmanのチャネルリストとして表示されないためです。
Connector	QuantumX MX878/879: 出力コネクタを指定(1-8)。 PMX: PX878を接続している場合、スロット番号を指定(1-4)。 PMXおよびQuantumX MX878/879のみに必要です。MGCplusでは、この引数は常に1です。
Voltage	印加電圧(V)。
OperMode	0 = コネクタは内部のソース接続へ戻し、*Voltage*は無視されます。 1 = 内部ソースと*Voltage*出力からコネクタを分離します。 この引数はQuantumXのみ考慮され、他は無視されます。
OutputNumber	MGCplus: 0 = 最初の出力設定 1 = 二番目の出力設定 PMX: 1-10 QuantumX MX878/879: 適用しません。無視されます。

注記

⊗ このメソッドは明白に、リアルタイムアプリケーションにも適していません。つまり、非常に速い反応時間を必要とする安全性が重要なアプリケーションには適していません。ハードウェアの種類によっては、反応時間が50msに達する場合があります。QuantumX MX878/MX879モジュールの場合、通常の反応時間は5～10msの範囲です。

EA_IO.SetDigitalOut

デジタル出力の状態を設定します。
EA_IO.ReadDigitalInも参照してください。

MGCplus
1または2個のAP75を備えたML78、または単一チャネルボードのすべてのデジタル出力を1つのビットマスクに結合します(最大16ビット)。デジタル出力ごとに個別のcatmanEasy/APチャネルが作成されることはありません。MGCplusの単一チャネルボードでは、アナログ入力(センサ)と同じように、catmanEasy/APのチャネルを指定します。ML78はデジタルのビットマスクを含むcatmanEasy/APのチャネルをを指定します。

Spider8
デジタル出力ははSpider8のチャネル8で、デジタル出力と組み合わせてビットマスクとして扱います。従って*Channel*引数で、このチャネルを使用します。

QuantumX MX879
*Channel*引数内のモジュール名を指定します。*BitPos*の範囲は1..32で、モジュールのピンに対応しています。"Output"モードに切り替えられたピンのみが反応することに注目してください。

PMX
PMXのすべてのデジタル出力はビットマスク(8ビット)で結合し、スロット10のサブチャネル2の単一チャネルとして扱います。従したがって、デジタルビットマスクを含むcatmanEasy/APチャネルを指定します。

```
  EA_IO.SetDigitalOut(ByVal Channel As Variant, ByVal BitPos As Integer, ByVal Status
As Long) As Long
```

引数	説明
`Channel`	チャネル名、または位置。DAQプロジェクトのチャネルは、1から始まります。 MX879はデバイス名。
`BitPos`	 MX879: 最初のピン番号は1で、32ピンまで利用可能です。ピンが"Output"に切り替えられていないと、この関数の効果がありません!
	ビット位置。すなわち、出力ピン番号。
	Spider8: チャネル8で8出力(入力/出力0から7)が利用可能です。出力位置は0から始まります。
	MGCplus: ML78およびAP75では最大16点の出力が利用可能で、単一チャネルプラグインでは4点の出力が利用可能です。出力位置は0(LIV 1)から始まります。
	PMX: スロット10のサブチャネル2(出力1～ 8)で、8点出力が利用可能です。出力位置は0から始まります。
	⚠ 初めてこのコマンドを使用する場合、デジタル出力は"software controlled"に設定されています。しかし、これは本来の機能を無効にします(デフォルト: リミットスイッチ)。catmanEasy/APを閉じると、変更したすべてのデジタル出力はデフォルトの"Limit switch"(単一チャネルボード)または、"Limit switch + Overflow" (ML78)にリセットされます。
`Status`	新しい状態(`0` = オフ、`1` = オン)。 これは出力の"論理"状態を示していることに注目してください。電気的状態(High/Low)は、構成された*極性*によっては異なる場合があります。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しない、デジタル出力を持たないなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

EA_IO.SetFormula

既存のオンライン演算チャネルの式を設定します。
EA_IO.GetFormulaも参照してください。

このメソッドを使用すると、オンライン演算チャネルの式に直接アクセスし、それを自身で完全に変更できます。ただし、catman演算チャネルの仕組みに精通していることが前提です(注記を参照)。

```
EA_IO.SetFormula(ByVal Channel As Variant, ByVal strFormula As String)
```

引数	説明
Channel	チャネル名、または位置。DAQプロジェクトのチャネルは、1から始まります。
Expression	数式。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- このメソッドは、既存のオンライン演算チャネルにのみ適用できます。
- 演算チャネルの種類によっては、数式が"~"などの区切り文字を含む複雑な文字列になる場合があります。代数式の場合、それは単なる式ですが(例: "PressureLeft + PressureRight/2")。ロゼットなどの他の場合は、"ROSETTE~Strain_a~Strain_b~Strain_c~45~235000~0.3~0.14~0.14~0.14~11"のようになります。SENSOR/FUNCTION列の上にマウスを置いていくつかの数式文字列を調べると、ほとんどの式の構造を簡単に理解できます。

EA_IO.SetName

DAQプロジェクトのチャネルに、名前およびコメント(任意)を割り当てて、チャネルを初期化します。**Transfer channel and device names into device**グループの**Store names in Device**オプションが有効な場合(Optionダイアログの**DAQ channels**タブ)、MGCplusおよびQuantumXのデバイスチャネルにこの名前で書き込みます。

対話型モードでの、DAQプロジェクトの**Channel settings**ウィンドウで**Channel name**列の対応するチャネルをダブルクリックして名前やコメントを指定するのと同様の機能です。

```
EA_IO.SetName(ByVal Channel As Variant, ByVal Name As String, Optional ByVal
Comment As String, Optional ByVal Unit As String) As Long
```

引数	説明
Channel	古いチャネル名、または位置。DAQプロジェクトのチャネルは、1から始まります。
Name	新しいチャネル名。
Comment	任意です。チャネルの新しいコメント。
Unit	任意です。チャネルの新しい単位。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しない、新しいチャネル名は既に使用されているなど。

注記

- チャネル名は一意である必要があります。
- チャネル名には少なくとも2文字以上である必要があります。
- チャネル名は、c1やC22などの内部予約名であってはなりません
- チャネル名は、integral、deriv、Time、floor、ceil、rnd、rms、bit、mod、min、max、asin、acos、atanなどの代数計算に使用されるキーワードの1つであってはなりません
- チャネル名には空白スペースを含めることはできません。スペースはアンダースコア(_)に置き換えられます。
- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
Sub SetSensors()
  EA_IO.SetName "Valve pressure", "Valve pressure_A95"
  EA_IO.SetName 5, "Temperature_1", "Temperature on the outlet valve", "°C"
End Sub
```

EA_IO.SetStatusMsg

Channel settingsウィンドウの**Reading**列にチャネルのテキストおよびシンボルを設定します。

```
  EA_IO.SetStatusMsg(ByVal Channel As Variant, ByVal Msg As String, ByVal Image As
Variant) As Long
```

引数	説明
Channel	チャネル名、または位置。DAQプロジェクトのチャネルは、1から始まります。
Msg	出力するテキスト。
Image	文字列: 読み込む画像のファイル名(.ICO、.BMP、.GIF) 番号: 0 = 画像なし >0 = **Channel settings**ウィンドウの画像リストのインデックス。画像リストには122の画像が含まており、例えば: 5 = 赤色禁止記号 6 = 赤色バツ印 7 = 緑色右矢印 9 = 緑色十字 10 = 赤色マイナス記号 11 = 黄色感嘆符(警告) 19 = オーバーフロー 20 = OK 43 = 青色情報記号

戻り値	説明
0	エラーなし。
<0	エラー。

EA_IO.Snapshot

実行中のDAQジョブを使用せずに、複数チャネルを計測します。

対話型モードでの、DAQプロジェクトの**Channel settings**ウィンドウで選択したチャネルの**Signal reading / Update**と同様の機能です。

```
  EA_IO.Snapshot((ByVal NofChannels As Integer, ChanList() As Variant, MV() As
Double) As Long
```

引数	説明
NofChannels	計測するチャネル数。
ChanList()	チャネル名、または位置のリスト。DAQプロジェクトのチャネルは、1から始まります。
MV()	計測値。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- このメソッドは、低速計測(毎秒最大10回の計測)にのみ、適しています。計測を高速化するにはDAQジョブの実行が必要です。EA_DAQ.CurrentReadingまたはEA_DAQ.CurrentBlockから、実行中のDAQジョブの計測値にアクセスすることができます。

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
  Sub DataToExcel()
    'Reads a measurement of multiple channels cyclically every 10 seconds and
transfers values to an Excel table.
    Dim xls As Object
    Dim Chans(1) As Variant, MV(1) As Double, Row As Integer
    Dim T0 As Single
    Set xls = CreateObject("Excel.Application")
    xls.Visible = True
    xls.Workbooks.Add
    T0 = Timer
    Chans(0) = "Valve pressure"
    Chans(1) = "Temperature"

    Do
      If Timer-T0 > 10 Then
        EA_IO.Snapshot 2, Chans, MV
        T0 = Timer
        Row = Row + 1
        xls.Cells(Row,1).Value = MV(0)
        xls.Cells(Row,2).Value = MV(1)
      End If
    Loop
  End Sub
```

EA_IO.ZeroBalanceAvgTime

EA_IO.ZeroBalanceMethod = 1(s.EA_IO.ZeroBalanceMethod)の場合、サンプルはゼロバランス値を決定するために平均化する時間(秒)を設定します。

対話型モードでの、**Options for zerobalancing**に相当します。

```
Property EA_IO.ZeroBalanceAvgTime
```

注記

- 平均時間(秒) * サンプルレート(Hz)の積は、1024を超えてはいけません。

例

```
EA_IO.ZeroBalanceAvgTime = 5
```

EA_IO.ZeroBalanceControl

チャネルのゼロバランスを実行します。

対話型モードでの、**Zero balance**コンテキストメニューの項目および、ゼロバランス値の手動入力に対応しています。

この関数は、DAQジョブの実行中は行えません。

```
  EA_IO.ZeroBalanceControl(ByVal Channel As Variant, ByVal OpCode As Integer,
Optional Param As Variant) As Long
```

引数	説明
Channel	チャネル名、または位置。ここにゼロが渡されると*OpCode* 1、2、または3が使用されている場合は、DAQプロジェクトのすべてのチャネルがゼロに設定され、ゼロ設定が元に戻されるか、ゼロバランスに対してチャネルがロック/ロック解除されます。*OpCode* 4および5ではゼロは無効です。
OpCode	操作の種類: 1 = ゼロバランス実行(ゼロに設定) 2 = ゼロバランスを元に戻す(メモリをクリア) 3 = ゼロバランスのロック/ロック解除 4 = ゼロバランス値の設定 5 = ゼロバランスメモリの現在値を読み出し
Param	OpCodeのオプション: *OpCode* 1: Omitted *OpCode* 2: Omitted *OpCode* 3: True = ゼロバランスをロック、False = ゼロバランスを許可 *OpCode* 4: ゼロバランス値 *OpCode* 5: メモリ内の現在のゼロバランス値

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- ゼロバンスを実行(OpCode = 1)する場合、決定したゼロバランス値は自動的にDAQプロジェクトに保存されます。
- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例

```
Sub ZeroFunctions()
  Dim Param
  Dim ZeroVal As Double
  'Permit zero balancing
  EA_IO.ZeroBalanceControl "Valve pressure", 3, False
  EA_IO.ZeroBalanceControl "Pump pressure", 3, False
  'Carry out zero balance
  EA_IO.ZeroBalanceControl "Valve pressure", 1
  'Set zero balance value manually
  EA_IO.ZeroBalanceControl "Pump pressure", 4, 75.8
  'Read out current zero balance value
  EA_IO.ZeroBalanceControl "Valve pressure", 5, Param
  ZeroVal = Param
End Sub
```

EA_IO.ZeroBalanceMethod

チャネルのゼロバランスを実行するメソッドを設定します。

対話型モードでの、**Options for zerobalancing**に相当します。

```
Property EA_IO.ZeroBalanceMethod
```

値	説明
0	ゼロバランス値は、デバイスによる単一計測で決定します。
1	ゼロバランス値は、選択したすべてのチャネルをcatmanEasy/APにより同期計測を実行します。このモードは平均化を使用することができます。ゼロバランス値は、いくつかの読み取り値の平均から決定されます。EA_IO.ZeroBalanceAvgTimeも参照してください。

例

```
EA_IO.ZeroBalanceMethod = 1
```

EA_IO.ZeroBalanceSampleRate

ゼロバランスのメソッド1(EA_IO.ZeroBalanceMethod = 1)で平均化を使用している場合、サンプルレート(Hz)を指定します。平均化はEA_IO.ZeroBalanceAvgTimeで指定されている時間行います。有効なサンプルレートは5Hz、10Hz、50Hz、100Hz、および300Hzです。

対話型モードでの、**Options for zerobalancing**に相当します。

```
Property EA_IO.ZeroBalanceSampleRate
```

注記

- 平均時間(秒) * サンプルレート(Hz)の積は、1024を超えてはいけません。

例

```
EA_IO.ZeroBalanceSampleRate = 50
```

SensorsメンバのSensorデータ型および列挙

```
Public Type Sensor
GenericType As eGenSensType
SensorType As eSensType 'ID code for the specific sensor type, e.g 350 = Fullbridge
ElecUnit As String 'Electrical unit, if this is an empty string catman determines it
automatically from the generic sensor type
EngUnit As String 'Physical i.e. engineering unit
SADPara As Integer
Excitation As Single 'Excitation voltage in V, for bridge type sensors
TID As String 'Read only - returns the sensor ID
ScaleType As eSensScaleType
P1X As Single 'Electrical zero if ScaleType=ZEROSPAN, 1st electrical point if
TWOPOINTS
P1Y As Single 'Physical zero if ScaleType=ZEROSPAN, 1st physical point if TWOPOINTS
P2X As Single 'Electrical span if ScaleType=ZEROSPAN, 2nd electrical point if
TWOPOINTS
P2Y As Single 'Physical span if ScaleType=ZEROSPAN, 2nd physical point if TWOPOINTS
RangeMax As Single 'Positive max. range of sensor in physical units
RangeMin As Single 'Negative range of sensor in physical units
NofScalePoints As Integer 'Number of table points or coefficients for LINTABLE or
POLYNIOMIAL scaling
ScaleData(64) As Double 'e.g. lin points x1,y1,x2,y2.... or coeffs a0, a1, a2..... for
LINTABLE or POLYNIOMIAL scaling
GageFactor As Single 'If this is set to a value > 0 the sensor operates in gage
factor scaling delivering microstrains
BridgeFactor As Single
Description As String
Type As String
SNR As String
CalibrationDate As Double 'In OLE date/time format, i.e. days since 1.1.1900
ExpirationDate As Double 'In days
Store As String
Comment As String
Descriptor As String
CAN_Id As Long
CAN_FrameFormat As Integer
CAN_ByteOrder As Integer
CAN_ByteCount As Integer
CAN_RawValueFormat As Integer
CAN_SigType As Integer
CAN_StartBit As Integer
CAN_BitCount As Integer
CAN_ScaleFac As Single
CAN_Offset As Single
CAN_MinVal As Single
CAN_MaxVal As Single
CAN_Unit As String
CAN_ModeSig As Integer
CAN_ModeByteOrder As Integer
CAN_ModeMode As Integer 'Mode value!
CAN_ModeStartBit As Integer
CAN_ModeBitCount As Integer
CAN_BufferValueFormat As Integer
CAN_MaxRepTime As Single
ExcitationFreq As Long '0=Auto,-1=DC,32000=Auto enforce AC, any other values specify
an AC frequency, e.g. 600, 1200, 4800
ActiveSupply As Single
CounterIndexEnable As Integer
CounterIndexDivisor As Integer
SSI_Clock As Integer
SSI_Resolution As Integer
SSI_Encoding As Integer
```

```
ExpectedShuntValue As Single
GlitchFilter As Integer '0=OFF, 1=0.1 µs, 2=1 µs, 3=10 µs, 4=100µs
Polarity As Integer '0=non-inverted, 1=inverted
PulsesPerRound As Long
CAN_ActiveQueryEnabled As Integer
CAN_ActiveQuery As String
CAN_ActiveQueryRate As Single
MeasRangeMicroStrain As Long 'Max. range if measuring microstrains, i.e. if
GageFactor > 0
BridgeResistance As Long 'Not mandatory, set to 0 if unknown
IsUnique As Integer
NeedsAdaptation As Integer
End Type

Generic sensor types
Enum eGenSensType
GENTYPE_STRAINGAGE = 1
GENTYPE_BRIDGE = 2
GENTYPE_LVDT = 3
GENTYPE_VOLTAGE = 4
GENTYPE_CURRENT = 5
GENTYPE_FREQUENCY = 6
GENTYPE_PT = 7
GENTYPE_THERMOCOUNPLE = 8
GENTYPE_PIEZOACTIVE = 9
GENTYPE_RESISTOR = 10
GENTYPE_INDUCTIVEBRIDGE = 11
GENTYPE_INCREMENTAL = 12
GENTYPE_PIEZOPASSIVE = 13
GENTYPE_POTENTIOMETER = 14
GENTYPE_PWM = 15
GENTYPE_TIME = 16
GENTYPE_SSI = 30
GENTYPE_IEPE = 31
GENTYPE_CANSIGNAL = 100
End Enum

Specific sensor types
Enum eSensType
SETYPE_FULLBRIDGE = 350
SETYPE_HALFBRIDGE = 351
SETYPE_DMS_FULL = 353
SETYPE_DMS_HALF = 354
SETYPE_DMS_QUART = 355
SETYPE_DMS_QUART_S8 = 352
SETYPE_IND_FULL = 356
SETYPE_IND_HALF = 357
SETYPE_DMS_FULL_LOW = 358
SETYPE_DMS_HALF_LOW = 359
SETYPE_DMS_FULL_HIGH = 360
SETYPE_DMS_HALF_HIGH = 361
SETYPE_DMS_FULL_120 = 362
SETYPE_DMS_FULL_350 = 363
SETYPE_DMS_FULL_700 = 364
SETYPE_DMS_HALF_120 = 365
SETYPE_DMS_HALF_350 = 366
SETYPE_DMS_HALF_700 = 367
SETYPE_DMS_QUART_120_4L = 368
SETYPE_DMS_QUART_350_4L = 369
SETYPE_DMS_QUART_700_4L = 370
SETYPE_DMS_QUART_120_3L = 371
SETYPE_DMS_QUART_350_3L = 372
SETYPE_DMS_QUART_700_3L = 373
```

```
SETYPE_DMS_QUART_1000_3L = 374
SETYPE_DMS_QUART_1000_4L = 376
SETYPE_DC_VOLT = 420
SETYPE_DC_VOLT_75M = 425
SETYPE_DC_VOLT_10 = 426
SETYPE_DC_VOLT_100MV = 429
SETYPE_DC_CURR = 421
SETYPE_DC_CURR_20M = 427
SETYPE_THERMO_J = 450
SETYPE_THERMO_K = 451
SETYPE_THERMO_T = 452
SETYPE_THERMO_S = 453
SETYPE_THERMO_B = 454
SETYPE_THERMO_E = 455
SETYPE_THERMO_R = 456
SETYPE_THERMO_N = 457
SETYPE_OHM = 475
SETYPE_OHM_500 = 476
SETYPE_OHM_5000 = 477
SETYPE_PT10 = 500
SETYPE_PT100 = 501
SETYPE_PT500 = 502
SETYPE_PT1000 = 503
SETYPE_FREQ_1KHZ = 536
SETYPE_FREQ_2KHZ = 530
SETYPE_FREQ_10KHZ = 535
SETYPE_FREQ_20KHZ = 531
SETYPE_FREQ_100KHZ = 534
SETYPE_FREQ_200KHZ = 532
SETYPE_FREQ_500KHZ = 538
SETYPE_FREQ_1MHZ = 533
SETYPE_FREQ_2MHZ = 537
SETYPE_FREQ = 520
SETYPE_FREQ_FuD = 521
SETYPE_FREQ_1P1F = 529
SETYPE_FREQ_2P1F = 522
SETYPE_FREQ_2P2F = 523
SETYPE_PWM = 527
SETYPE_DURATION = 524
SETYPE_COUNT = 525
SETYPE_COUNT_1K = 526
SETYPE_DELTA = 555
SETYPE_DELTA_01 = 550
SETYPE_DELTA_1 = 551
SETYPE_DELTA_10 = 552
SETYPE_COULOMB = 570
SETYPE_COULOMB_01 = 571
SETYPE_COULOMB_1 = 572
SETYPE_COULOMB_10 = 573
SETYPE_COULOMB_100 = 574
SETYPE_POTI = 385
SETYPE_RESISTOR = 475
SETYPE_LVDT = 380
SETYPE_MX460_PULSEWIDTHMODULATION = 539
SETYPE_MX460_PULSEWIDTH = 540
SETYPE_MX460_PULSEDURATION = 541
SETYPE_MX460_IND_COUNTER = 590
SETYPE_MX460_IND_FREQ = 591
End Enum
```

```
Scaling types
Enum eSensScaleType
SESCALING_ZEROSPAN = 1
SESCALING_LINTABLE = 2
SESCALING_POLYNOMIAL = 4
SESCALING_TWOPOINTS = 32
End Enum
```

EA_Job

EA_JobからcatmanEasy/APのDAQジョブにアクセスします。

このオブジェクトでは、以下が行えます:

- DAQジョブの作成/削除: EA_Job.Add、EA_Job.Delete
- DAQプロジェクトで利用可能なジョブ数を取得: EA_Job.JobCount
- 実行中ジョブのインデックスを取得: EA_Job.CurrentJob
- 反復実行中の現在の反復数を取得: EA_Job.CurrentJobRepetition
- DAQジョブの開始/終了: EA_Job.Start、EA_Job.Terminate
- DAQジョブ設定の取得/設定: EA_Job.GetProperty、EA_Job.SetProperty
- チャネルのアクティブ状態を取得/設定: EA_Job.GetChannelActivation、EA_Job.SetChannelActivation
- チャネルのフィルタ設定を取得/設定: EA_Job.GetFilter、EA_Job.SetFilter
- チャネルのエクスポートモードの取得/設定: EA_Job.GetExportMode、EA_Job.SetExportMode
- チャネルのストレージモードの取得/設定: EA_Job.GetStorageMode、EA_Job.SetStorageMode
- 統計ジャーナルのアクティブ化の取得/設定: EA_Job.GetJournalActivation、EA_Job.SetJournalActivation
- テストパラメータ数を取得: EA_Job.GetTestParameterCount
- テストパラメータの名前および値を取得: EA_Job.GetTestParameter
- テストパラメータの作成、修正、削除: EA_Job.AddTestParameter、EA_Job.ModifyTestParameter、EA_Job.DeleteTestParameter
- DAQジョブの測定データの最終保存を実行: EA_Job.StoreData
- DAQジョブの現在のステータスを取得: EA_Job.Status、EA_Job.GetStatusInfo
- 秒単位のリアルタイムラグを取得: EA_Job.RTLag
- 残りの計測可能時間を秒で取得: EA_Job.TimeLeft
- チャネルで取得した値数を取得: EA_Job.SamplesAcquired
- デジタルトリガの設定を取得: EA_Job.GetDigitalTriggerSettings
- 計測データのファイル名にするグローバルプレースホルダカウンタの現在値を取得/設定: EA_Job.GlobalTestCounter

以下のアクションは、現在のバージョンでサポートされていません:

- MGCplusのPCカードにDAQジョブを転送

このオブジェクトでは以下のメソッドが公開されています:

EA_Job.Add ・・・

EA_Job.ModifyTestParameter ・・・

EA_Job.Add

標準値を使用して新規DAQジョブを作成します。EA_Job.SetPropertyで値を変更します。

```
EA_Job.Add(ByVal Name As String) As Long
```

引数	説明
Name	新規DAQジョブの名前。

戻り値	説明
>0	DAQジョブ番号。
<0	エラー。

- DAQジョブ設定の標準値は、**DAQ jobs**ダイアログボックスのものに相当します。

例

```
Sub RunJobs()
  Dim i As Integer, numJobs As Integer, indexJob As Integer
  Dim numValues_D As Long, numValues_S As Long, numValues_F As Long
  Dim T0 As Single

  EA_Job.Add("NewJob") 'Add a new DAQ job with standard settings.
  EA_Job.SetProperty("NewJob", JOB_EXPORTTYPE, 3) 'Set the Data saving property
after DAQ job: via script.

  numJobs = EA_Job.JobCount 'Get the number of available jobs in DAQ project.
  For i=1 To numJobs
    EA_Job.Start(i) 'Start DAQ job.
    indexJob = EA_Job.CurrentJob 'Get the current job index.

    'Acquire 5 seconds of dummy data.
    T0=Timer
    While Timer-T0 < 5
      DoEvents
    Wend

    EA_Job.Terminate(1) 'Stop the current DAQ job.
    EA_Job.SamplesAcquired(numValues_D, numValues_S, numValues_F) 'Get number of
acquired values with sample rates.
    'Display statistics information of this job.
    EA.MessageBox("Job index: " & CStr(indexJob) & " of total " & CStr(numJobs) & "
jobs" & vbCrLf & _
            "Values(default sample rate): " & CStr(numValues_D) & vbCrLf & _
            "Values(slow sample rate): " & CStr(numValues_S) & vbCrLf & _
            "Values(fast sample rate): " & CStr(numValues_F), "Statistics", MESSAGE_
INFO, BTN_OK)

    ''Save data manually via script.
    If EA_Job.GetProperty(i, JOB_EXPORTTYPE) = 3 Then
      EA_Job.StoreData("Job_" & CStr(indexJob), CATMAN50, "Comment: HBM catman
demo!")
    End If
  Next

  EA_Job.Delete("NewJob") 'Delete the newly inserted DAQ job.
End Sub
```

EA_Job.AddTestParameter

DAQジョブのテストパラメータに、値を含む新規のパラメータを追加します。

対話型モードでの**DAQ jobs**ダイアログボックス、**Job parameters**タブでのテストパラメータテーブルの編集に相当します。

```
  EA_Job.AddTestParameter(ByVal Job As Variant, ByVal ParamName as string, ByVal
ParamVal as String) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
ParamName	新規パラメータの名前。
ParamVal	新規パラメータの値。 複数選択値の区切り文字としてセミコロンを使用できます。これにより、編集可能なテキストの代わりにパラメータ値に対して選択可能なドロップダウンリストを提供します。

戻り値	説明
0	エラーなし。
<0	エラー。DAQジョブが存在しないなど。

EA_Job.SetPropertyの**ジョブ終了後のDAQ設定をテストパラメータに追加**も参照してください。

例

```
Sub CustomizeTestParams()
  EA_Job.AddTestParameter(1, "DrivingTestModel", "High speed loop")
  EA_Job.ModifyTestParameter(1, "Operator", "HBM_TC")
  EA_Job.DeleteTestParameter(1, "Comment")
End Sub

Sub CustomizeListTestParam()
  EA_Job.AddTestParameter(1, "RoadConditions", "dry;wet;icy")
  EA_Job.ModifyTestParameter(1, "RoadConditions", "wet")
End Sub
```

EA_Job.CurrentJob (Read only)

実行中のジョブ番号を返します。

```
Property EA_Job.CurrentJob
```

EA_Job.CurrentJobRepetition (Read only)

反復実行中DAQジョブの現在の反復数を返します。

```
Property EA_Job.CurrentJobRepetition
```

EA_Job.Delete

DAQジョブを削除します。

```
EA_Job.Delete(ByVal Job as Variant) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。

戻り値	説明
0	エラーなし。
<0	エラー。DAQジョブが存在しないなど。

注記

- サンプルはEA_Job.Addを参照してください。

EA_Job.DeleteTestParameter

DAQジョブのテストパラメータを削除します。

- 対話型モードでの**DAQ jobs**ダイアログボックス、**Job parameters**タブでのテストパラメータテーブルの編集に相当します。

```
EA_Job.DeleteTestParameter(ByVal Job As Variant, ByVal Param as Variant) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Param	削除するパラメータの名前または位置。パラメータは1から始まります。

戻り値	説明
0	エラーなし。
<0	エラー。DAQジョブが存在しないなど。

注記

- サンプルはEA_Job.AddTestParameterを参照してください。

EA_Job.GetChannelActivation

チャネルのアクティブ状態を返します。

```
EA_Job.GetChannelActivation(ByVal Job As Variant, ByVal Channel As Variant) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。

戻り値	説明
0	チャネルは非アクティブです。
1	チャネルはアクティブ(このDAQジョブでの計測に加わっています)。
<0	エラー。チャネルまたはDAQジョブが存在しないなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。
- アクティブステータスは、各DAQジョブ毎に管理されています。

例:

```
Sub CheckChannel()
  Dim actStatus As Long
  actStatus = EA_Job.GetChannelActivation(1, "Pressure")

  'Change the channel status to Active if it was Inactive.
  If actStatus = 0 Then
    EA_Job.SetChannelActivation(1, "Pressure", 1)
  End If
End Sub
```

EA_Job.GetDigitalTriggerSettings

デジタルトリガの設定を読み込みます。

```
  EA_Job.GetDigitalTriggerSettings(ByVal idxJob As Integer, ByVal TrigType As
Integer, IOChanIndex As Integer, Logic As Integer, Bit As Integer, Pin As Integer,
DevType As Integer, SubDevType As Integer) As Long
```

引数	説明
idxJob	DAQジョブの位置。DAQジョブは1から始まります。
TrigType	トリガタイプ(0 = 開始トリガ、それ以外の場合は停止トリガ)。
IOChanIndex	トリガとして使用するデジタルI/Oチャネルのインデックス。 CX22モジュールの特別デジタルIO: デジタル入力1 = -1003、論理 = 1 デジタル入力2 = -1002、論理 = 1 デジタル入力3 = -1001、論理 = 1
Logic	論理レベル(0 = LOW、1 = HIGH)
Bit	トリガビット。
Pin	ピン番号。
DevType	デバイスタイプ。
SubDevType	サブデバイスタイプ。

EA_Job.GetExportMode

チャネルが計測データファイルに保存しているかどうか返します。

```
  EA_Job.GetExportMode(ByVal Job As Variant, ByVal Channel As Variant, Mode As
Integer) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
Mode	チャネルの保存状態: 0 = チャンネルは保存されていません 1 = チャンネルは保存されています

戻り値	説明
0	エラーなし。
<0	エラー。チャネルまたはDAQジョブが存在しないなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

- 保存ステータスは、各DAQジョブ毎に管理されています。

EA_Job.GetFilter

指定したDAQジョブに割り当てられたデジタルローパスフィルタを読み取ります。

対話型モードでの**DAQ jobs**ダイアログボックス(Registerタブの"Channel specific")にあるチャネルフィルタリストの編集に相当します。対話型モードでは、"Allow manual filter settings"が有効な場合("File/Options/DAQ Channel")のみ、設定が可能です。

```
EA_Job.GetFilter(ByVal Job As Variant, ByVal Channel As Variant, FilterType As
Integer, FilterFreq As Integer) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Channel	チャネル名または位置。
FilterType	フィルタタイプ: 0 = 現在のハードウェア設定をそのまま使用 1 = 自動 141 = バターワース 142 = ベッセル 144 = 高精細バターワース(HD) 145 = 高精細ベッセル(HD)
FilterFreq	フィルタ周波数:

```
   FREQ_POINT001HZ = 901          FREQ_40HZ_ALL = 948
   FREQ_POINT002HZ = 904          FREQ_80HZ_BU = 953
   FREQ_POINT004HZ = 907          FREQ_100HZ_BE = 955
   FREQ_POINT005HZ_BE = 908       FREQ_200HZ_BE = 958
   FREQ_POINT01HZ_BE = 914        FREQ_250HZ_BU = 959
   FREQ_POINT02HZ_BE = 917        FREQ_400HZ_BE = 961
   FREQ_POINT04HZ = 920           FREQ_500HZ_BU = 962
   FREQ_POINT05HZ_BE = 921        FREQ_800HZ = 967
   FREQ_POINT08HZ = 925           FREQ_900HZ_BE = 968
   FREQ_1HZ = 927                 FREQ_1000HZ_BU = 969
   FREQ_1POINT25HZ_BE = 929       FREQ_1500HZ_BU = 972
   FREQ_2HZ = 931                 FREQ_2000HZ_BU = 973
   FREQ_2POINT5HZ_BE = 932        FREQ_4000HZ = 977
   FREQ_4HZ = 934                 FREQ_5000HZ_BU = 978
   FREQ_5HZ_ALL = 935             FREQ_8000HZ = 982
   FREQ_8HZ = 939                 FREQ_10000HZ_BU = 1100
   FREQ_10HZ_ALL = 941            FREQ_20000HZ = 986
   FREQ_20HZ_ALL = 945
```

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。
- チャネルの初期化後にフィルタ設定を確認する必要があります。
- すべての周波数が指定したハードウェアで実行できるわけでなく、EA_Job.SetFilterへの呼び出しであることに注意してください。要求された周波数は、おそらくハードウェアでサポートされている次に低い周波数に調整されます。*GetFilter*を使用して設定の結果を確認してください。

例:

```
  Sub SetFilters()
    Sub FType As Integer
    Sub FFreq As Integer
    EA_Job.SetFilter 1, "Pressure", 141, 945 'Select Butterworth filter with a
frequency of 20 Hz.
    EA_IO.InitializeChannel "Pressure", 0 'ReInitialize the channel.
    EA_Job.GetFilter 1, "Pressure", FType, FFreq 'FType=141, FFreq=945.
  End Sub
```

EA_Job.GetJournalActivation

DAQジョブの統計ジャーナルでチャネルがアクティブ化されているかどうかを返します。

```
  EA_Job.GetJournalActivation(ByVal Job As Variant, ByVal Channel As Variant, Mode
as Integer) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
Mode	チャネルの統計ジャーナルの状態: 0 = チャンネルはアクティブ化されていません 1 = チャンネルはアクティブ化されています

戻り値	説明
0	エラーなし。
<0	エラー。チャネルまたはDAQジョブが存在しないなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。
- ジャーナルのステータスは、各DAQジョブ毎に管理されています。

例:

```
Sub CheckJournalActivation()
  Dim ActiveMode As Integer
  EA_Job.GetJournalActivation(1, "Pressure", ActiveMode)

  'Change the channel journal activation mode to JOURNAL_ON if it was JOURNAL_OFF.
  If ActiveMode = 0 Then
    EA_Job.SetJournalActivation(1, "Pressure", 1)
  End If
End Sub
```

EA_Job.GetProperty

DAQジョブのパラメータ値を返します。

対話型モードでの**DAQ jobs**ダイアログボックスにある**General**および**Job parameters**タブに相当します。

```
EA_Job.GetProperty(ByVal Job As Variant, ByVal PropIndex As Integer) As Variant
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。 現在実行中のDAQジョブからプロパティを取得するには、0を指定します。Jobを指定していない場合、1を想定します。
PropIndex	DAQジョブのパラメータのインデックス: 1 = DAQジョブ名 2 = DAQジョブのアクティブ化 (0 = アクティブ化、1 = 非アクティブ化) 3 = DAQジョブの繰り返し数(-1 = 無制限の繰り返し) 4 = DAQジョブの繰り返し間の遅延 (秒) 5 = DAQ ジョブ開始を開始する開始パネル(名前または位置) 6 = デフォルトのサンプルレート(Hz) 7 = 低速サンプルレート(Hz) 8 = 開始モード(0 = 手動、1 = トリガ、2 = 時点) 9 = 停止モード(0 = 手動、1 = トリガ、2 = 時点、3 = サンプル数、4 = 予約、5 = 曜日、6 = 時刻) 10 = 開始トリガモード(0 = レベル以上、1 = レベル以下、2 = 立ち上がりエッジ、3 = 立ち下がりエッジ、4 = デジタル入力、5 = イベント監視またはEasyScript) 11 = 開始トリガレベル(レベルトリガおよびエッジトリガの場合) 12 = 開始トリガチャネル(名前) 13 = 開始トリガビット(デジタル入力によるトリガの場合は0 - 15) 14 = 開始時刻(開始モードが時点の場合) 15 = 停止トリガモード(0 = レベル以上、1 = レベル以下、2 = 立ち上がりエッジ、3 = 立ち下がりエッジ、4 = デジタル入力、5 = イベント監視またはEasyScript) 16 = 停止トリガレベル(レベルトリガおよびエッジトリガの場合) 17 = 停止トリガチャネル(名前) 18 = 停止トリガビット(デジタル入力によるトリガの場合は0 - 15) 19 = 秒単位の計測期間(一定期間にわたる測定の場合) 20 = プリトリガ時間(秒) 21 = ポストトリガ時間 (秒) 22 = 計測を開始する前にゼロバランスを実行(0 = ハードウェアチャネルをゼロにしない、1 = ハードウェアチャネルをゼロに設定) 23 = 制限監視を非アクティブ化(0 = アクティブ化、1 = 非アクティブ化) 24 = データストレージ モード(0 = すべてのデータを保持、1 = 手動制御、2 = オートシーケンス/スクリプトによる制御、3 = ピーク値、4 = サイクル制御の間隔ストレージ、5 = タイマ制御の間隔ストレージ、6 = 高速ストリーム) 25 = DAQジョブ後のデータ保存(エクスポート) (0 = DAQ終了時にプロンプトを表示、1 = DAQ終了時に自動的に実行、2 = 計測中に定期的に実行、3 = なし、テストモードはそれぞれスクリプトコマンドで制御) Fast-Streamストレージ モードでは、保存オプション1(DAQ ジョブ終了時に自動的に)と2(ジョブ中に定期的に)のみをサポートしています。 26 = エクスポートファイル名。パスが指定されていない場合は、ファイル名のみが返されます。

	27 = エクスポート形式(1 = catman、2 = ASCII、3 = ASCII + チャネル情報、4 = Excel(Office XP/2003)、5 = Diadem(.dat)、6 = nSoft DAC、7 = Vector MDF 3、8 = Excel(Office 2007 XLSX XML)、9 = Excel (Office 2007 XLSB バイナリ)、10 = Matlab 5.0、11 = RPC III、12 = nCode s3t、13 = ASAM MDF 4.1、14 = UFF58、15 = NI TDMS) 28 = エクスポートファイルのコメント 29 = 定期的なデータストレージのファイルモード (0 = 単一の増加ファイル、1 = 間隔ごとに1つのファイル) 30 = ジョブ終了後のDAQ設定をテストパラメータに追加(0 = 追加しない、1 = 追加) 31 = DAQジョブをPCカードジョブとしてマーク 32 = PCカードジョブのファイル名(ファイル拡張子として.MEAのみを許可) 33 = PCカードジョブのファイル形式(0 = 4 バイト(INT)、1 = 2バイト(INT)、2 = 4バイト(FLOAT)) 34 = 圧縮ファイルの圧縮率(0 = 圧縮なし、可能な圧縮率: 500、1000、2000、5000、10000) 35 = PCカードジョブファイルのファイルコメント 36 = 各参加デバイスで一意のPC カードジョブファイル名を自動生成 37 = 計測開始時のMGCplus内部クロックをPCクロックへ調整 38 = 複数のMGCplusデバイスのマスタ/スレーブモードでの動作 39 = *DAQジョブ後のデータ保存*に**計測中に定期的に実行**が使用されている場合のデータ保存間隔(秒単位)。パラメータ25も参照してください。 40 = *停止モード*に**サンプル数**を使用した場合に取得する値(サンプル)数。パラメータ9を参照。 41 = 一時データストレージのストレージ深さ(秒単位) (0 = 時間制限のないすべての値) 42 = 高速サンプルレート(Hz) 43 = NTPサーバアドレス(xxx.xxx.xxx.xxx) 44 = NTP同期を使用(1 = 使用する、0 = 使用しない) 45 = NTPマスタタイムスタンプチャネル "|"で区切られた3つまでのチャネル(標準、低速、高速)のリスト 46 = ピークストレージモードの時間間隔(秒単位)。パラメータ24を参照。 47 = 保存(エクスポート)データの数値精度(0 = 8バイト倍精度、1 = 4バイト単精度、2 = 2バイト整数) 48 = DAQジョブ終了後に定期保存(パラメータ25を参照)で作成されたファイルを自動マージ(0 = マージしない、1 = マージする) 49 = DAQジョブ終了後にデジタル表示器とテーブルに統計を表示(0 = 表示しない、1 = 表示) 50 = DAQジョブ中に接続が失われた場合のデバイスの非アクティブ化(0 = タイムアウトエラーでジョブを終了、1 = このデバイスのすべてのチャネルを非アクティブにしてジョブを続行、2 = 完全な初期化でジョブを再起動、3 = タイムアウト エラーなしでジョブを終了) 51 = eDAQリモートコントロールをアクティブ化 = 廃止 52 = eDAQリモートコントロールビット = 廃止 53 = SY42同期を使用(MGCplusのみ) (0 = SY42同期なし、1 = SY42同期あり) 54 = ASCIIエクスポートの小数点以下の桁数(0 = 自動小数点以下の桁数、1 = 完全精度、2 - 7 = 小数点以下1 - 6桁) 55 = 非同期読み取りモードを使用(0 = 非同期読み取りなし、1 = 非同期読み取り) 56 = PCタイミングを使用(0 = PCタイミングを使用しない、1 = PCタイミングを使用) 57 = 演算チャネルの自動ゼロ化(0 = 演算チャネルをゼロ化しない、1 = 演算チャネルをゼロ化) 58 = 統計ジャーナルファイルを作成(0 = ジャーナルを作成しない、1 = ジャーナルを作成) 59 = ジャーナルファイルの更新間隔(秒単位) 60 = トリガのバーストモード設定(0 = バースト モードなし、1 = バーストモード) 61 = 最大バースト数 62 = DAQジョブ開始時にQuantumXデバイス上の他のクライアントプロセスを閉じる(0 = 閉じない、1 = 閉じる) 63 = MGCとマスタデバイスの開始の開始遅延(秒単位) 64 = NTP同期補正(ミリ秒)

	65 = DAQジョブ開始時にすべてのデバイスを再接続(0 = デバイスを再接続しない、1 = デバイスを再接続する) 66 = DAQ停止時にテストパラメータ編集ダイアログを表示(DAQ停止後の値の自動保存のみ)(0 = 表示しない、1 = 表示する) 67 = 統計ジャーナルのバックアップ間隔(0 = バックアップなし、1 = 毎日午前12時、2 = 毎日午後12時、3 = 毎週午前12時、4 = 毎週午後12時、5 = 毎月午前12時、6 = 毎月午後12 時) 68 = DAQ 状態"トリガ待ち"中のジャーナル更新(0 = "トリガ待ち"中にジャーナル更新しない、1 = "トリガ待ち"中に更新する) 69 = ジャーナルに書き込まれる値を指定するビットマスク(1 = スナップショット、2 = 最小、4 = 最大、8 = 平均)。 70 = ポストトリガフェーズ中の再トリガの最大数 71 = レベル開始トリガ: DAQの保存を開始するために、トリガチャネル値がトリガ値を上回るか下回る必要がある最小時間(秒)。 74 = DAQ開始時にイベントカウントをリセット(0 = カウンタをリセットしない、1 = カウンタをリセットする) 75 = ジョブ停止後にイベントファイルを作成(0 = ファイルを作成しない、1 = ファイルを作成する)。 79 = FTP/SFTPサーバへのローカルエクスポート後にファイルをアップロード(0 = ファイルをアップロードしない、1 = ファイルをアップロードする) 80 = FTP/SFTPサーバ名。この情報は、プロジェクトのすべてのジョブとレコーダで同じです。 81 = FTP/SFTPサーバ上のアカウントのユーザ名。この情報は、プロジェクトのすべてのジョブとレコーダで同じです。 82 = 非適用(安全上の理由から、FTP/SFTPアカウントのパスワードを取得することは許可されていません) 83 = ファイルを保存するFTP/SFTPサーバ上の作業ディレクトリ 84 = FTP/SFTP転送後にローカルファイルシステムからソースファイルを削除 85 = ビデオファイルをFTP/SFTPサーバにアップロード(0 = ビデオファイルをアップロードしない、1 = ビデオファイルをアップロードする) 86 = FTPデータ接続の確立方法を決定(0 = アクティブモード、1 = パッシブモード)。この情報は、プロジェクトのすべてのジョブとレコーダで同じです。 87 = SFTP認証用のPEM秘密キー。この情報は、プロジェクトのすべてのジョブとレコーダで同じです。 88 = 非適用(安全上の理由から、SFTP秘密キーのパスフレーズを取得することは許可されていません) 89 = SFTP認証方法のタイプ(0 = パスワード認証、1 = 公開キー認証)。この情報は、プロジェクトのすべてのジョブとレコーダで同じです。 90 = FTP/SFTP接続に使用されるプロキシサーバのタイプ。この情報は、プロジェクトのすべてのジョブとレコーダで同じです。 91 = リクエストがルーティングされるプロキシサーバの名前またはアドレス。この情報は、プロジェクトのすべてのジョブとレコーダーで同じです。 92 = リクエストがルーティングされるプロキシサーバによって使用されるポート。この情報は、プロジェクトのすべてのジョブとレコーダーで同じです。 93 = 認証を必要とするプロキシサーバ経由のすべてのリクエストで送信されるユーザ名。この情報は、プロジェクトのすべてのジョブとレコーダーで同じです。 94 = 非適用(安全上の理由から、プロキシサーバのパスワードを取得することは許可されていません) 95 = レコーダのダウンサンプリングローパスフィルタを非アクティブ化(0 = ダウンサンプリングローパスフィルタをアクティブ、1 = ダウンサンプリングローパスフィルタを非アクティブ) 96 = 停止トリガの最大待機時間(秒単位) 97 = レコーダのデータ記録の停止時間(0 = 毎日24:00、1 = 毎日18:00、2 = 毎日12:00、3 = 毎日06:00、4 = 12時間ごと24:00:00と12:00、5 = 12時間ごとの06:00と18:00、6 = 6時間ごとの24:00、06:00、12:00、18:00、7 = 3時間ごとの24:00:00、03:00、06:00、09:00、12:00、15:00、18:00、21:00、8 = 2時間ごとの24:00、02:00、04:00.. .、9 = 毎時24:00、01:00、02:00...) 98 = 2つのデータ転送間の時間(ミリ秒)。

	99 = データ転送のタイムアウト(ミリ秒)。 100 = ビデオカメラ1で記録されたビデオファイル名 101 = ビデオカメラ2で記録されたビデオファイル名 102 = ビデオカメラ3で記録されたビデオファイル名 103 = ビデオカメラ4で記録されたビデオファイル名 104 = ビデオカメラ5で記録されたビデオファイル名 105 = ビデオカメラ6で記録されたビデオファイル名 106 = ビデオカメラ7で記録されたビデオファイル名 107 = ビデオ カメラ8で記録されたビデオファイル名 200 = ビデオカメラ1のビデオ録画の開始条件 201 = ビデオカメラ2のビデオ録画の開始条件 202 = ビデオカメラ3のビデオ録画の開始条件 203 = ビデオカメラ4のビデオ録画の開始条件 204 = ビデオカメラ5のビデオ録画の開始条件 205 = ビデオカメラ6のビデオ録画の開始条件 206 = ビデオカメラ7のビデオ録画の開始条件 207 = ビデオカメラ8のビデオ録画の開始条件 0: DAQ ジョブの開始とともに自動実行 1: 開始トリガにより自動 2: 制限値とイベント監視を手動で使用するか、スクリプトまたはAutoSequenceを使用 300 = ビデオカメラ1のビデオ録画の停止条件 301 = ビデオカメラ2のビデオ録画の停止条件 302 = ビデオカメラ3のビデオ録画の停止条件 303 = ビデオカメラ4のビデオ録画の停止条件 304 = ビデオカメラ5のビデオ録画の停止条件 305 = ビデオカメラ6のビデオ録画の停止条件 306 = ビデオカメラ7のビデオ録画の停止条件 307 = ビデオカメラ8のビデオ録画の停止条件 0: DAQ ジョブの停止とともに自動実行 1: 停止トリガにより自動 2: 制限値とイベント監視を手動で使用するか、スクリプトまたはAutoSequenceを使用 400 = DAQジョブの開始またはレコーダの記録開始時にスクリプトコードを実行(0 = コードを実行しない、1 = コードを実行する) 401 = DAQジョブの停止またはレコーダの記録開始時にスクリプトコードを実行(0 = コードを実行しない、1 = コードを実行する) 402 = DAQジョブの開始時またはレコーダの記録開始時に実行するスクリプトコード 403 = DAQジョブの停止時またはレコーダの記録開始時に実行するスクリプトコード 404 = トリガによるジョブ開始の場合、DAQ停止後に一時データ ストレージでcatman時間チャネルを正規化(0 = 時間チャネルを正規化しない、1 = 時間チャネルを0にリセット) 405 = 開始トリガのデジタル入力のデバウンス時間(ミリ秒) 406 = 停止トリガのデジタル入力のデバウンス時間(ミリ秒) 407 = MGCplusの記録データをNFS共有に出力 408 = MGCplusの記録データ出力のNFS URL 409 = "曜日"の停止条件。日曜日(0)から土曜日(6)までのビットマスク。 例: 月曜日 + 水曜日 + 金曜日 = 2^1 + 2^3 + 2^5 = 42。毎日 = 127。 410 = "曜日"の停止条件。停止する時刻。フルタイムのみ。0 = 00:00、1 = 01:00、14 = 14:00... 411 = "時刻"の停止条件。毎日。1 = 00:00 から1時間ごと、2 = 00:00 から2時間ごと、3 = 00:00から3時間ごと、4 = 00:00から6時間ごと、5 = 00:00、12 :00、6 = 06:00、18:00。

戻り値	説明
-14	DAQジョブが存在しません。
-32767	エラー。

EA_Job.SetProperty、EA_Job.GetChannelActivation、EA_Job.AddTestParameter、EA_Job.ModifyTestParameter、EA_Job.DeleteTestParameterも参照してください。

注記

- EasyScript Code Builderより、例えば`Const JOB_DEFAULTSAMPLERATE = 6`のように、意味のある名前を持つDAQジョブの引数のインデックスを定数リストに挿入することができます。これらの関数はCode Builderの"Constant declarations"グループにあります。

- サンプルはEA_Job.Addを参照してください。

EA_Job.GetStatusInfo

実行中のDAQジョブに関する情報を取得します。これは、ステータスバーまたは拡張DAQ情報ウィンドウに表示される情報と似ています。

```
EA_Job.GetStatusInfo(InfoList() As Variant) As Long
```

引数	説明
InfoList	InfoList配列には、さまざまなDAQジョブステータスが含まれます。配列の要素数は32より大きくなければなりません。 InfoList(0): DAQジョブステータス 0 = DAQジョブは実行していません 1 = DAQジョブが実行中でデータを取得中 2 = DAQジョブが実行中であり、開始トリガ待ち 3 = DAQジョブが実行中で、停止トリガ待ち InfoList(1): 利用可能なストレージ容量に応じた残りの記録時間(秒) InfoList(2): デフォルトのサンプルレートで測定されたサンプルの数 InfoList(3): 低速サンプルレートで計測されたサンプル数 InfoList(4): 高速サンプルレートで計測されたサンプル数 InfoList(5): デフォルトサンプルレート(Hz) (使用しない場合は0) InfoList(6): 低速サンプルレート (Hz)(使用されていない場合は0) InfoList(7): 高速サンプルレート(Hz) (使用しない場合は0) InfoList(8): 秒単位のリアルタイムラグ InfoList(9): DAQプロセスの作業負荷(%) InfoList(10): カメラ1のステータス InfoList(11): カメラ2のステータス InfoList(12): カメラ3のステータス InfoList(13): カメラ4のステータス InfoList(14): カメラ5のステータス InfoList(15): カメラ6のステータス InfoList(16): カメラ7のステータス InfoList(17): カメラ8のステータス InfoList(18): ステータス バーに表示される実行中のDAQジョブのステータステキスト 他の配列の要素は将来使用するために予約されています。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルまたはDAQジョブが存在しないなど。

注記

- ジョブに関する詳細情報は、EA_Job.GetPropertyから取得できます。

EA_Job.GetStorageMode

チャネルが一時データストレージに格納されるかどうかを返します。

```
  EA_Job.GetStorageMode(ByVal Job As Variant, ByVal Channel As Variant, Mode as
Integer) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Channel	チャネル名または位置。DAQプロジェクトのチャネルは、1から始まります。
Mode	チャネルのストレージ状態: 0 = チャネルは保存されません 1 = チャネルは保存されます

戻り値	説明
0	エラーなし。
<0	エラー。チャネルまたはDAQジョブが存在しないなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

- ストレージ状態は、各DAQジョブ毎に管理されています。

例:

```
  Sub CheckChannel()
    Dim saveMode As Integer
    EA_Job.GetStorageMode(1, "Pressure", saveMode)

    'Change the channel storage mode to STORAGE_ON if it was STORAGE_OFF.
    If saveMode = 0 Then
      EA_Job.SetStorageMode(1, "Pressure", 1)
    End If
  End Sub
```

EA_Job.GetTestParameter

インデックスより識別されたテストパラメータの名前および値を取得します。

```
  EA_Job.GetTestParameter(ByVal Job As Variant, ByVal Index As Integer, Name As
String, Value As String) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Index	パラメータの位置。パラメータは1から始まります。
Name	パラメータ名。
Value	パラメータ値。

戻り値	説明
0	エラーなし。
<0	エラー。DAQジョブが存在しないなど。

注記

- 定義されているパラメータの合計数を取得するにはEA_Job.GetTestParameterCountを参照してください。

例

```
Sub IterateTestParams()
  Dim i As Integer, countParam As Integer
  Dim nameParam As String, valueParam As String

  EA_Job.GetTestParameterCount(1, countParam)

  For i=1 To countParam
    EA_Job.GetTestParameter(1, i, nameParam, valueParam)
    MsgBox nameParam & " = " & valueParam
  Next
End Sub
```

EA_Job.GetTestParameterCount

テストパラメータ数を取得します。

```
EA_Job.GetTestParameterCount(ByVal Job As Variant, Count As Integer) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Count	パラメータ数。

戻り値	説明
0	エラーなし。
<0	エラー。DAQジョブが存在しないなど。

例

```
Sub IterateTestParams()
  Dim i As Integer, countParam As Integer
  Dim nameParam As String, valueParam As String

  EA_Job.GetTestParameterCount(1, countParam)

  For i=1 To countParam
    EA_Job.GetTestParameter(1, i, nameParam, valueParam)
    MsgBox nameParam & " = " & valueParam
  Next
End Sub
```

EA_Job.GlobalTestCounter

計測データのファイル名に使用されるグローバルプレースホルダカウンタの現在値を設定または読み取ります。

```
Property EA_Job.GlobalTestCounter
```

注記

- プレースホルダー"%00%"と"%000%"は実際にはサポートされています。カウンタは、データ取得後にデータエクスポートが発生するたびに増加します。スクリプト関数*EA_Job.GlobalTestCounter*を使用してカウンタを希望の数値に設定するか、対話モードで DAQ ジョブ設定ダイアログの**Reset counter**ボタンを使用してカウンタを0に設定します(このボタンはプレースホルダー"%00%"または計測データのファイル名に"%000%"が使用されている場合にのみ利用できます)。

EA_Job.JobCount (Read only)

DAQ プロジェクトで使用可能なDAQジョブとレコーダの合計数を返します。

```
Property EA_Job.JobCount
```

EA_Job.ModifyTestParameter

DAQジョブ内のテストパラメータの値を変更します。

対話型モードでの**DAQ jobs**ダイアログボックス、**Job parameters**タブでのテストパラメータテーブルの編集に相当します。

```
  EA_Job.ModifyTestParameter(ByVal Job As Variant, ByVal Param as Variant, ByVal
ParamVal as String) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Param	パラメータ名または位置。パラメータは1から始まります。
ParamVal	パラメータの新規の値。

戻り値	説明
0	エラーなし。
<0	エラー。DAQジョブが存在しないなど。

注記

- サンプルはEA_Job.AddTestParameterを参照してください。

EA_Job.RTLag (Read only)

リアルタイムのタイムラグを秒単位で返します。

対話型モードでは、計測中にステータスバーに表示されるRT-Lag表示(データ収集と表示間の時間遅延)に相当します。

```
Property EA_Job.RTLag
```

EA_Job.SamplesAcquired

デフォルト、低速および高速サンプリングレートで取得した、チャネルの計測値数を返します。

```
  EA_Job.SamplesAcquired(Standard As Long, Slow As Long, Optional Fast as Long) As
Long
```

引数	説明
Standard	デフォルトサンプリングレートのチャネルで計測した数。
Slow	低速サンプリングレートのチャネルで計測した数。
Fast	高速サンプリングレートのチャネルで計測した数。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Job.SetChannelActivation

チャネルのアクティブ状態を設定します。

対話型モードでの**DAQ jobs**ダイアログボックスにあるchannel activation tableの編集に相当します。

```
  EA_Job.SetChannelActivation(ByVal Job As Variant, ByVal Channel As Variant, ByVal
State As Integer) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Channel	チャネル名または位置。
State	チャネルのアクティブ状態: 0 = Inactive (DAQの計測に参加しません) 1 = Active (DAQジョブの計測に参加します)

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

例:

```
Sub CheckChannel()
  Dim actStatus As Long
  actStatus = EA_Job.GetChannelActivation(1, "Pressure")

  'Change the channel status to Active if it was Inactive.
  If actStatus = 0 Then
    EA_Job.SetChannelActivation(1, "Pressure", 1)
  End If
End Sub
```

EA_Job.SetExportMode

DAQ ジョブ中にチャネルを計測データファイルに保存するか設定します。

対話型モードでの**DAQ jobs**ダイアログボックスの**Channel parameters**の編集に相当します。

```
EA_Job.SetExportMode(ByVal Job As Variant, ByVal Channel As Variant, ByVal Mode As Integer) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Channel	チャネル名または位置。
Mode	チャネルの保存状態: 0 = No save (このDAQジョブでは、計測データファイルに保存しません) 1 = Save (このDAQジョブで、計測データファイルに保存します)

戻り値	説明
0	エラーなし。
<0	エラー。チャネルまたはDAQジョブが存在しないなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

- このプロパティは、ジョブの開始前に設定する必要があることに注意してください! ジョブの実行中は変更できません。

EA_Job.SetFilter

デジタルローパスフィルタを割り当てます。この設定は、チャネルの初期化時に有効となります。

対話型モードでの**DAQ jobs**ダイアログボックス(Registerタブの"Channel specific")にあるチャネルフィルタリストの編集に相当します。対話型モードでは、"Allow manual filter settings"が有効な場合("File/Options/DAQ Channel")のみ、設定が可能です。

```
  EA_Job.SetFilter(ByVal Job As Variant, ByVal Channel As Variant, ByVal FilterType
As Integer, ByVal FilterFreq As Integer) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Channel	チャネル名または位置。
FilterType	フィルタタイプ: 0 = 現在のハードウェア設定を使用 1 = 自動 141 = バターワース 142 = ベッセル 144 = 高精細バターワース(HD) 145 = 高精細ベッセル(HD)
FilterFreq	フィルタ周波数: FREQ_POINT001HZ = 901 FREQ_POINT002HZ = 904 FREQ_POINT004HZ = 907 FREQ_POINT005HZ_BE = 908 FREQ_POINT01HZ_BE = 914 FREQ_POINT02HZ_BE = 917 FREQ_POINT04HZ = 920 FREQ_POINT05HZ_BE = 921 FREQ_POINT08HZ = 925 FREQ_1HZ = 927 FREQ_1POINT25HZ_BE = 929 FREQ_2HZ = 931 FREQ_2POINT5HZ_BE = 932 FREQ_4HZ = 934 FREQ_5HZ_ALL = 935 FREQ_8HZ = 939 FREQ_10HZ_ALL = 941 FREQ_20HZ_ALL = 945 FREQ_40HZ_ALL = 948 FREQ_80HZ_BU = 953 FREQ_100HZ_BE = 955 FREQ_200HZ_BE = 958 FREQ_250HZ_BU = 959 FREQ_400HZ_BE = 961 FREQ_500HZ_BU = 962 FREQ_800HZ = 967 FREQ_900HZ_BE = 968 FREQ_1000HZ_BU = 969 FREQ_1500HZ_BU = 972 FREQ_2000HZ_BU = 973 FREQ_4000HZ = 977 FREQ_5000HZ_BU = 978 FREQ_8000HZ = 982 FREQ_10000HZ_BU = 1100 FREQ_20000HZ = 986

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。
- すべての周波数がすべてのフィルタタイプに適用できるわけではないことに注意してください。このため周波数定数は、ButterworthのBUまたは、BesselのBEを選択します。
- すべての周波数が指定したハードウェアで実行できるわけではないことに注意してください。要求された周波数は、おそらくハードウェアでサポートされている次に低い周波数に調整されます。チャネルの初期化後に、EA_Job.GetFilterを使用して実際のフィルタ設定を確認しすることができます。

例:

```
  Sub SetFilters()
    Sub FType As Integer
    Sub FFreq As Integer
    EA_Job.SetFilter 1, "Pressure", 141, 945 'Select Butterworth filter with a
frequency of 20 Hz.
    EA_IO.InitializeChannel "Pressure", 0 'ReInitialize the channel.
    EA_Job.GetFilter 1, "Pressure", FType, FFreq 'FType=141, FFreq=945.
  End Sub
```

EA_Job.SetJournalActivation

DAQジョブ中のチャネルの統計ジャーナルをアクティブに設定します。

対話型モードでの**DAQ jobs**ダイアログボックスの**Channel parameters**の編集に相当します。

```
EA_Job.SetJournalActivation(ByVal Job As Variant, ByVal Channel As Variant, ByVal
Mode As Integer) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Channel	チャネル名または位置。
Mode	チャネルの統計ジャーナルの状態: 0 = No activation (DAQジョブの統計ジャーナルに格納されません) 1 = Activation(DAQジョブの統計ジャーナルに格納されます)

戻り値	説明
0	エラーなし。
<0	エラー。チャネルまたはDAQジョブが存在しないなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。
- このプロパティは、ジョブの開始前に設定する必要があることに注意してください！ ジョブの実行中は変更できません。

例:

```
Sub CheckJournalActivation()
  Dim ActiveMode As Integer
  EA_Job.GetJournalActivation(1, "Pressure", ActiveMode)

  'Change the channel journal activation mode to JOURNAL_ON if it was JOURNAL_OFF.
  If ActiveMode = 0 Then
    EA_Job.SetJournalActivation(1, "Pressure", 1)
  End If
End Sub
```

EA_Job.SetProperty

DAQジョブのパラメータ値を設定します。

対話型モードでの**DAQ jobs**ダイアログボックスにある**General**および**Job parameters**タブに相当します。

```
  EA_Job.SetProperty(ByVal Job As Variant, ByVal PropIndex As Integer, ByVal PropVal
As Variant) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
PropIndex	DAQジョブのパラメータのインデックス: 1 = DAQジョブ名 2 = DAQジョブのアクティブ化 (*PropVal*: True/False) 3 = DAQジョブの繰り返し数(-1 = 無制限の繰り返し) 4 = DAQジョブの繰り返し間の遅延 (秒) 5 = DAQ ジョブ開始を開始する開始パネル(名前または位置)。*PropVal* = 0(デフォルト)は、現在表示されているパネルをアクティブのままにします。 6 = デフォルトのサンプルレート(Hz) • MGCplus: 1, 2, 5, 10, 20, 50, 75, 100, 200, 300, 600, 1200, 2400, 4800, 9600(シングルチャネルボードのみ), 19200 (シングルチャネルボードのみ) • QuantumX: 1, 2, 5, 10, 20, 50, 100, 300, 600, 1200, 2400, 4800, 9600, 19200, 96000 (MX410, MX411, MX403, MX460のみ), 192000 (MX410 2チャネル, MX411,MX403のみ) • 光インテロゲータ: 任意 < 単独で動作する場合はインタロゲータタイプの最大レート、MGCplusと併用して動作する場合はMGCplusレートに制限される 7 = 低速サンプルレート(Hz) • MGCplus: 1, 2, 5, 10, 20, 50, 75, 100, 200, 300, 600, 1200, 2400, 4800, 9600(シングルチャネルボードのみ), 19200 (シングルチャネルボードのみ) • QuantumX: 1, 2, 5, 10, 20, 50, 100, 300, 600, 1200, 2400, 4800, 9600, 19200, 96000 (MX410, MX411, MX403, MX460のみ), 192000 (MX410 2チャネル, MX411, MX403のみ) • 光インテロゲータ: 任意 < 単独で動作する場合はインタロゲータタイプの最大レート、MGCplusと併用して動作する場合はMGCplusレートに制限される 1Hz未満の任意の数値(例: 0.1Hz)を指定することもできます。この場合、ハードウェア組み込まれているサンプルレートジェネレータは使用しません。代わりに、catmanはPCタイマに基づいて単一ポイント値を収集します。 バージョン3.0以降に導入された*高速サンプルレート*については、プロパティインデックス42を参照してください。 8 = 開始モード(0 = 手動、1 = トリガ、2 = 時点) 9 = 停止モード(0 = 手動、1 = トリガ、2 = 時点、3 = サンプル数、4 = 予約、5 = 曜日、6 = 時刻) 10 = 開始トリガモード(0 = レベル以上、1 = レベル以下、2 = 立ち上がりエッジ、3 = 立ち下がりエッジ、4 = デジタル入力、5 = イベント監視またはEasyScript) 11 = 開始トリガレベル(レベルトリガおよびエッジトリガの場合) 12 = 開始トリガチャネル(名前) 13 = 開始トリガビット(デジタル入力によるトリガの場合は0 - 15) 14 = 開始時刻(開始モードが時点の場合) 15 = 停止トリガモード(0 = レベル以上、1 = レベル以下、2 = 立ち上がりエッジ、3 = 立ち下がりエッジ、4 = デジタル入力、5 = イベント監視またはEasyScript) 16 = 停止トリガレベル(レベルトリガおよびエッジトリガの場合) 17 = 停止トリガチャネル(名前) 18 = 停止トリガビット(デジタル入力によるトリガの場合は0 - 15)

	19 = 秒単位の計測期間(一定期間にわたる測定の場合) 20 = プリトリガ時間(秒) 21 = ポストトリガ時間 (秒) 22 = 計測を開始する前にゼロバランスを実行(PropVal: True/False) 23 = 制限監視を非アクティブ化 (PropVal: True/False) 24 = データストレージ モード(0 = すべてのデータを保持、1 = 手動制御、2 = オートシーケンス/スクリプトによる制御、3 = ピーク値、4 = サイクル制御の間隔ストレージ、5 = タイマ制御の間隔ストレージ、6 = 高速ストリーム) 25 = DAQジョブ後のデータ保存(エクスポート) (0 = DAQ終了時にプロンプトを表示、1 = DAQ終了時に自動的に実行、2 = 計測中に定期的に実行、3 = なし、テストモードはそれぞれスクリプトコマンドで制御) Fast-Streamストレージ モードでは、保存オプション1(DAQ ジョブ終了時に自動的に)と2(ジョブ中に定期的に)のみをサポートしています。 26 = エクスポートファイル名。パスが指定されていない場合は、DAQ およびテストデータのデフォルトディレクトリが使用されます。EA_Test.DataDirも参照してください。 27 = エクスポート形式(1 = catman、2 = ASCII、3 = ASCII + チャネル情報、4 = Excel(Office XP/2003)、5 = Diadem(.dat)、6 = nSoft DAC、7 = Vector MDF 3、8 = Excel(Office 2007 XLSX XML)、9 = Excel (Office 2007 XLSB バイナリ)、10 = Matlab 5.0、11 = RPC III、12 = nCode s3t、13 = ASAM MDF 4.1、14 = UFF58、15 = NI TDMS) 28 = エクスポートファイルのコメント 29 = 定期的なデータストレージのファイルモード (0 = 単一の増加ファイル、1 = 間隔ごとに1つのファイル) 30 = ジョブ終了後のDAQ設定をテストパラメータに追加(PropVal: True/False) 31 = DAQジョブをPCカードジョブとしてマーク(PropVal: True/False) 32 = PCカードジョブのファイル名(パス指定はできません。すべてのファイルはPC カードメモリ上の1つのフォルダに書き込まれます) 33 = PCカードジョブのファイル形式(0 = 4 バイト(INT)、1 = 2バイト(INT)、2 = 4バイト(FLOAT)) 34 = 圧縮ファイルの圧縮率(0 = 圧縮なし、可能な圧縮率: 500、1000、2000、5000、10000) 35 = PCカードジョブファイルのファイルコメント 36 = 各参加デバイスで一意のPC カードジョブファイル名を自動生成 (PropVal: True/False) 37 = 計測開始時のMGCplus内部クロックをPCクロックへ調整(PropVal: True/False) 38 = 複数のMGCplusデバイスのマスタ/スレーブモードでの動作(PropVal: True/False) 39 = *DAQジョブ後のデータ保存*に**計測中に定期的に実行**が使用されている場合のデータ保存間隔(秒単位)。パラメータ25も参照してください。 40 = *停止モード*に**サンプル数**を使用した場合に取得する値(サンプル)数。パラメータ9を参照。 41 = 一時データストレージのストレージ深さ(秒単位) (0 = 時間制限のないすべての値) 42 = 高速サンプルレート(Hz) • MGCplus: 1200, 2400, 4800, 9600(シングルチャネルボードのみ), 19200(シングルチャネルボードのみ) • 光インテロゲータ: 適用しない 43 = NTPサーバアドレス(xxx.xxx.xxx.xxx) 44 = NTP同期を使用(1 = 使用する、0 = 使用しない) 45 = NTPマスタタイムスタンプチャネル "|"で区切られた3つまでのチャネル(標準、低速、高速)のリスト 46 = ピークストレージモードの時間間隔(秒単位)。パラメータ24を参照。 47 = 保存(エクスポート)データの数値精度(0 = 8バイト倍精度、1 = 4バイト単精度、2 = 2バイト整数) 48 = DAQジョブ終了後に定期保存(パラメータ25を参照)で作成されたファイルを自動マージ(0 = マージしない、1 = マージする) 49 = DAQジョブ終了後にデジタル表示器とテーブルに統計を表示(0 = 表示しない、1 = 表示)

	50 = DAQジョブ中に接続が失われた場合のデバイスの非アクティブ化(0 = タイムアウトエラーでジョブを終了、1 = このデバイスのすべてのチャネルを非アクティブにしてジョブを続行、2 = 完全な初期化でジョブを再起動、3 = タイムアウト エラーなしでジョブを終了) 51 = eDAQリモートコントロールをアクティブ化 = 廃止 52 = eDAQリモートコントロールビット = 廃止 53 = SY42同期を使用(MGCplusのみ) (0 = SY42同期なし、1 = SY42同期あり) 54 = ASCIIエクスポートの小数点以下の桁数(0 = 自動小数点以下の桁数、1 = 完全精度、2 - 7 = 小数点以下1 - 6桁) 55 = 非同期読み取りモードを使用(0 = 非同期読み取りなし、1 = 非同期読み取り) 56 = PCタイミングを使用(0 = PCタイミングを使用しない、1 = PCタイミングを使用) 57 = 演算チャネルの自動ゼロ化(0 = 演算チャネルをゼロ化しない、1 = 演算チャネルをゼロ化) 58 = 統計ジャーナルファイルを作成(0 = ジャーナルを作成しない、1 = ジャーナルを作成) 59 = ジャーナルファイルの更新間隔(秒単位) 60 = トリガのバーストモード設定(0 = バースト モードなし、1 = バーストモード) 61 = 最大バースト数 62 = DAQジョブ開始時にQuantumXデバイス上の他のクライアントプロセスを閉じる(0 = 閉じない、1 = 閉じる) 63 = MGCとマスタデバイスの開始の開始遅延(秒単位) 64 = NTP同期補正(ミリ秒) 65 = DAQジョブ開始時にすべてのデバイスを再接続(0 = デバイスを再接続しない、1 = デバイスを再接続する) 66 = DAQ停止時にテストパラメータ編集ダイアログを表示(DAQ停止後の値の自動保存のみ)(0 = 表示しない、1 = 表示する) 67 = 統計ジャーナルのバックアップ間隔(0 = バックアップなし、1 = 毎日午前12時、2 = 毎日午後12時、3 = 毎週午前12時、4 = 毎週午後12時、5 = 毎月午前12時、6 = 毎月午後12 時) 68 = DAQ 状態"トリガ待ち"中のジャーナル更新(0 = "トリガ待ち"中にジャーナル更新しない、1 = "トリガ待ち"中に更新する) 69 = ジャーナルに書き込まれる値を指定するビットマスク(1 = スナップショット、2 = 最小、4 = 最大、8 = 平均)。 70 = ポストトリガフェーズ中の再トリガの最大数 71 = レベル開始トリガ: DAQの保存を開始するために、トリガチャネル値がトリガ値を上回るか下回る必要がある最小時間(秒)。 74 = DAQ開始時にイベントカウントをリセット(0 = カウンタをリセットしない、1 = カウンタをリセットする) 75 = ジョブ停止後にイベントファイルを作成(0 = ファイルを作成しない、1 = ファイルを作成する)。 79 = FTP/SFTPサーバへのローカルエクスポート後にファイルをアップロード(0 = ファイルをアップロードしない、1 = ファイルをアップロードする) 80 = FTP/SFTPサーバ名。この情報は、プロジェクトのすべてのジョブとレコーダで同じです。 81 = FTP/SFTPサーバ上のアカウントのユーザ名。この情報は、プロジェクトのすべてのジョブとレコーダで同じです。 82 = 非適用(安全上の理由から、FTP/SFTPアカウントのパスワードを取得することは許可されていません) 83 = ファイルを保存するFTP/SFTPサーバ上の作業ディレクトリ 84 = FTP/SFTP転送後にローカルファイルシステムからソースファイルを削除 85 = ビデオファイルをFTP/SFTPサーバにアップロード(0 = ビデオファイルをアップロードしない、1 = ビデオファイルをアップロードする) 86 = FTPデータ接続の確立方法を決定(0 = アクティブモード、1 = パッシブモード)。この情報は、プロジェクトのすべてのジョブとレコーダで同じです。 87 = SFTP認証用のPEM秘密キー。この情報は、プロジェクトのすべてのジョブとレコーダで同じです。

	88 = 非適用(安全上の理由から、SFTP秘密キーのパスフレーズを取得することは許可されていません) 89 = SFTP認証方法のタイプ(0 = パスワード認証、1 = 公開キー認証)。この情報は、プロジェクトのすべてのジョブとレコーダで同じです。 90 = FTP/SFTP接続に使用されるプロキシサーバのタイプ。この情報は、プロジェクトのすべてのジョブとレコーダで同じです。 91 = リクエストがルーティングされるプロキシサーバの名前またはアドレス。この情報は、プロジェクトのすべてのジョブとレコーダーで同じです。 92 = リクエストがルーティングされるプロキシサーバによって使用されるポート。この情報は、プロジェクトのすべてのジョブとレコーダーで同じです。 93 = 認証を必要とするプロキシサーバ経由のすべてのリクエストで送信されるユーザ名。この情報は、プロジェクトのすべてのジョブとレコーダーで同じです。 94 = 非適用(安全上の理由から、プロキシサーバのパスワードを取得することは許可されていません) 95 = レコーダのダウンサンプリングローパスフィルタを非アクティブ化(0 = ダウンサンプリングローパスフィルタをアクティブ、1 = ダウンサンプリングローパスフィルタを非アクティブ) 96 = 停止トリガの最大待機時間(秒単位) 97 = レコーダのデータ記録の停止時間(0 = 毎日24:00、1 = 毎日18:00、2 = 毎日12:00、3 = 毎日06:00、4 = 12時間ごと24:00:00と12:00、5 = 12時間ごとの06:00と18:00、6 = 6時間ごとの24:00、06:00、12:00、18:00、7 = 3時間ごとの24:00:00、03:00、06:00、09:00、12:00、15:00、18:00、21:00、8 = 2時間ごとの24:00、02:00、04:00.. .、9 = 毎時24:00、01:00、02:00...) 100 = ビデオカメラ1で記録されたビデオファイル名 101 = ビデオカメラ2で記録されたビデオファイル名 102 = ビデオカメラ3で記録されたビデオファイル名 103 = ビデオカメラ4で記録されたビデオファイル名 104 = ビデオカメラ5で記録されたビデオファイル名 105 = ビデオカメラ6で記録されたビデオファイル名 106 = ビデオカメラ7で記録されたビデオファイル名 107 = ビデオ カメラ8で記録されたビデオファイル名 200 = ビデオカメラ1のビデオ録画の開始条件 201 = ビデオカメラ2のビデオ録画の開始条件 202 = ビデオカメラ3のビデオ録画の開始条件 203 = ビデオカメラ4のビデオ録画の開始条件 204 = ビデオカメラ5のビデオ録画の開始条件 205 = ビデオカメラ6のビデオ録画の開始条件 206 = ビデオカメラ7のビデオ録画の開始条件 207 = ビデオカメラ8のビデオ録画の開始条件 0: DAQ ジョブの開始とともに自動実行 1: 開始トリガにより自動 2: 制限値とイベント監視を手動で使用するか、スクリプトまたはAutoSequenceを使用

	300 = ビデオカメラ1のビデオ録画の停止条件 301 = ビデオカメラ2のビデオ録画の停止条件 302 = ビデオカメラ3のビデオ録画の停止条件 303 = ビデオカメラ4のビデオ録画の停止条件 304 = ビデオカメラ5のビデオ録画の停止条件 305 = ビデオカメラ6のビデオ録画の停止条件 306 = ビデオカメラ7のビデオ録画の停止条件 307 = ビデオカメラ8のビデオ録画の停止条件 0: DAQ ジョブの停止とともに自動実行 1: 停止トリガにより自動 2: 制限値とイベント監視を手動で使用するか、スクリプトまたはAutoSequenceを使用 400 = DAQジョブの開始またはレコーダの記録開始時にスクリプトコードを実行(0 = コードを実行しない、1 = コードを実行する) 401 = DAQジョブの停止またはレコーダの記録開始時にスクリプトコードを実行(0 = コードを実行しない、1 = コードを実行する) 402 = DAQジョブの開始時またはレコーダの記録開始時に実行するスクリプトコード 403 = DAQジョブの停止時またはレコーダの記録開始時に実行するスクリプトコード 404 = トリガによるジョブ開始の場合、DAQ停止後に一時データ ストレージでcatman時間チャネルを正規化(0 = 時間チャネルを正規化しない、1 = 時間チャネルを0にリセット) 405 = 開始トリガのデジタル入力のデバウンス時間(ミリ秒) 406 = 停止トリガのデジタル入力のデバウンス時間(ミリ秒) 407 = MGCplusの記録データをNFS共有に出力 408 = MGCplusの記録データ出力のNFS URL 409 = "曜日"の停止条件。日曜日(0)から土曜日(6)までのビットマスク。 例: 月曜日 + 水曜日 + 金曜日 = 2^1 + 2^3 + 2^5 = 42。毎日 = 127。 410 = "曜日"の停止条件。停止する時刻。フルタイムのみ。0 = 00:00、1 = 01:00、14 = 14:00... 411 = "時刻"の停止条件。毎日。1 = 00:00 から1時間ごと、2 = 00:00 から2時間ごと、3 = 00:00から3時間ごと、4 = 00:00から6時間ごと、5 = 00:00、12 :00、6 = 06:00、18:00。
PropVal	DAQジョブパラメータの値。

戻り値	説明
0	エラーなし。
<0	エラー。DAQジョブが存在しないなど。

EA_Job.GetProperty、EA_Job.GetChannelActivation、EA_Job.AddTestParameter、EA_Job.ModifyTestParameter、EA_Job.DeleteTestParameterも参照してください。

注記

- 引数のTrueやFalseの場合、0 = Falseで、他はすべてTrueです。

- EasyScript Code Builderより、例えばConst JOB_DEFAULTSAMPLERATE = 6のように、意味のある名前を持つDAQジョブの引数のインデックスを定数リストに挿入することができます。これらの関数はCode Builderの"Constant declarations"グループにあります。

- プロパティインデックス26で指定されたデータ保存(エクスポート)のファイル名には、次のプレースフォルダが含まれる場合があります。:
  - %Job% : ジョブ名
  - %yy% : 年
  - %mm% : 月
  - %dd% :日
  - %h%: 時
  - %m%: 分
  - %s%: 秒
  - %Event%: イベントや制限トリップによるトリガでデータ保存操作を行う場合、イベントや制限トリップの名前に置き換えられます。
  - %ParName%形式のすべてのテストパラメータ
  - すべてのプロセス変数名(例えば"%MyVar%")

- サンプルはEA_Job.Addを参照してください。

EA_Job.SetStorageMode

DAQジョブ中のチャネルが一時データ保管に格納するかどうかを設定します。

対話型モードでの**DAQ jobs**ダイアログボックスにある**General**および**Job parameters**タブに相当します。

```
  EA_Job.SetStorageMode(ByVal Job As Variant, ByVal Channel As Variant, ByVal Mode
As Integer) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Channel	チャネル名または位置。
Mode	チャネルストレージの状態: 0 = No storage (このDAQジョブの計測は格納しません) 1 = Storage(このDAQジョブの計測は格納します)

戻り値	説明
0	エラーなし。
<0	エラー。チャネルまたはDAQジョブが存在しないなど。

注記

- 名前にチャネルの詳細を使用すると、実行に時間がかかります。代わりにEA_IO.ChanPosとチャネル名から、チャネル位置を決定します。

- このプロパティは、ジョブの開始前に設定する必要があることに注意してください! ジョブの実行中は変更できません。

例:

```
Sub CheckChannel()
  Dim saveMode As Integer
  EA_Job.GetStorageMode(1, "Pressure", saveMode)

  'Change the channel storage mode to STORAGE_ON if it was STORAGE_OFF.
  If saveMode = 0 Then
    EA_Job.SetStorageMode(1, "Pressure", 1)
  End If
End Sub
```

EA_Job.Start

指定したDAQジョブを開始します。一般的には通常のDAQジョブ(デバイスからPCへデータ転送)として実行しますが、MGCplusのPCカード上でスタンドアロン計測の、単独や追加の実行を開始することができます。

対話型モードでの、**Start DAQ job**ボタンや、対応するメニュー項目に相当します。

```
  EA_Job.Start(ByVal Job As Variant, Optional ByVal Mode As Integer, Optional ByVal
NoInit As Integer) As Long
```

引数	説明
Job	DAQジョブ名または位置。DAQジョブは1から始まります。
Mode	任意です。DAQジョブの実行モード 0 = 標準DAQジョブ(デフォルト) 1 = MGCplus内蔵のPCカードで計測 2 = 両方のモードで同時に計測
NoInit	任意です。チャネルの初期化を行うかどうかを指定します。 0 = チャネルの初期化を行います(デフォルト) 1 = チャネルの初期化を行いません

戻り値	説明
0	エラーなし。
<0	エラー。DAQジョブが存在しないなど。

注記

- DAQジョブ後の自動実行ステージの何れかでこのメソッドを呼び出す場合、ステージが"After transfer DAQ data block"、"After online computation"および"After storing DAQ data block"ではDAQジョブを再起動できません。実行ステージが"After complete data transfer cycle"の場合のみ、ジョブを再起動することができます。

- Mode = 1または2では、MGCplusのPCカード収集でパラメータ化されている必要があります。EA_Job.SetProperty を参照してください。

- サンプルはEA_Job.Addを参照してください。

EA_Job.Status (Read only)

DAQジョブの現在のステータスを取得します。

```
Property EA_Job.Status
```

戻り値	説明
0	DAQジョブは実行していません。
1	DAQジョブは実行しており、データを取得しています。
2	DAQジョブは実行しており、開始トリガの待機中です。

例

```
Sub WaitForJobEnd()
  Dim status As Long
  EA_Job.Start "Job1" 'Start the DAQ job with name "Job1".

  'Wait until the DAQ job is stopped, also status=0.
  Do
    status = EA_Job.Status
  Loop While status > 0

  'Carry out your own action here after the measurement data is stored.
  ...

  EA.Terminate TERMINATE_SCRIPT 'Now terminate the script execution.
End Sub
```

EA_Job.StoreData

最後の計測データをファイルに保存します。

この関数は廃止されました。代わりにEA_DAQ.SaveDataを使用してください。

対話型モードでの、**File** → **Save data from last DAQ job**に相当します。

```
  EA_Job.StoreData(ByVal FileName As String, ByVal Format As Integer, ByVal Comment
As String) As Long
```

引数	説明
FileName	ファイル名(任意のパス)。ファイルの拡張子を指定していない場合、デフォルトの拡張子を使用します(*Format*を参照)。
Format	ファイル形式: 0 = catman 1 = ASCII 2 = ASCII + チャネル情報 3 = Excel (Office XP/2003) 4 = Diadem DAT 5 = nSoft DAC 6 = Vector MDF 3 8 = Excel (Office 2007 XLSX XML) 8 = Excel (Office 2007 XLSB Binary) 9 = Matlab 5.0 10 = RPC III 11 = nCode s3t 12 = ASAM MDF 4.1 13 = UFF58 14 = NI TDMS
Comment	ファイルのコメント。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- 指定したファイルが既に存在しているかどうかはチェックしません。既存ファイルは上書きされます。
- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。
- サンプルはEA_Job.Addを参照してください。

EA_Job.Terminate

実行中のジョブを終了します。

対話型モードでの、**Stop DAQ job**ボタンや、対応するメニュー項目に相当します。

```
EA_Job.Terminate(Optional ByVal Mode As Integer) As Long
```

引数	説明
Mode	任意です。DAQジョブの実行モード 0 = すべてのDAQジョブを中止(デフォルト) 1 = 現在のDAQジョブを中止 2 = 反復実行中のDAQジョブをキャンセル 3 = 反復実行中のDAQジョブの待ち時間をキャンセル

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- DAQジョブの自動実行ステージの何れかでこのメソッドを呼び出す場合、ステージが"After transfer DAQ data block"、"After online computation"および"After storing DAQ data block"ではDAQジョブを再起動できません。実行ステージが"After complete data transfer cycle"の場合のみ、ジョブを再起動することができます。

- 特定の場合のみ、それぞれのモードが実用的です: Mode = 1は、すべてのDAQジョブを処理する場合のみ。Mode = 2または3は、ジョブの反復計測を使用している場合(*EA_Job.GetProperty* (Job, 3) > 0)、または反復間の遅延計測(*EA_Job.GetProperty* (Job, 4) > 0)のみ。
  EA_Job.GetPropertyも参照してください。

- サンプルはEA_Job.Addを参照してください。

EA_Job.TimeLeft (Read only)

計測可能な残り時間を秒で返します。計測中のすべてのデータは一時ファイルに書き込まれます(**File** → Options、**Data storage**タブ、**Temporary measurement storage**)。取得済みのデータを計測後に一時ファイルと同じドライブにエクスポートできるよう、可能な計測時間を計算します。すなわち、二倍の記憶容量をドライブの空き容量から差し引きます。

ドライブの空き容量が十分でない場合、計測は終了します。

対話型モードでは、計測中のステータスバーに計測時間を表示する( )に相当します。

```
Property EA_Job.TimeLeft
```

EA_Job.TriggerPowerBIDataTransfer

このメソッドを使用すると、catman Power BIコンソールで設定された更新間隔に関係なく、チャネルデータを Microsoft Power BIまたはInfluxdbに即時転送することができます。

```
EA_Job.TriggerPowerBIDataTransfer()
```

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- 実行する前に、PowerBIアドインがインストールされ、読み込まれていることを確認してください。
- 詳細については、catmanナレッジベースのトピック*Power BI*を参照してください。

EA_Math

catmanは、数学エンジンの下層に直接アクセスできる、いくつかのコアエンジンを提供しています。これらには多くの数学関数が含まれています。常にテストファイルのチャネルに含まれるデータを操作するEA_Testオブジェクトの数学関数とは対照的に、数学のコアクラスの関数はスクリプトの*変数*や*配列*を操作します。

ここで説明するクラスは、まだ数学ライブラリのコアではありあせん。ここでは数学のコアのサブセットのみ(それでもかなり大きいです)を公開しています。上級プログラマのためにcatmanは**EA_DSP**、**EA_Array**、**EA_Complex**、**EA_Matrix**および**EA_Stat**クラスから実際のコアクラスに直接アクセスすることを許容しています。これらのクラスは、*EA_Math*で公開されている関数だけでなく、より多くの関数を認識します。catmanは、これらクラス内の関数のオンラインヘルプを提供していませんが、多くの意味は*インテリセンス機能*で示される名前および引数リストより明確になる場合があります。

これらクラスと*EA_Test*のもう一つの違いは、*EA_Test*のメソッドは解析モードでのみ呼び出せるのに対し、コアメソッドは解析モードだけでなくDAQモードでも呼び出すことができるということです!

テストデータを解析するワークフローは、常に以下の手順で構成されています:

DAQモード(計測データは一時的にデータを保管しています):

1. 一時保管から配列に読み出します: EA_DAQ.GetTempStore
2. バリアント型へ*倍精度*配列を*コピー*します。
3. 配列データに数学演算を実行します。
4. 倍精度配列にデータをコピーバックします。これは一度に割り当てることはできません - ループ内で要素毎に割り当てる必要があります。
5. データをプロットするにはEA_Graph.PlotArrayXYメソッドを使用します(DAQモードでは*EA_Test.Plotxxx*メソッドは使用できません)。
6. 必要に応じて計算データを一時保管に書き戻します: EA_DAQ.SetTempStore

解析モード(計測データはテストファイルになります):

1. テストフィルから配列に読み出します: EA_Test.GetChannelData
2. バリアント型へ*倍精度*配列を*コピー*します。
3. 配列データに数学演算を実行します。
4. 倍精度配列にデータをコピーバックします。これは一度に割り当てることはできません - ループ内で要素毎に割り当てる必要があります。
5. EA_Test.PlotXYおよびEA_Test.PlotYTメソッドによりデータをプロットする場合、必要に応じてEA_Test.CreateChannelを使用します。これはデータのプロットに最善で安全な方法ですが、必須ではありません。またEA_Graph.PlotArrayXYメソッドを使用しても構いません。

コアクラスには五つのグループがあります:

- *EA_Math.DSP*
  信号解析関数(FFT、逆FFT、高速ハートレー変換、スペクトラム、積分、微分、ピーク検出、他)
- *EA_Math.Filter*
  デジタルフィルタリングデータ(IIRおよびFIRフィルタ)
- *EA_Math.Stat*
  統計関数、カーブフィッティングおよび補間
- *EA_Math.Arr*
  高速かつ効率的な配列操作
- EA_Math.Matrix
  基本的な行列演算および連立一次方程式の解を含む関数

注記

数学関数のほとんどはバリアント型の配列を扱います。関数はテストファイルや一時保管からデータを、倍精度型で取得または設定します。

```
Dim dBuf(10000) As Double
Dim vBuf As Variant
EA_Test.GetChannelData 1, "Pressure", dBuf
vBuf = dBuf
```

"Dim vBuf(10000) As Variant"のようなバリアント配列の割り当ては必要でなく、推奨していません! スクリプトで独自データを作成する場合でも、最初に倍精度型の配列を追加する必要があり、その後バリアント型の変数に割り当てます。

演算結果の更なる処理(例えばプロット)のため、通常データを倍精度配列にコピーバックする必要があります。これは一度に割り当てることはできません - ループ内で要素毎に割り当てる必要があります。

```
For i=0 To 99
  dBuf(i) = vBuf(i)
Next
```

結果データの配列は、数学関数自身で割り当てます。この場合のみ、バリアント変数を宣言する必要があります。

```
Dim out_arrFFTReal As Variant
```

⚠ 単一配列内の最大要素数は、おおよそ6.000.000値に制限されています。

このオブジェクトでは以下のメソッドが公開されています:

EA_Math.Arr.Absolute ・・・

EA_Math.DSP.PulseDetect ・・・

EA_Math.Arr.Absolute

入力配列の要素の絶対値を返します。

```
EA_Math.Arr.Absolute x As Variant, y As Variant
```

引数	説明
x	任意の次元の入力配列。
y	出力配列。入力配列と同一次元。

EA_Math.Arr.Addition

次の式を使用して二つの入力配列を加算します:

$z_i = x_i + y_i$

for i = 0, 1,...,n-1 nは入力配列の要素数です。入力配列は同じサイズである必要があります。

```
EA_Math.Arr.Addition x As Variant, y As Variant, z As variant
```

引数	説明
x	任意の次元の入力配列。
y	任意の次元の入力配列。
z	出力配列。入力配列と同一次元。

EA_Math.Arr.Clear

入力配列のすべての要素を0.0に設定します。

```
EA_Math.Arr.Clear x As Variant
```

引数	説明
x	入力および出力配列。

EA_Math.Arr.Copy

入力配列のすべての要素を出力配列にコピーします。

```
EA_Math.Arr.Copy x As Variant, y As Variant
```

引数	説明
x	入力配列。
y	出力配列。

EA_Math.Arr.Division

次の式を使用して二つの入力配列を除算します:

$z_i = x_i / y_i$

for i = 0, 1,...,n-1 nは入力配列の要素数です。入力配列は同じサイズである必要があります。

```
EA_Math.Arr.Division x As Variant, y As Variant, z As variant
```

引数	説明
x	任意の次元の入力配列。
y	任意の次元の入力配列。
z	出力配列。入力配列と同一次元。

EA_Math.Arr.MaxMin

入力配列xの最小値および最大値、および最初に出現したそれぞれのインデックスを検出します。

```
  EA_Math.Arr.MaxMin x As Variant, Max As Variant, MaxIndex As Variant, Min As
Variant, MinIndex As Variant
```

引数	説明
x	入力配列。
Max	入力配列で検出した最大値。
MaxIndex	入力配列内の*Max*のインデックス。
Min	入力配列で検出した最小値。
MinIndex	入力配列内の*Min*のインデックス。

EA_Math.Arr.Multiplication

次の式を使用して二つの入力配列を乗算します:

$z_i = x_i * y_i$

for i = 0, 1,...,n-1 nは入力配列の要素数です。入力配列は同じサイズである必要があります。

```
EA_Math.Arr.Multiplication x As Variant, y As Variant, z As variant
```

引数	説明
x	任意の次元の入力配列。
y	任意の次元の入力配列。
z	出力配列。入力配列と同一次元。

EA_Math.Arr.Negative

入力配列の要素の符号を反転して返します。

```
EA_Math.Arr.Negative x As Variant, y As Variant
```

引数	説明
x	任意の次元の入力配列。
y	入力配列と同一次元の出力配列。

EA_Math.Arr.Normalize

入力配列を正規化します。出力配列の形式は次の通りです:

```
y(i) = (x(i) - Average)/sDev)
```

*Average*は入力配列の平均であり、*sDev*は入力配列の標準偏差です。

```
EA_Math.Arr.Normalize x As Variant, y As Variant, Average As Variant, sDev As
Variant
```

引数	説明
x	入力配列。
y	出力配列。
Average	入力配列の平均。
sDev	入力配列の標準偏差。

EA_Math.Arr.Preset

入力配列のすべての要素を指定した値に設定します。

```
EA_Math.Arr.Preset x As Variant, Val As Variant
```

引数	説明
x	入力および出力配列。
Val	要素に出力する値。

EA_Math.Arr.Reverse

次の式を使用して入力配列内の要素の順序を逆にします:

$y_i = x_{n-1-i}$

for i = 0, 1,...,n-1 nは入力配列の要素数です。

```
EA_Math.Arr.Reverse x As Variant, y As Variant
```

引数	説明
x	入力配列。
y	出力配列。

EA_Math.Arr.Shift

次の式を使用して入力配列を*Shifts*で指定した個数分をシフトします:

$y_i = x_{i-Shifts}$

for i = 0, 1,...,n-1 nは入力配列の要素数です。

```
EA_Math.Arr.Shift x As Variant, Shifts As Variant, y As Variant
```

引数	説明
x	入力配列。
Shifts	要素がシフトされる配列内の位置の数。シフト数は正(右)方向または負(左) 方向に指定できます。
y	出力配列。

EA_Math.Arr.Sort

入力配列内の要素を昇順または降順に並べ替えます。

```
EA_Math.Arr.Sort x As Variant, Order As Variant, y As Variant
```

引数	説明
x	入力配列。
Order	並べ替えの方向。0 = 昇順、それ以外= 降順
y	出力配列。

EA_Math.Arr.Subset

入力配列から、*StartIndex*で始まり*Length*で指定した要素数の長さを含む部分配列を返します。

```
  EA_Math.Arr.Subset x As Variant, StartIndex As Variant, Length As Variant, y As
Variant
```

引数	説明
x	入力配列。
StartIndex	入力配列に部分配列を行う最初の要素のインデックス。
Length	入力配列から抽出する要素数。
y	抽出した配列。

EA_Math.Arr.Subtraction

次の式を使用して二つの入力配列を減算します:

$z_i = x_i - y_i$

for i = 0, 1,...,n-1 nは入力配列の要素数です。

```
EA_Math.Arr.Subtraction x As Variant, y As Variant, z As variant
```

引数	説明
x	任意の次元の入力配列。
y	任意の次元の入力配列。
z	入力配列と同一次元の出力配列。

EA_Math.DSP.AmpPhaseSpectrum

]時間領域信号の振幅と位相スペクトルを計算します。

```
  EA_Math.DSP.AmpPhaseSpectrum x As Variant, ByVal Unwrap As Variant, ByVal dt As
Variant, ya As Variant, yp As Variant, df As Variant
```

引数	説明
x	時間領域信号の配列。要素数は、2の累乗でなければなりません。
Unwrap	アンラップ: 1: 位相のアンラップは有効 0: 位相のアンラップは無効(-π ≤ phase ≤ +π).
dt	時間領域信号のサンプル周期(秒)。 dt = 1/fs fsは時間領域信号のサンプリング周波数です。
ya	各点で得られた振幅(RMS)の配列。0, df, 2*df,.... dfは時間領域信号のサンプルレート(df = 1/Samplerate)により決定されます。
yp	位相スペクトル(rad)。
df	周波数間隔をHzで返します。

EA_Math.DSP.ApplyWindow

時間領域信号に窓関数を適用します。

重要な注意点:　この関数は時間領域信号の元の振幅を維持しないことに注意してください。スペクトル振幅の定量解析を行う場合は、代わりに*ScaledWindow*を使用してください。

```
EA_Math.DSP.ApplyWindow x As Variant, y As Variant, ByVal WindowType As Integer
```

x	時間領域信号の配列。
y	窓化データの配列。
WindowType	WIN_TRIANGULAR = 1(三角) WIN_HANNING = 2(ハニング) WIN_HAMMING = 3(ハミング) WIN_BLACKMAN = 4(ブラックマン) WIN_EXACTBLACKMAN = 5(完全ブラックマン) WIN_BLACKMANHARRIS = 6(ブラックマンハリス) WIN_COSTAPERED = 7(コサインテーパ) WIN_FLATTOP = 8(フラットトップ)

EA_Math.DSP.AutoPowerSpectrum

時間領域信号のオートパワースペクトルを計算します。

```
  EA_Math.DSP.AutoPowerSpectrum x As Variant, ByVal dt As Variant, y As Variant, df
As Variant
```

オートパワースペクトルは、FFT*(x) * FFT(x)/n^2によって基本的なFFTから得られます。ここで、nは信号配列x内の点の数であり、*は複素共役を示します。

引数	説明
x	最初の時間領域信号の配列。要素数は、2の累乗でなければなりません。
dt	時間領域信号のサンプル周期(秒)。 dt = 1/fs fsは時間領域信号のサンプリング周波数です。
y	各点で得られたスペクトル振幅の配列。0, df, 2*df,.... dfは時間領域信号のサンプルレート(df = 1/Samplerate)により決定されます。
df	周波数間隔をHzで返します。

EA_Math.DSP.ComplexFFT

複素入力データの高速フーリエ変換を計算します。2の累乗でなければなりません。

```
EA_Math.DSP.ComplexFFT xr As Variant, xi As Variant, yr As Variant, yi As Variant
```

引数	説明
xr	データセットの実数部を含む入力配列。
xi	データセットの虚数部を含む入力配列。
yr	FFTの実数部を返す配列。
yi	FFTの虚数部を返す配列。

EA_Math.DSP.ComplexInvFFT

FFTから逆高速フーリエ変換を計算し、複素数を返します。

```
  EA_Math.DSP.ComplexInvFFT xr As Variant, xi As Variant, yr As Variant, yi As
Variant
```

引数	説明
xr	FFTの実数部を含む入力配列。
xi	FFTの虚数部を含む入力配列。
yr	実数部を返す配列。
yi	虚数部を返す配列。

EA_Math.DSP.Convolve

xおよびy配列入力の畳み込みを行います。

```
  EA_Math.DSP.Convolve in_arrX As Variant, in_arrY As Variant, out_arrConvolved As
Variant
```

引数	説明
in_arrX	x値の一次元配列。
in_arrY	y値の一次元配列。
out_arrConvolved	畳み込みデータの配列。

EA_Math.DSP.CrossPowerSpectrum

二つの時間領域信号のクロスパワースペクトルを計算します。

```
  EA_Math.DSP.CrossPowerSpectrum x As Variant, y As Variant, ByVal dt As Variant, ya
As Variant, yp As Variant, df As Variant
```

引数	説明
x	最初の時間領域信号の配列。要素数は、2の累乗でなければなりません。
y	二番目の時間領域信号の配列。要素数は、2の累乗でなければなりません。
dt	時間領域信号のサンプル周期(秒)。 `dt = 1/fs` fsは時間領域信号のサンプリング周波数です。
ya	各点で得られた振幅(RMS)の配列。0, df, 2*df,.... dfは時間領域信号のサンプルレート(df = 1/Samplerate)により決定されます。
yp	信号yと信号x間の位相差(rad)。
df	周波数間隔をHzで返します。

EA_Math.DSP.CrossSpectrum

二つの時間領域信号のクロススペクトルを計算します。

```
EA_Math.DSP.CrossSpectrum x As Variant, y As Variant, sr As Variant, si As Variant
```

クロスパワースペクトルは、基本的なFFTからFFT*(x) * FFT(y)/n_2から得られます。ここで、nは入力シーケンスxまたはyのサンプル数であり、FFT*(x)はFFT(x)の共役複素数です。nは2の累乗でなければなりません。

引数	説明
x	最初の時間領域信号の配列。要素数は、2の累乗でなければなりません。
y	二番目の時間領域信号の配列。要素数は、2の累乗でなければなりません。
sr	各点 0、df、2*df、.... における結果のスペクトル振幅の実数部を返す配列。ここで、dfは時間領域信号のサンプルレート(df = 1/Samplerate)によって決まります。
si	各点 0、df、2*df、.... における結果のスペクトル振幅の虚数部を返す配列。ここで、dfは時間領域信号のサンプルレート(df = 1/Samplerate)によって決まります。

EA_Math.DSP.Deconvolve

配列入力xおよびyの逆畳み込みを計算します。

```
  EA_Math.DSP.Deconvolve in_arrY As Variant, in_arrX As Variant, out_arrDeconvolved
As Variant
```

引数	説明
in_arrX	逆畳み込みを行うxの一次元配列。
in_arrY	逆畳み込みを行うyの一次元配列。
out_arrDeconvolved	逆畳み込みデータの配列。

EA_Math.DSP.Differentiate

式に従って入力配列xの微分を計算します。

$y_i = (x_{i+1} - x_{i-1}) / 2dt$

ここで、x_{-1} = xStart、x_n = xEnd、nはxの要素数です。

```
EA_Math.DSP.Differentiate x As Variant, ByVal dt As Variant, ByVal xStart As
Variant, ByVal xEnd As Variant, y As Variant
```

引数	説明
x	時間領域信号の配列。
dt	時間領域信号のサンプル周期(秒)。 dt = 1/fs fsは時間領域信号のサンプリング周波数です。
xStart	xの最初の要素に先行する、微分計算における入力配列xの初期条件。
xEnd	xの最後の要素に続く微分計算における入力配列xの最終条件。
y	離散差分データの配列。

EA_Math.DSP.FHT

高速ハートレー変換を計算します。

```
EA_Math.DSP.FHT x As Variant, y As Variant
```

引数	説明
x	入力配列。
y	出力配列。

EA_Math.DSP.Integrate

式に従って配列入力xの積分を計算します。

$$y_i = \sum_{j=0}^{i} [x_{j-1} + 4x_j + x_{j+1}] \times \frac{dt}{6}$$

```
  EA_Math.DSP.Integrate x As Variant, ByVal dt As Variant, ByVal xStart As Variant,
ByVal xEnd As Variant, y As Variant
```

引数	説明
x	時間領域信号のサンプル周期(秒)。 `dt = 1/fs` fsは時間領域信号のサンプリング周波数です。
dt	xの最初の要素に先行する、積分計算における入力配列xの初期条件。
xStart	xの最後の要素に続く積分計算における入力配列xの最終条件。
xEnd	離散差分データの配列。
y	離散積分の配列。

EA_Math.DSP.PeakDetect

データセット内のピーク(場所、振幅)の発生を決定します。

```
  EA_Math.DSP.PeakDetect in_arrData As Variant, ByVal in_Threshold As Variant, ByVal
in_Width As Variant, ByVal in_PeakType As Variant, ByVal in_Initialize As Variant,
ByVal in_EndOfData, out_arrLocations As Variant, out_arrAmplitudes As Variant, out_
arrSDerivatives As Variant
```

関数は極大値を決定するのに、二次多項式近似を使用します。*in_ Width*引数は、各近似に使用するデータ点数を指定します。極大値が*in_ Threshold*より低く、または極小値が*in_ Threshold*より高いと無視されます。しきい値は入力データセットの単位で指定する必要があります。

ピーク検出関数は、非常に大きなデータセットを処理することができます。このため、データの最初のチャンクが渡された場合を、*in_Initialize*フラグで指定する必要があります。データの最後のチャンクは*in_EndOfData*フラグにより通知します。データセット内のデータが約500万点を超える場合、チャンク化が必要となります。

引数	説明
in_arrData	処理を行うデータ配列。
in_Threshold	極大値が*in_ Threshold*より低く、または極小値が*in_ Threshold*より高いと無視されます。しきい値は入力データセットの単位で指定する必要があります。
in_Width	関数は極大値を決定するのに、二次多項式近似を使用します。*in_ Width*引数は、各近似に使用するデータ点数を指定します。
in_PeakType	0: 山 1: 谷
in_Initialize	TRUEはデータの最初のチャンクを渡します。すべてのデータを一つのチャンクに渡す場合、TRUEに設定します。
in_EndOfData	TRUEはデータの最後のチャンクを渡します。すべてのデータを一つのチャンクに渡す場合、TRUEに設定します。
out_arrLocations	検出した山/谷の位置の配列を返します。配列はチャンクの先頭から始まる位置のインデックスが含まれています。最初のインデックスは0です。例えばデータセットのdtとの積で位置から時間を決定することができます。
out_arrAmplitudes	検出した山/谷の振幅(入力データセットの単位)の配列を返します。
out_arrSDerivatives	検出した山/谷の二次導関数を含む配列返します。

EA_Math.DSP.PulseDetect

入力パルスを解析し、最良のパルスを表現するパラメータを生成します。

```
  EA_Math.DSP.PulseDetect in_arrX As Variant, out_Amp As Variant, out_Amp90 As
Variant, out_Amp50 As Variant, out_Amp10 As Variant, out_Top As Variant, out_Base
As Variant, out_TopOvershoot As Variant, out_BaseOvershoot As Variant, out_Delay As
Variant, out_Width As Variant, out_RiseTime As Variant, out_FallTime As Variant, out_
SlewRate As Variant
```

入力データセットは双峰分布を持つと仮定します。入力はパルスの離散表現であるため、*out_Delay*、*out_Width*、*out_RiseTime*、および*out_FallTime*引数は整数です。

引数	説明
in_arrX	処理を行うデータ配列。
out_Amp	ベース値とトップ値の差である振幅を返します。
out_Amp90	パルス振幅の90%の値を返します。
out_Amp50	パルス振幅の50%の値を返します。
out_Amp10	パルス振幅の10%の値を返します。
out_Top	入力パルスのトップ値を返します。
out_Base	入力パルスのベース値を返します。
out_TopOvershoot	最大値とトップ値の差であるオーバーシュート値を返します。
out_BaseOvershoot	ベース値と最小値の差であるアンダーシュート値を返します。
out_Delay	パルス検出前の要素数を返します。
out_Width	パルスの持続時間を表す要素数を返します。
out_RiseTime	*out_Amp90*と*out_Amp10*の差から計算したパルス立ち上がり時間を返します。
out_FallTime	*out_Amp10*と*out_Amp90*の差から計算したパルス立ち下がり時間を返します。
out_SlewRate	パルスのスルーレート値を返します。

EA_Math.DSP.RealFFT

時間領域信号の高速フーリエ変換を計算します 。入力データセットの要素数は、2の累乗でなければなりません。

```
EA_Math.DSP.RealFFT x As Variant, yr As Variant, yi As Variant
```

引数	説明
x	時系列の入力配列。
yr	FFTの実数部を返す配列。
yi	FFTの虚数部を返す配列。

出力データの周波数範囲は、空間0, df, 2*df,.... dfは時間領域信号のサンプルレート(df = 1/Samplerate)により決定されます。

EA_Math.DSP.RealInvFFT

逆高速FFTを計算し、時間領域信号を返します。

```
EA_Math.DSP.RealInvFFT xr As Variant, xi As Variant, yr As Variant
```

引数	説明
xr	FFTの実数部を含む入力配列。
xi	FFTの虚数部を含む入力配列。
yr	時間領域信号の結果配列。

EA_Math.DSP.ScaledWindow

スケールされた窓を時間領域信号に適用します。スケールされた窓は、時間領域信号全体のパワーを維持します。周波数スペクトルにおける振幅の定量解析を行う場合、常にスケールされた窓を使用する必要があります(すなわち*EA_Math.DSP.PeakDetect*によるピーク検出など、大きさと関連する場合)。

```
  EA_Math.DSP.ScaledWindow x As Variant, ByVal WindowType As Variant, WindowedData
As Variant, ENBW As Variant, CoherentGain As Variant
```

またこの関数はUnitSpectrumConversionなどのパワースペクトル上で計算を行う関数を使用する場合に、便利な定数を返します。

引数	説明
x	時間領域信号の配列。
WindowType	0: ユニフォーム 1: ハニング 2: ハミング 3: ブラックマンハリス 4: 完全ブラックマン 5: ブラックマン 6: フラットトップ 7: 4項ブラックマンハリス 8: 7項ブラックマンハリス
y	窓化データの配列。
ENBW	選択した窓の等価ノイズ帯域幅。周波数スパンでパワーを計算するのに、この値を使用します。
CoherentGain	ユニフォーム窓のピークゲインに対するの窓のピークゲイン。この値はユニフォーム窓のピーク信号ゲインの正規化に使用します。

EA_Math.DSP.Spectrum

時間領域信号のパワースペクトルを計算します。

```
EA_Math.DSP.Spectrum x As Variant, y As Variant
```

パワースペクトルは基本的なFFTから得られます。$|FFT(x)|^2/n^2$

引数	説明
x	時間領域信号の配列。要素数は、2の累乗でなければなりません。
y	各点で得られたスペクトル振幅の配列。0, df, 2*df,.... dfは時間領域信号のサンプルレート(df = 1/Samplerate)により決定されます。

EA_Math.DSP.SpectrumUnitConversion

入力スペクトル(パワー、振幅、ゲイン)をログ(dBまたはdBm)やスペクトル密度などの代替形式へ変換します。

```
  EA_Math.DSP.SpectrumUnitConversion Spectrum As Variant, ByVal SpectrumType As
eUnitConversionInputTypes, ByVal ScaleType As eUnitConversionScaleTypes, ByVal
OutputUnit As eUnitConversionOutputUnits, ByVal df, ByVal ENBW, ByVal CoherentGain,
ConvertedSpectrum As Variant
```

ENBWおよびCoherentGain引数は、*ScaledWindow*関数によって出力されます。この入力は、スペクトル密度の出力形式(OutputUnitの最後の4つを選択)を使用する場合にのみ必要です。

引数	説明
SpectrumType	0: AutoPowerSpectrumから得たパワー(rmsの二乗) 1: AmpPhaseSpectrumから得た振幅(rms)
ScalingType	0: 線形 1: dB 2: dBm
OutputUnit	0: rms 1: peak 2: rms^2 3: $peak^2$ 4: $rms\sqrt{df}$ 5: $rms\sqrt{peak}$ 6: $rms^2\sqrt{df}$ 7: $peak^2\sqrt{df}$
df	周波数間隔(Hz)。
ENBW	等価ノイズ帯域幅は*ScaledWindow*から得られます。
CoherentGain	窓のピークゲインは*ScaledWindow*から得られます。
ConvertedSpectrum	変換したスペクトルの配列。

EA_Math.Filter.Butterworth

デジタルバタワースフィルタを使用して、入力配列(x)をフィルタリングします。

```
  EA_Math.Filter.Butterworth x As Variant, ByVal FilterType As Variant, ByVal fs
As Variant, ByVal fl As Variant, ByVal fh As Variant, ByVal Order As Variant, y As
Variant
```

引数	説明
x	入力配列。
FilterType	フィルタの種類: 1 = ローパス 2 = ハイパス 3 = バンドパス 4 = バンドストップ
fs	サンプリング周波数(Hz)。
fl	ローパス、バンドパス、およびバンドストップフィルタの低域カットオフ周波数。
fh	ハイパス、バンドパス、およびバンドストップフィルタの高域カットオフ周波数。
Order	フィルタ次数。
y	フィルタリングされた信号の出力配列。

EA_Math.Filter.Chebyshev

デジタルチェビシェフフィルタを使用して、入力配列(x)をフィルタリングします。

```
  EA_Math.Filter.Butterworth x As Variant, ByVal FilterType As Variant, ByVal fs
As Variant, ByVal fl As Variant, ByVal fh As Variant, ByVal Order As Variant, y As
Variant
```

引数	説明
x	入力配列。
FilterType	フィルタの種類: 1 = ローパス 2 = ハイパス 3 = バンドパス 4 = バンドストップ
fs	サンプリング周波数(Hz)。
fl	ローパス、バンドパス、およびバンドストップフィルタの低域カットオフ周波数。
fh	ハイパス、バンドパス、およびバンドストップフィルタの高域カットオフ周波数。
Ripple	通過帯域リップル(db)。
Order	フィルタ次数。
y	フィルタリングされた信号の出力配列。

EA_Math.Filter.Elliptic

デジタル楕円フィルタを使用して、入力配列(x)をフィルタリングします。

```
  EA_Math.Filter.Elliptic x As Variant, ByVal FilterType As Variant, ByVal fs As
Variant, ByVal fl As Variant, ByVal fh As Variant, ByVal Ripple As Variant, ByVal
Atten As Variant, ByVal Order As Variant, y As Variant
```

引数	説明
x	入力配列。
FilterType	フィルタの種類: 1 = ローパス 2 = ハイパス 3 = バンドパス 4 = バンドストップ
fs	サンプリング周波数(Hz)。
fl	ローパス、バンドパス、およびバンドストップフィルタの低域カットオフ周波数。
fh	ハイパス、バンドパス、およびバンドストップフィルタの高域カットオフ周波数。
Ripple	通過帯域リップル(db)。
Atten	阻止帯域減衰(db)。
Order	フィルタ次数。
y	フィルタリングされた信号の出力配列。

EA_Math.Filter.InvChebyshev

デジタル逆チェビシェフフィルタを使用して、入力配列(x)をフィルタリングします。

```
  EA_Math.Filter.InvChebyshev x As Variant, ByVal FilterType As Variant, ByVal fs As
Variant, ByVal fl As Variant, ByVal fh As Variant, ByVal Atten As Variant, ByVal Order
As Variant, y As Variant
```

引数	説明
x	入力配列。
FilterType	フィルタの種類: 1 = ローパス 2 = ハイパス 3 = バンドパス 4 = バンドストップ
fs	サンプリング周波数(Hz)。
fl	ローパス、バンドパス、およびバンドストップフィルタの低域カットオフ周波数。
fh	ハイパス、バンドパス、およびバンドストップフィルタの高域カットオフ周波数。
Atten	阻止帯域減衰(db)。
Order	フィルタ次数。
y	フィルタリングされた信号の出力配列。

EA_Math.Stat.ExponentialFit

最小二乗法を使用してデータポイント(x, y)の最良の指数近似`y = a*exp(b*x)`のを検索します。

```
EA_Math.Stat.ExponentialFit x, y, z, a, b, mse
```

引数	説明
x	バリアント型[入力]: x値の一次元配列。
y	バリアント型[入力]: y値の一次元配列。
z	バリアント型[出力]: 指数近似の配列。
a	バリアント型[出力]: 振幅。
b	バリアント型[出力]: 指数。
mse	バリアント型[出力]: 二乗平均誤差。

MSE(二乗平均誤差)はEA_Math.Stat.Mseより求められます。

EA_Math.Stat.Histogram

入力配列xのヒストグラムを計算します。ヒストグラムは、入力配列内の要素が区間iに該当する回数をカウントすることによって取得します。

```
y = EA_Math.Stat.Histogram(x, xStart, xEnd, nIntervals, y, z)
```

引数	説明
x	バリアント型[入力]: 一次元配列。
xStart	バリアント型[入力]: 範囲の下限。
xEnd	バリアント型[入力]: 範囲の上限。
nIntervals	バリアント型[入力]: 範囲を分割する間隔の数。
y	バリアント型[出力]: ヒストグラムカウントデータの配列。
z	バリアント型[出力]: 間隔制限の配列。

EA_Math.Stat.LinearFit

最小二乗法を使用してデータポイント(x, y)の最良の指数近似y = m*x+bを最もよく表す傾きと切片の値を見つけます。

```
EA_Math.Stat.LinearFit x, y, z, Slope, Intercept, mse
```

引数	説明
x	バリアント型[入力]: x値の一次元配列。
y	バリアント型[入力]: y値の一次元配列。
z	バリアント型[出力]: 指数近似の配列。
Slope	バリアント型[出力]: 線の傾き(m)。
Intercept	バリアント型[出力]: 切片(b)。
mse	バリアント型[出力]: 二乗平均誤差。

MSE(二乗平均誤差)はEA_Math.Stat.Mseより求められます。

EA_Math.Stat.Mean

入力配列の平均値を返します。

```
y = EA_Math.Stat.Mean(x)
```

引数	説明
x	バリアント型[入力]: 一次元配列。
y	バリアント型[戻り値]: 平均値。

EA_Math.Stat.Median

入力配列の中央値を返します。

```
y = EA_Math.Stat.Median(x)
```

引数	説明
x	バリアント型[入力]: 一次元配列。
y	バリアント型[戻り値]: 中央値。

EA_Math.Stat.Mode

入力配列のモードを検索します。最頻値は、特定のサンプルのセットで最も頻繁に発生する値として定義されます。たとえば、入力配列が`x = {0,1,3,3,4,4,4,5,5,7}`の場合、xの最頻値は4.375です。これは、配列内で最も頻繁に出現する値であるためです。この関数は、入力配列のヒストグラムの観点からモードを決定します。

```
y = EA_Math.Stat.Mode(x, xStart, xEnd, nIntervals)
```

引数	説明
x	バリアント型[入力]: 一次元配列。
xStart	バリアント型[入力]: 範囲の下限。
xEnd	バリアント型[入力]: 範囲の上限。
nIntervals	バリアント型[入力]: 範囲の間隔。
y	バリアント型[戻り値]: モード値。

EA_Math.Stat.Moment

指定された順序で入力配列の平均に関するモーメントを計算します。

```
y = EA_Math.Stat.Moment(x, Order)
```

引数	説明
x	バリアント型[入力]: 一次元配列。
Order	バリアント型[入力]: モーメントの次数。次数は0よりも大きくなければなりません。
y	バリアント型[戻り値]: モーメント値。

EA_Math.Stat.Mse

入力データセットxおよびyの平均二乗誤差(MSE)を計算します。

```
mse = EA_Math.Stat.Moment(x, y)
```

引数	説明
x	バリアント型[入力]: 最初の一次元配列。
y	バリアント型[入力]: 二番目の一次元配列。
mse	バリアント型[戻り値]: 平均二乗誤差。

EA_Math.Stat.PolynomialFit

最小二乗法を使用して、データ点 (x, y) を最もよく表す多項式近似y = a0 + a1*x + a2*x2 + ... を検索します。

```
EA_Math.Stat.PolynomialFit x, y, Order, z, a, b, mse
```

引数	説明
x	バリアント型[入力]: x値の一次元配列。
y	バリアント型[入力]: y値の一次元配列。
Order	多項式の次数。
z	バリアント型[出力]: 最適な配列。
Coeff	バリアント型[出力]: 係数の配列(a0,a1,a2....)。
mse	バリアント型[出力]: 二乗平均誤差。

MSE(二乗平均誤差)はEA_Math.Stat.Mseより求められます。

EA_Math.Stat.RMS

入力配列の二乗平均平方根値(Root Mean Square: RMS)を返します。RMS 値は次のように計算します。

```
y = EA_Math.Stat.RMS(x)
```

引数	説明
x	バリアント型[入力]: 一次元配列。
y	バリアント型[戻り値]: RMS値。

EA_Math.Stat.StdDev

入力配列の平均値と標準偏差を返します。

```
EA_Math.Stat.StdDev x, Mean, sDev
```

引数	説明
x	バリアント型[入力]: 一次元配列。
Mean	バリアント型[戻り値]: 平均値。
sDev	バリアント型[戻り値]: 標準偏差値。

EA_Math.Stat.Variance

入力配列の分散と平均値を計算します(*分散*は標準偏差の2乗です)。

```
EA_Math.Stat.Variance x, Mean, v
```

引数	説明
x	バリアント型[入力]: 一次元配列。
Mean	バリアント型[戻り値]: 平均値。
v	バリアント型[戻り値]: 分散値。

EA_Panel

catmanEasy/AP のパネルとほとんどの視覚化オブジェクトには、*EA_Panel*経由でアクセスします。

このオブジェクトは以下が行えます:

- パネルを表示: EA_Panel.ShowPanel
- すべてのパネル数を取得(通常のパネル、スコープおよびページ): EA_Panel.PanelCount
- フルスクリーンモードに切り替え: EA_Panel.FullScreenMode
- パネルのロック/アンロック: EA_Panel.LockPanels
- ファイルからパネルを読み込み: EA_Panel.Load
- パネルをファイルに保存: EA_Panel.Save
- 既存のパネルを削除: EA_Panel.Unload
- パネルまたは印刷ページのタイトルを変更: EA_Panel.Rename
- 現在のパネル(通常のパネル、スコープ、ページ)のインデックスとキャプションを取得: EA_Panel.GetCurrentPanel
- 指定したテンプレートファイルから、オブジェクト設定を読み込み: EA_Panel.LoadObjectFromTemplate
- キャプション、色、背景画像および状態といったプロパティ設定: EA_Panel.SetCaption、EA_Panel.SetColor、EA_Panel.SetBackgroundImage、EA_Panel.SetState
- パネルの背景またはキャンバスオブジェクトにテキストを出力: EA_Panel.CanvasSetText
- パネル上の可視化オブジェクトの制御: EA_Panel.GetObjectList、EA_Panel.LockObject、EA_Panel.SetObjectState、EA_Panel.SetAccessMode
- 可視化オブジェクトへのアクセス: EA_Panel.GetDirectControl、EA_Panel.GetObjectProperty、EA_Panel.ReleaseDirectControl
- テーブルオブジェクトプロパティの設定: EA_Panel.SetTableProperty、EA_Panel.GetCell、EA_Panel.SetCell、EA_Panel.InsertRow、EA_Panel.RemoveRow、EA_Panel.RecalcTable、EA_Panel.RefreshTable、EA_Panel.SetColWidth、EA_Panel.HideTableFocusRectangle
- テーブルの読み込み/保存: EA_Panel.LoadTable、EA_Panel.SaveTable
- リストボックスのための関数: EA_Panel.SetListboxItems、EA_Panel.ClearListbox、EA_Panel.AddListboxItem、EA_Panel.RemoveListboxItem、EA_Panel.GetListboxItem、EA_Panel.GetListboxSelIndex、EA_Panel.SetListboxSelIndex
- 表示するチャネルを割り当て: EA_Panel.SetRTDisplayChannel
- チャネルをグラフのX軸に割り当て: EA_Panel.SetXAxisChannel
- オブジェクトの内容や状態を取得/設定: EA_Panel.GetValue、EA_Panel.SetValue
- フローティングパネルの関数: EA_Panel.ShowFloatingPanel、EA_Panel.HideFloatingPanel、EA_Panel.SetFloatingPanelURL
- アナログメータやバーオブジェクトの警報および警告レベルを設定: EA_Panel.SetAlarmLevel
- デジタル表示器、メータ、バー表示器、テーブルの統計(最小/最大)表示をリセット: EA_Panel.ResetStatistics
- テキストオブジェクトのリフレッシュ: EA_Panel.RefreshText
- パネル上のすべての可視化オブジェクトを更新: EA_Panel.RefreshAll
- ユーザ入力オブジェクト(テキストボックスなど)にフォーカスを設定: EA_Panel.SetObjectFocus
- 解析モードでパネルを印刷: EA_Panel.PrintPage
- パネルやページをクリップボードにコピー、または画像として保存: EA_Panel.PrintScreen、EA_Panel.SavePanelToJPEG
- 解析モードでテーブルを印刷: EA_Panel.PrintTable
- 既存のレポートテンプレートに基づいて新しいMicrosoft Wordレポートを作成: EA_Panel.CreateReport

以下のアクションは、現在のバージョンでサポートされていません:

- パネルオブジェクトの移動や
- オブジェクトプロパティの読み出し

注記

- グラフの構成には、EA_Graphオブジェクトを使用します。多くのオプションがここから入手でき、個別のオブジェクトを使用してグラフのみのプリンタ出力も可能です。
- 上級プログラマは直接個々のパネルオブジェクトにアクセスし、完全に制御することができます(catmanのチャネルに関連しないの配列データのプロットや、テーブルオブジェクトにシートを追加するなど)。EA.GetObjectメソッドも参照してください。

このオブジェクトでは以下のメソッドが公開されています:

EA_Panel.AddListboxItem ・・・

EA_Panel.SetColor ・・・

EA_Panel.AddListboxItem

リストボックス内に項目を挿入します。項目は任意の位置、または最後尾に追加することができます。また項目(ItemData)に番号を割り当てることもできます。これにより、テキストと番号の組み合わせを管理することができます。

```
EA_Panel.AddListboxItem(ByVal Panel As Variant, ByVal ObjectName As String, ByVal ItemText As String, ByVal ItemData As Long, Optional ByVal Index As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、すべてのパネルでリストボックスを検索することを意味します。
ObjectName	リストボックス名。
ItemText	項目リストのテキスト。
ItemData	項目リストの番号。
Index	オプションです。項目を挿入するインデックス。Index = 0または未記入の場合、項目はリストの最後に追加されます。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはリストボックスが存在しないなど。

注記

- 単一コマンドでリストを埋めるにはEA_Panel.SetListboxItemsを使用します。
- 複数のリストボックスが同じ名前で、すべてのパネルを検索(Panel = 0)している場合、最初に見つかったリストボックスを使用します。

例

```
Dim index As Integer
Dim value As String
Dim itemText As String
Dim itemData As Long

'Clear all listbox entries.
EA_Panel.ClearListbox(1, "COMBOBOX_1")

'Add three new entries: "ListItem1", "ListItem2" and "ListItem4".
EA_Panel.SetListboxItems(1, "COMBOBOX_1", "ListItem1" & vbCrLf & "ListItem2" & vbCrLf & "ListItem4")

'Remove the first entry "ListItem1".
EA_Panel.RemoveListboxItem(1, "COMBOBOX_1", 1)

'Insert a new entry "ListItem3" at the 2nd position, also between entry "ListItem2" and "ListItem4".
EA_Panel.AddListboxItem(1, "COMBOBOX_1", "ListItem3", 3, 2)

'Select the first entry "ListItem2".
EA_Panel.SetListboxSelIndex(1, "COMBOBOX_1", 1)

'Get the index of currently selected entry.
EA_Panel.GetListboxSelIndex(1, "COMBOBOX_1", index) 'index = 1. Result of last run step.
```

```
  'Get the text of currently selected entry.
  EA_Panel.GetValue(1, "COMBOBOX_1", value) 'value = "ListItem2"

  'Get the text and number of entry at position 2.
  EA_Panel.GetListboxItem(1, "COMBOBOX_1", 2, itemText, itemData) 'itemText =
"ListItem3", itemData = 3
```

EA_Panel.CanvasSetText

パネルの背景またはキャンバスオブジェクトにテキストを出力します。

注: パネルの背景への描画と、catman視覚化ツールボックスによって公開されるキャンバスオブジェクトタイプを混同しないでください。詳細については、キャンバスオブジェクトの操作を参照してください。

```
  EA_Panel.CanvasSetText(ByVal Panel As Variant, ByVal ObjectName As String, ByVal
Text As String) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、*すべて*のパネルを検索することを意味します。
ObjectName	キャンバスオブジェクトの名前。ObjectNameが空の場合、テキストは現在のパネルに出力されます。
Text	出力される新しいテキスト。

注記

- パネルキャンバスのすべての関数は、*PictureBox*データ型として公開されます。関数のフルセット(テキスト位置、フォント、色、線や円の描画などの設定)に対処するため、EA_GetObjectとともにこの変数を使用します。テキストは常に*CurrentX*、*CurrentY*座標に出力します。

例

```
  Dim oCanvas As PictureBox
  Set oCanvas = EA.GetObject(OB_PANELCANVAS, "")

  oCanvas.Cls 'Erase background
  oCanvas.CurrentX = 500 'TWIPS
  oCanvas.CurrentY = 200 'TWIPS
  oCanvas.FontName= "Verdana"
  oCanvas.FontSize = 10
  oCanvas.ForeColor = vbBlue
  EA_Panel.CanvasSetText "Hello world" 'Text on panel.
  EA_Panel.CanvasSetText 1, "CANVAS_1", "Hello world" 'Text on canvas object.
```

EA_Panel.ClearListbox

リストボックスの内容を削除します。

```
EA_Panel.ClearListbox(ByVal Panel As Variant, ByVal ObjectName As String) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、すべてのパネルでリストボックスを検索することを意味します。
ObjectName	リストボックス名。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはリストボックスが存在しないなど。

注記

- リストの項目を一つ削除するにはEA_Panel.RemoveListboxItemを使用します。
- 複数のリストボックスが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったリストボックスを使用します。

EA_Panel.CreateReport

既存のレポートテンプレートに基づいて、Microsoft Wordのレポートを新規作成します。

```
  EA_Panel.CreateReport(ByVal DocTemplate As String, ByVal FileName As String, ByVal
DoPrint As Integer)
```

引数	説明
DocTemplate	レポートテンプレートファイル名。
FileName	新規レポートファイル名。
DoPrint	新規レポートを自動的に印刷するかどうかを示します。

EA_Panel.FullScreenMode

フルスクリーンモードのオン/オフを切り替えます。

対話型モードでの、**Visualization**タブの**Full screen mode**ボタンに相当します。

```
Property EA_Panel.FullScreenMode
```

戻り値	説明
0	フルスクリーンモード: オフ
1	フルスクリーンモード: オン

注記

- フルスクリーンモードでは、リボンおよびすべてのコンポーネントウィンドウはマスクされ、グラフィック表示に利用できるスペースを多く確保します。
- フルスクリーンモードではフルスクリーンモードをオフにするボタン(*Full screen mode off*)を、画面左上に表示します。

例

```
EA_Panel.FullScreenMode = 1 'Enable full-screen mode.
```

EA_Panel.GetCell

これはテーブルオブジェクトに対処する最も汎用的な関数の一つです。個々のセルからの情報を返します。*DataType*により型が異なる可能性があり、セルの種類にも依存します(例えば、編集可能なテキストであればテキストを返し、数値のセルは数字を返します)。

このメソッドは、現在のアクティブシート上のセルに適用されることに注意してください。シートをアクティブにするには*EA_Panel.SetTableProperty*プロパティを使用します。一枚のシートのみのテーブルの場合、アクティブにする必要はありません。

```
  EA_Panel.GetCell(ByVal Panel As Variant, ByVal ObjectName As String, ByVal Col As
Long, ByVal Row As Long, Value As Variant, Optional ByVal DataType As Integer) As
Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、*すべての*パネルがテーブルを検索することを意味します。
ObjectName	テーブル名。
Col	セルの列番号。列の最初のインデックスは1です。
Row	セルの行番号。行の最初のインデックスは1です。
Value	返される情報。*DataType*だけでなく、セルの種類によって異なります。
DataType	TYPE_CONTENT (0) テキストセル: セル内のテキストを返します。 数値セル: セルの値を返します。 コンボボックスセル: ";"で区切った項目のリストを返します。 他のすべてのセル: 適用されません。 TYPE_SELECTION (1) チェックボックスセル: チェックボックスの状態を返します(0=チェックなし、1=チェックあり)。 コンボボックスセル: 選択したアイテムのインデックスを返します(最初のインデックスは0)。 他のすべてのセル: 適用されません。 TYPE_CAPTION (2) ボタンおよびチェックボックスセル: キャプションを返します。 他のすべてのセル: 適用されません。 TYPE_FORECOLOR (3) テキストの色を返します。 ピクチャセルには適用されません。 TYPE_BACKCOLOR (4) セルの背面色を返します。 ピクチャセルには適用されません。

DataType	TYPE_PICTURE (5) 未対応 TYPE_PICTURESOURCE (6) ピクチャセルにリンクされているグラフの名を返します。 他のすべてのセル: 適用されません。 解析モードでのみ可能です! TYPE_DATASOURCE (7) テキストセルと数値セル: チャネルから取得したリンクされた情報をセルに返します(最大値など)。値にはデータソースの説明文字列が含まれます。セルデータソースの表記では、データソースを指定する方法について説明します。 他のすべてのセル: 適用されません。 解析モードでのみ可能です! TYPE_CELLTYPE (8) セルのタイプを返します: 1 = 動的テキスト 2 = 動的数値 3 = 静的テキスト 4 = 静的数値 5 = ボタン 6 = 画像 7= チェックボックス 8 = コンボボックス TYPE_NUMPRECISION (9) 数値セル: 数値セルの小数点右側に割り当てられた桁数を返します。 他のすべてのセル: 適用されません。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはテーブルが存在しないなど。

EA_Panel.SetCellも参照してください。

注記

- この関数は通常、ユーザインタラクティブオブジェクトを情報を交換するために適用し、*SetValue*、*SetColor*、*SetCaption*などはテーブルのセルに適用されない場合があります。

例

```
  Sub CellValue()
    'Fetches a numerical value from Table "Result" of the first panel. Content of the
cell in second col and third row is written into variable "Res".
    Dim Res As Double
    EA_Panel.GetCell 1, "Result", 2, 3, Res, TYPE_CONTENT
  End Sub
```

EA_Panel.GetCurrentPanel

現在の可視化パネルの、インデックスおよびキャプションを返します。

```
EA_Panel.GetCurrentPanel(Index As Integer, Caption As String)
```

引数	説明
Index	現在のパネルのインデックス。最初のパネルは1です。
Caption	現在のパネルのキャプション。

注記

- これは通常のパネル、スコープパネルおよび、印刷ページのみをサポートしています。フローティングパネルはサポートしていません。

EA_Panel.GetDirectControl

可視化パネルオブジェクトを低レベルでプログラムする準備をします。オブジェクトをロックすることにより、独自プログラムとcatmanが干渉するのを防ぎます。catmanに制御を戻すにはEA_Panel.ReleaseDirectControlを使用します。

```
EA_Panel.GetDirectControl(ByVal ObjectName As String, Optional ByVal Panel As
Variant) As Long
```

引数	説明
ObjectName	オブジェクト名。スコープまたはフローティングパネルでは、グラフ、テーブルや表示オブジェクトを含む"layer"を指定します。 任意です。省略した場合、オブジェクトのすべての通常のパネルを検索します。
Panel	パネル名または位置。最初のパネルは1です。省略した場合、すべての通常のパネルオブジェクトを検索します。パネルは通常のパネル、スコープパネルおよびフローティングパネルのいずれかになります。 スコープやフローティングパネルの場合、Panelパラメータは必須です。

戻り値	説明
0	エラーなし。
<0	エラー。オブジェクトが存在しないなど。

注記

- EA.GetObjectおよびローレベルのグラフプログラミングも参照してください。
- このメソッドは今のところ、グラフ、表、アナログメータ、バー、数字、LED表示器、およびテキストオブジェクトタイプをサポートしています。

EA_Panel.GetListboxItem

リストボックスの任意の位置の項目リストを返します。

```
  EA_Panel.GetListboxItem(ByVal Panel As Variant, ByVal ObjectName As String, ByVal
Index As Integer, ItemText As String, ItemData As Long) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、*すべてのパネル*でリストボックスを検索することを意味します。
ObjectName	リストボックス名。
Index	項目の位置。リストの最初の項目の位置は1です。
ItemText	項目リストのテキスト。
ItemData	項目リストの番号。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはリストボックスが存在しないなど。

注記

- 選択した項目の位置を検索するにはEA_Panel.GetListboxSelIndexを使用します。
- 複数のリストボックスが同じ名前で、すべてのパネルを検索(*Panel* = 0)する場合、最初に見つかったリストボックスを使用します。

EA_Panel.GetListboxSelIndex

選択したリストボックス項目の位置を検索します。

```
  EA_Panel.GetListboxSelIndex(ByVal Panel As Variant, ByVal ObjectName As String,
Index As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、すべてのパネルでリストボックスを検索することを意味します。
ObjectName	リストボックス名。
Index	選択した項目の位置。リストの最初の項目は1です。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはリストボックスが存在しないなど。

注記

- 選択した項目の値(テキスト)を検索するにはEA_Panel.GetValueを使用します。リストボックスの任意の要素の値を知るにはEA_Panel.GetListboxItemを使用します。

- 複数のリストボックスが同じ名前で、すべてのパネルを検索(*Panel* = 0)する場合、最初に見つかったリストボックスを使用します。

EA_Panel.GetObjectList

パネル上にある可視化オブジェクトの数および詳細(型、インデックスおよび名前)を返します。

```
  EA_Panel.GetObjectList(ByVal Panel As Variant, NofObjects As Integer, ObjectTypes()
As Integer, ObjectIndices() As Integer, ObjectNames() As String) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。
NofObjects	指定したパネル上の可視化オブジェクト数。
ObjectTypes	可視化オブジェクトのコレクション型: 1 = テーブル 2 = Real-timeグラフまたはPost-processグラフ 3 = デジタル表示器 4 = アナログメータ 5 = LED 6 = バー表示器 7 = テキスト 8 = ボタン 9 = テキストボックス 10 = リストボックス 11 = チェックボックス 19 = Polarグラフ 21 = ノブ 22 = スライダ 23 = LED列 26 = Cursorグラフ 28 = スライドスイッチ 29 = On/Offスイッチ 30 = ウォーターフォールダイアグラム 31 = 3Dダイアグラム 32 = スペクトログラム
ObjectIndices	可視化オブジェクトのインデックスコレクション。コレクション内のインデックス要素は、同じ型のオブジェクト順序を表しています。各オブジェクト型のインデックスは1から始まります。
ObjectNames	可視化オブジェクトの名前コレクション。

戻り値	説明
0	エラーなし。
<0	エラー。パネルが存在しないなど。

例

```
  Dim i As Integer
  Dim NofObjects As Integer
  Dim ObjectTypes() As Integer
  Dim ObjectIndices() As Integer
  Dim ObjectNames() As String

  EA_Panel.GetObjectList 1, NofObjects, ObjectTypes, ObjectIndices, ObjectNames

  For i = 0 To NofObjects - 1
  If ObjectTypes(i) = 2 Then 'All Real-time graphs and Post-process graphs on the
first panel.
    EA_Graph.PrintOut 1, ObjectNames(i) 'Print out the graph.
  End If
  Next
```

EA_Panel.GetObjectProperty

アナログメータまたはバー表示器オブジェクトの最小/最大軸の制限値を取得します。

```
  EA_Panel.GetObjectProperty(ByVal Panel As Variant, ByVal ObjectName As String,
ByVal PropID As Integer, PropVal As Variant)
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、*すべてのパネル*でオブジェクトを検索することを意味します。
ObjectName	アナログメータまたはバーインジケータオブジェクト
PropID	Flags = 0: アナログメーターまたはバー表示器オブジェクトの最小軸制限を取得。 Flags = 1: アナログメーターまたはバー表示器オブジェクトの最大軸制限を取得。
PropVal	表示器の最小/最大軸制限値。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはオブジェクトが存在しないなど。

EA_Panel.SetObjectPropertyも参照してください。

注記

- 複数のオブジェクトが同じ名前で、すべてのパネルを検索(*Panel* = 0)する場合、最初に見つかったオブジェクトを使用します。
- この関数はアナログメータとバー表示器オブジェクトにのみ適用されます。

EA_Panel.GetValue

オブジェクトの内容またはステータスを読み出します。データ型は、オブジェクトの種類によって異なります。

```
  EA_Panel.GetValue(ByVal Panel As Variant, ByVal ObjectName As String, Value As
Variant, Optional ByVal SubPartID As Integer, Optional ByVal Flags As Integer) As
Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、*すべて*のパネルでオブジェクトを検索することを意味します。パネルは通常のパネル、スコープパネル、およびフローティングパネルのいずれかになります。スコープおよびフローティングパネルは、常にパネル名から参照する必要があります!
ObjectName	オブジェクト名。スコープまたはフローティングパネルでは、グラフ、テーブルや表示オブジェクトを含む"layer"を指定します。
Value	オブジェクトの内容またはステータス:
	チェックボックス: チェックボックスが一つのみの選択オプション(デフォルト)の場合、このオプションのステータス(0 = チェックしていない、1 = チェックしている)を返します。 チェックボックスに複数の選択オプションがあり*ラジオボタン*として構成している場合、選択したオプションを返します。最初のオプションのインデックスは1です。 チェックボックスに複数の選択オプションがあり*チェックボックス*として構成している場合、選択したオプションのステータスはビットマスクとして返します。選択したオプションのビットが設定されます。最初のオプションは最下位ビット(bit 0)に相当します。
	LED: LEDのステータス(0 = オフ、1 = 警報、2 = 警告)
	リストボックス: 選択した項目のテキスト。
	テキストボックス: 入力フィールド内のテキスト。
	デジタル表示器および7セグ表示器: 表示値。
	アナログメータおよびバー表示器: ポインタ値。
	ノブおよびスライダ: ポインタ値。
	On/Offスイッチ: ONの場合Trueで、それ以外はFalse。
	スライドスイッチ: スイッチ位置(0、1、2....、一番下/左端の位置が0です)。
	7セグ時計: 日付/時刻またはストップウォッチの経過時間。
	テキスト: 現在表示されているテキスト。
SubPartID	オプションです。値ポインタのインデックス。最初の値ポインタのインデックスは0です。 これはアナログメータおよびバー表示器ににのみ適用され、オプションのパラメータFlagsは省略されるか、0になります。
Flags	オプションです。 Flags = 0: 現在のパネルまたはページにある場合は表示値を取得します。 Flags = 1: 現在のパネルまたはページにない場合は表示値を取得します。 アナログメータとバー表示器のみが使用します。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはオブジェクトが存在しないなど。

EA_Panel.SetValueも参照してください。

注記

- 複数のオブジェクトが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったオブジェクトを使用します。

- テーブルのセル内容を読み取るには、EA_Panel.GetCellを使用します。

例

```
  'Gets the index of the selected option of a checkbox whose operating mode is set
to RadioButton. In this example the 4th option is selected.
  Dim status As Long
  EA_Panel.GetValue(1, "CHECKBOX_1", status) 'status = 4

  'Gets the index of the selected option of a checkbox whose operating mode is set
to CheckBox. In this example the 4th option is selected.
  EA_Panel.GetValue(1, "CHECKBOX_2", status) 'status = 8 = 2^(4-1)
  'Gets the indices of the selected options of the checkbox. In this example the 3rd
and 4th options are selected.
  EA_Panel.GetValue(1, "CHECKBOX_2", status) 'status = 12 = 2^(3-1) + 2^(4-1)
```

EA_Panel.HideFloatingPanel

フローティングパネルを非表示にします。

```
EA_Panel.HideFloatingPanel(ByVal PanelName As Variant) As Long
```

引数	説明
PanelName	パネルの名前またはインデックス。最初のパネルのインデックスは1です。

EA_Panel.HideTableFocusRectangle

テーブルのアクティブセルのフォーカス枠を表示または非表示にします。

```
  EA_Panel.HideTableFocusRectangle(ByVal Panel As Variant, ByVal ObjectName As
String, ByVal HideState As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、すべてのパネルがテーブルを検索することを意味します。
ObjectName	テーブル名。
HideState	0 = フォーカス枠を表示 1 = フォーカス枠を非表示

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはテーブルが存在しないなど。

EA_Panel.InsertRow

テーブルに空の行を挿入します。

```
  EA_Panel.InsertRow(ByVal Panel As Variant, ByVal TableName As String, ByVal Row As
Long) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、すべてのパネルがテーブルを検索することを意味します。
TableName	テーブル名。
Row	挿入する行番号の位置。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはテーブルが存在しないなど。

注記

- テーブル内の行の合計数が一つ増加します。以降の行は一つ下に移動します。

EA_Panel.Load

指定した位置に、パネルまたは印刷ページを読み込みます。

```
EA_Panel.Load(ByVal PanelFileName As String, ByVal Panel As Variant) As Long
```

引数	説明
PanelFileName	パネルのファイル名(ファイル拡張子.PAN)。
Panel	パネルまたは印刷ページを読み込む、名前または位置。最初のパネルまたは印刷ページの位置は1です。 *Panel* = 0の場合、既存パネルの最後尾に新規パネルを作成します。

戻り値	説明
0	エラーなし。
-1	エラー。パネルが存在しない。

注記

- ファイル名にパスが含まれていない場合、最後に使用した保存/開くダイアログのパスまたは、Documents and Settings\Current userディレクトリの\HBM\catmanEasy/AP\Projectsを使用します。

例

```
Sub PrepareAnalysis()
  EA_Panel.Load "c:\catmanEasy/AP\PROJECTS\Analysis.PAN", 2
End Sub
```

EA_Panel.LoadObjectFromTemplate

指定したテンプレートファイルから、オブジェクト設定を読み込みます。

```
  EA_Panel.LoadObjectFromTemplate(ByVal Panel As Variant, ByVal ObjectName As String,
ByVal FileName As String) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。
ObjectName	オブジェクト名。 グラフ、テーブル、アナログメータ、またはバー表示器オブジェクトのみ利用可能。
FileName	テンプレートファイル名。

戻り値	説明
0	エラーなし。
-1	エラー。パネルが存在しない。

例

```
  Sub CloneMeter()
    EA_Panel.LoadObjectFromTemplate 1, "Meter_2" "C:\Users\catman\Documents\HBM\
catmanEasy\TEMPLATES\TMP1.Meter"
  End Sub
```

EA_Panel.LoadTable

EA_Panel.SaveTableで以前に保存したテーブルをファイルから読み込みます。この関数は、アクティブシートの内容だけでなく、セルの書式も読み込みます。

```
  EA_Panel.LoadTable(ByVal Panel As Variant, ByVal ObjectName As String, ByVal
FileName As String, ByVal Format As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。Panel = 0の場合、すべてのパネルのテーブルを検索することを意味します。
ObjectName	テーブル名。
FileName	ファイル名。
Format	形式: 0 = バイナリ 1 = テキスト(TAB区切り) 2 = Microsoft Excel(BIFF8)

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはテーブルが存在しないなど。

例

```
  'Load data from a tab delimited text file to the active table sheet.
  EA_Panel.LoadTable(1, "TABLE_1", "C:\HBM\catmanEasy\DATA\myTable.txt", TAB_TEXT)
```

EA_Panel.LockObject

指定した表示オブジェクトをロックまたはロック解除します。ロック状態ではオブジェクトの移動、サイズ変更、構成ダイアログでの修正が行えません。チャネルの割り当ても行えません。

```
  EA_Panel.LockObject(ByVal Panel As Variant, ByVal ObjectName as String, ByVal State
As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。Panel = 0の場合、すべてのパネルのテーブルを検索することを意味します。
ObjectName	オブジェクト名。
State	False = オブジェクトをロック解除。 True = オブジェクトをロック。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはオブジェクトが存在しないなど。

注記

- 複数のオブジェクトが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったオブジェクトを使用します。
- すべての表示オブジェクトをロックするには、EA_Panel.LockPanelsを使用します。

EA_Panel.LockPanels

パネル上のすべての表示オブジェクトをロック/ロック解除の設定/取得を行います。ロック状態では以下が行えません:

- 表示オブジェクトの生成、削除またはコピー
- 表示オブジェクトの移動やサイズ変更
- 任意のチャネル割り当て

```
Property EA_Panel.LockPanels
```

値	説明
False	パネルのロック解除。
True	パネルのロック。

注記

- *個別に*表示オブジェクトをロックするには、EA_Panel.LockObjectを使用します。

- EA_Panel.LockObjectは、GUIの設計/実行モードボタンと同じ効果があります。

EA_Panel.PanelCount (Read only)

現在のDAQまたは解析プロジェクト内の、すべてのパネル数を返します。これはVisualizationタブの通常のパネル、スコープパネルおよび印刷ページが含まれます。

```
Property EA_Panel.PanelCount
```

注記

- フローティングパネルは含まれません。

EA_Panel.PrintPage

パネルまたは印刷ページを印刷します。またEA_Panel.PrintScreenで、クリップボードにスクリーンショットをコピーすることもできます。

```
  EA_Panel.PrintPage(ByVal Panel As Variant, Optional ByVal ShowDialog as Integer)
As Long
```

引数	説明
Panel	パネル/印刷ページ名または位置。最初のパネルまたは印刷ページの位置は1です。パネルおよび印刷ページは連続的に番号付けされますが、印刷ページで番号付けを再開しません。
ShowDialog	印刷ダイアログの表示: 0 = 印刷ダイアログを表示せず、すぐに印刷(デフォルト) 1 = 印刷ダイアログの表示

戻り値	説明
0	エラーなし。
<0	エラー。ページが存在しないなど。

EA_Graph.Export、EA_Graph.PrintOutも参照してください。

注記

- 例えば複数のプリンタにアクセスでき、デフォルトのプリンタに印刷させたくない場合、印刷ダイアログを使用します。

例

```
Sub Print()
  'Print out page "PressureGraph"
  EA_Panel.PrintPage "PressureGraph"
End Sub
```

EA_Panel.PrintScreen

指定したパネルや印刷ページをクリップボードにコピー(スクリーンショット)します。省略した場合、現在選択しているパネルや印刷ページを使用します。

```
EA_Panel.PrintScreen(Optional ByVal Panel As Variant)
```

EA_Panel.PrintPageも参照してください。

引数	説明
Panel	パネル/印刷ページ名または位置。最初のパネルまたは印刷ページの位置は1です。Panel = 0は、現在選択しているパネルや印刷ページを使用することを意味します。

注記

- MS WordのApplicationオブジェクトを作成することにより、クリップボードの内容をWord文書に貼り付けることができます(下記の例を参照してください)。
- スクリーンショットには、リボンバーや構成ダイアログは含まれません。

例

```
Dim objWrd As Object
Dim newDoc As Object
Dim myRange As Object

EA_Panel.PrintScreen

Set objWrd = CreateObject("Word.Application")
objWrd.Visible = True

'Create an edit range and assign it to variable myRange:
Set newDoc = objWrd.Documents.Add
Set myRange = newDoc.Range


Add the clipboard contents behind the range:
myRange.Collapse wdCollapseEnd
myRange.Paste
```

EA_Panel.PrintTable

テーブルを印刷します。

```
  EA_Panel.PrintTable(ByVal Panel As Variant, ByVal ObjectName As String, ByVal
Orientation As Integer, Optional ByVal ShowDialog As Integer, Optional ByVal SheetNo
As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	テーブル名。
Orientation	形式: 0 = デフォルト設定 1 = 縦向き 2 = 横向き
ShowDialog	印刷ダイアログの表示: 0 = 印刷ダイアログを表示せず、すぐに印刷(デフォルト) 1 = 印刷ダイアログを表示
SheetNo	テーブルシートのインデックス: SheetNo > 0: SheetNoで指定したシートのみを印刷。 SheetNo <= 0: シートのインデックス順にテーブル内のすべてのシートを印刷。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはページが存在しないなど。

注記

- 例えば複数のプリンタにアクセスでき、デフォルトのプリンタに印刷させたくない場合、印刷ダイアログを使用します。

例

```
  Sub Print()
    EA_Panel.SetCell "PressureTable", "Table1", 2, 2, 0.03
    EA_Panel.PrintTable "PressureTable", "Table1", 0, 1, 1 'Show print dialog, and
print only the first sheet.
  End Sub
```

EA_Panel.RecalcTable

数式で参照するセル内容が変更された場合、テーブル内の数式を再計算します。

```
  EA_Panel.RecalcTable(ByVal Panel As Variant, ByVal ObjectName As String, Optional
ByVal Sheet As Integer, Optional ByVal Row As Long, Optional ByVal Col As Long) As
Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。
ObjectName	オブジェクト名。
Sheet	任意です。テーブルのシート番号。最初のシートは1です。
Row	任意です。セルの行番号。最初の行番号は1です。
Col	任意です。セルの列番号。最初の列番号は1です。

戻り値	説明
0	エラーなし。
<0	エラー。パネルが存在しないなど。

注記

- *Sheet*を指定しない(0または省略)場合、テーブル全体のすべてのシートのすべての式を再計算します。
- 個々のセルの数式の再計算は、シート番号、セルの行および列番号を指定する必要があります。

EA_Panel.RefreshAll

すべてのパネル/ページ上のすべてのオブジェクト(グラフ、テーブル、テキスト)を更新します。""オブジェクトの"データソース"を変更した場合、これが必要です。例えばチャネルがグラフやテーブルセルを参照して新規データを受け取ったり、演算チャネルを再計算した場合などです。

```
EA_Panel.RefreshAll()
```

戻り値	なし

注記

- 解析モードで、テキストやテーブルのセルにチャネル属性への参照(最大値など)が含まれていたり、チャネルに多くのサンプルが含まれている(> 1000000)と、更新に時間がかかることがあります。オブジェクトを個別に更新するEA_Panel.RefreshTextやEA_Panel.RefreshTableを使用してください。

EA_Panel.RefreshTable

テーブルを再描画し、セルにリンクしているすべてのデータソースを更新します。解析プロジェクト内のデータが変更された場合(例えばテストの読み込み後、演算の実行など)、このメソッドを使用します。

```
EA_Panel.RefreshTable(ByVal Panel As Variant, ByVal ObjectName As String) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	テーブル名。

戻り値	説明
0	エラーなし。
<0	エラー。パネルが存在しないなど。

例

```
Sub Print()
  EA_Panel.SetCell "PressureTable", "Table1", 2, 2, 0.03
  EA_Panel.RefreshTable "PressureTable", "Table1"
  EA_Panel.PrintTable "PressureTable", "Table1", 0
End Sub
```

EA_Panel.RefreshText

テキストオブジェクトを再描画し、リンクしているすべてのテキストソースを更新します。

```
EA_Panel.RefreshText(ByVal Panel As Variant, ByVal ObjectName As String) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。
ObjectName	テキストオブジェクト名。

戻り値	説明
0	エラーなし。
<0	エラー。パネルが存在しないなど。

EA_Panel.ReleaseDirectControl

EA_Panel.GetDirectControlで、ローレベルプログラミングのために要求していた、可視化パネルオブジェクトの制御をcatmanに戻します。

```
   EA_Panel.ReleaseDirectControl(ByVal ObjectName As String, Optional ByVal Panel As
Variant) As Long
```

引数	説明
ObjectName	オブジェクト名。スコープまたはフローティングパネルでは、グラフ、テーブルや表示オブジェクトを含む"layer"を指定します。
Panel	パネル名または位置。最初のパネルは1です。省略した場合、すべての通常のパネルオブジェクトを検索します。パネルは通常のパネル、スコープパネルおよびフローティングパネルのいずれかになります。 スコープやフローティングパネルの場合、Panelパラメータは必須です。

戻り値	説明
0	エラーなし。
<0	エラー。オブジェクトが存在しないなど。

注記

- EA.GetObjectおよびローレベルのグラフプログラミングも参照してください。
- このメソッドは現在、グラフ、表、アナログメータ、バー、数値、LED表示器、およびテキストオブジェクトをサポートしています。

EA_Panel.RemoveListboxItem

リストボックスの項目を削除します。

```
  EA_Panel.RemoveListboxItem(ByVal Panel As Variant, ByVal ObjectName As String,
ByVal Index As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	リストボックス名。
Index	リストボックスの項目を削除する指標。最初の項目の位置は1です。

戻り値	説明
0	エラーなし。
<0	エラー。パネルやリストボックスが存在しないなど。

注記

- 単一コマンドでリストを削除するにはEA_Panel.ClearListboxを使用します。
- 複数のリストボックスが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったリストボックスを使用します。

EA_Panel.RemoveRow

テーブルの行を削除します。

```
  EA_Panel.RemoveRow(ByVal Panel As Variant, ByVal TableName As String, ByVal Row As
Long) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
TableName	テーブル名。
Row	行番号。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはテーブルが存在しないなど。

注記

- テーブル内の行の合計数は変わりません。次の行が一つ上に移動し、新しい空の行をテーブルの最後尾に追加します。

EA_Panel.Rename

パネルまたは印刷ページのタイトルを変更します。

```
EA_Panel.Rename(ByVal Panel As Variant, ByVal NewPanelTitle As String) As Long
```

引数	説明
Panel	パネル/印刷ページ名または位置。最初のパネルまたは印刷ページの位置は1です。
NewPanelTitle	パネルまたは印刷ページの新規タイトル。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはページが存在しないなど。

EA_Panel.ResetStatistics

デジタル表示器、メータ、バー表示器およびテーブルの統計(最小/最大)をリセットします。

```
  EA_Panel.ResetStatistics(ByVal Panel As Variant, ByVal ObjectName As String) As
Long
```

引数	説明
Panel	オブジェクトが属するパネル(インデックスまたは名前の何れか)を示します。
ObjectName	表示器オブジェクトの名前。

戻り値	説明
0	エラーなし。
<0	エラー。オブジェクトが存在しないなど。

注記

- パネル上のすべての統計情報をリセットするには、EA_DAQ.ResetAllStatisticsを使用します。

EA_Panel.Save

パネルまたは印刷ページをファイルに保存します。

```
EA_Panel.Load(ByVal PanelFileName As String, ByVal Panel As Variant) As Long
```

引数	説明
PanelFileName	パネルのファイル名(拡張子.PAN)。
Panel	パネル/印刷ページ名または位置。最初のパネルまたは印刷ページの位置は1です。

戻り値	説明
0	エラーなし。
<0	エラー。パネルが存在しないなど。

注記

- ファイル名にパスが含まれていない場合は、**Save/Loadパネル**ダイアログで最後に使用したパス、またはDocuments and Settings\Current userの\HBM\catmanEasy/AP\Projectsディレクトリのいずれかを使用します。

EA_Panel.SavePanelToJPEG

指定したパネルや印刷ページをJPEG画像として保存します。

```
  EA_Panel.SavePanelToJPEG(ByVal Panel As Variant, ByVal FileName As String, Optional
ByVal Quality as Byte)
```

引数	説明
Panel	パネル/印刷ページ名または位置。最初のパネルまたは印刷ページの位置は1です。
FileName	JPEG画像のファイル名。
Quality	オプションです。画像の圧縮率を指定します。可能な値の範囲: 0～100 (0 = 最も高い圧縮率、最も低い画質、100 = 最も低い圧縮率、最も高い画質)。指定しない場合は、値50が使用します。

注記

- 画像には、リボンバーや構成ダイアログは含まれません。
- 画質が高くなると、ファイル容量が大きくなります。

EA_Panel.SaveTable

テーブルをファイルに保存します。この関数は、指定したシートの内容だけでなく、セルの書式も保存します。

```
  EA_Panel.SaveTable(ByVal Panel As Variant, ByVal ObjectName As String, ByVal
FileName As String, ByVal Format As Integer, Optional ByVal Sheet As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	テーブル名。
FileName	ファイル名。
Format	形式: 0 = バイナリ 1 = テキスト(TAB区切り) 2 = Microsoft Excel(BIFF8) 3 = JPEG画像
Sheet	任意です。テーブルのシートインデックス。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはテーブルが存在しないなど。

注記

- 保存したテーブルの読み込みはEA_Panel.LoadTableを使用します。

例

```
  'Save the active table sheet as a tab delimited text file.
  EA_Panel.SaveTable(1, "TABLE_1", "C:\HBM\catmanEasy\DATA\myTable.txt", TAB_TEXT)
```

EA_Panel.SetAccessMode

単一の入力オブジェクトを有効/無効化または、表示/非表示します。サポートしているオブジェクトは、ボタン、テキストボックス、リストボックス、チェックボックス、スライダ、ノブ、スライドスイッチ、およびOn/Offスイッチです。

```
  EA_Panel.SetAccessMode(ByVal Panel As Variant, ByVal ObjectName As String, ByVal
Mode As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	オブジェクト名。
Mode	1 = 入力許可 2 = 入力無効(グレーアウト) 3 = オブジェクト表示 4 = オブジェクト非表示

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはオブジェクトが存在しないなど。

注記

- 複数のオブジェクトが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったオブジェクトを使用します。

- この関数は、他のパネルオブジェクト(グラフ、デジタル表示器など)には影響しません。

EA_Panel.SetAlarmLevel

アナログメータやバーオブジェクトの、警報および警告レベルを設定します。

```
  EA_Panel.SetAlarmLevel(ByVal Panel As Variant, ByVal ObjectName As String, ByVal
LevelType As Integer, ByVal Value As Single) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	オブジェクト名。
LevelType	変更するレベルを定義します: 0 = ローレベル警報 1 = ローレベル警告 2 = ハイレベル警告 3 = ハイレベル警報
Value	レベル値。

戻り値	説明
0	エラーなし。
<0	エラー。パネルが存在しない、オブジェクトの型が違うなど。

注記

- レベル色の変更にはEA_Panel.SetColorを使用します。
- 複数のオブジェクトが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったオブジェクトを使用します。

例

```
  EA_Panel.SetAlarmLevel(1, "METER_1", LEVEL_HIGHWARN, 5)
```

EA_Panel.SetBackgroundImage

パネルオブジェクトの背景画像を設定します。

```
  EA_Panel.SetBackgroundImage(ByVal Panel As Variant, ByVal ObjectName As String,
ByVal ImgFile As String) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	オブジェクト名。このメソッドは以下に適用します: - テキストオブジェクト - ボタンオブジェクト - デジタル表示器オブジェクト - LEDオブジェクト
ImgFile	画像ファイル。ファイル名にパスが含まれていない場合、またはパスの下で見つけられない場合、catmanはデフォルトの画像フォルダでファイルを検索します。有効なファイル形式は以下の通りです。 - ビットマップ(.BMP) - JPEG(.JPG) - Portable Network Graphics(.PNG) - メタファイル(.EMF) - アイコンファイル(.ICO) - GIF(.GIF)

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはオブジェクトが存在しないなど。

注記

- 複数のオブジェクトが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったオブジェクトを使用します。

例

```
  EA_Panel.SetBackgroundImage 1, "TEXT_1", "C:\HBM\catmanEasy\DATA\MyPicture.jpg"
```

EA_Panel.SetCaption

オブジェクトのキャプションを設定します。サポートしているオブジェクトは、デジタル表示器(タイトル)、LED、チェックボックス、ボタン、およびOn/Offスイッチです。

チェックボックスに複数のオプションが含まれている場合、個々の項目はCRLF (vbCrLf)で区切る必要があります。

```
  EA_Panel.SetCaption(ByVal Panel As Variant, ByVal ObjectName As String, ByVal
Caption As String, Optional ByVal SubPartID As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	オブジェクト名。
Caption	オブジェクトの新規キャプション。
SubPartID	任意です。キャプションテキストのインデックス。最初のキャプションテキストのインデックスは0です。これはアナログメータおよびバー表示器ににのみ、適用されます。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはオブジェクトが存在しないなど。

注記

- 複数のオブジェクトが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったオブジェクトを使用します。

例

```
  'Sets the caption of the 1st checkbox option to "Execute zero balance" and caption
of 2nd to "Autocal ON".
  Dim Caption As String
  Caption = "Execute zero balance" & vbCrLf & "Autocal ON"
  EA_Panel.SetCaption(1, "CHECKBOX_1", Caption)
```

EA_Panel.SetCell

これはテーブルオブジェクトに対処する最も汎用的な関数の一つです。個々のセルに情報を書き込みます。*DataType*に応じて、異なるタイプになる可能性があり、セルのタイプにも依存する場合があります(例えば、編集可能なテキストであればテキストとして受け入れ、数値のセルは数値として受け入れます)。

このメソッドは、現在のアクティブシート上のセルに適用されることに注意してください。シートをアクティブにするにはEA_Panel.SetTablePropertyプロパティを使用します。一枚のシートのみのテーブルの場合、アクティブにする必要はありません。

```
  EA_Panel.SetCell(ByVal Panel As Variant, ByVal ObjectName As String, ByVal Col As
Long, ByVal Row As Long, ByVal Value As Variant, Optional ByVal DataType As Integer)
As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	テーブル名。
Col	セルの列番号。列の最初のインデックスは1です。
Row	セルの行番号。行の最初のインデックスは1です。
Value	設定する情報。セルの種類だけでなく、*DataType*にも依存します。
DataType	TYPE_CONTENT (0) テキストセル: セル内のテキストを設定します 数値セル: セルの値を設定します コンボボックスセル: ";"で区切った項目のリストを設定します 他のすべてのセル: 適用されません TYPE_SELECTION (1) チェックボックスセル: チェックボックスの状態を設定します(0=チェックなし、1=チェックあり) コンボボックスセル: 選択したアイテムのインデックスを設定します(最初のインデックスは0) 他のすべてのセル: 適用されません TYPE_CAPTION (2) ボタンおよびチェックボックスセル: キャプションを設定します 他のすべてのセル: 適用されません TYPE_FORECOLOR (3) テキストの色を設定します。 ピクチャセルセルには適用されません。 TYPE_BACKCOLOR (4) セルの背景色を設定します。 画像セルには適用されません。 TYPE_PICTURE (5) セルの画像やボタンの画像を設定します。*Value*は画像ファイルのファイル名を含める必要があります。 指定したパスにファイルが見つからない場合、デフォルトのIMAGE フォルダを検索します。 他のすべてのセル: 適用されません

DataType	TYPE_PICTURESOURCE (6) ピクチャセルにグラフをリンクし、セルにグラフの画像を表示させます。*Value*はグラフ名を含める必要があります。 解析モードでのみ可能です! 画像は、すぐにはレンダリングされないことに注意してください。EA_Panel.RefreshTableを使用してテーブルを再描画してください。 他のすべてのセル: 適用されません TYPE_DATASOURCE (7) テキストセルと数値セル: チャネルから取得した情報(例えば最大値)にセルをリンクします。*Value*はデータソースの説明文字列が含まれます。セルのデータソース表記に、データソースを指定する方法を説明しています。 解析モードでのみ可能です! テーブルは、自動的にデータソースを反映しないことに注意してください。EA_Panel.RefreshTableを使用してテーブルを再描画してください。 TYPE_CELLTYPE (8) セルの種類を設定: 1 = 動的テキスト 2 = 動的数値 3 = 静的テキスト 4 = 静的数値 5 = ボタン 6 = 画像 7= チェックボックス 8 = コンボボックス TYPE_NUMPRECISION (9) 数値セル: 数値セルの小数点右側に割り当てられた桁数を設定します。 他のすべてのセル: 適用されません。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはテーブルが存在しないなど。

EA_Panel.GetCellも参照してください。

注記

- 計測の実行中は、リアルタイム表示用に割り当てられたチャネルを持つセルを変更しないでください。これらのセルはcatmanEasy/APで継続的にリアルタイムで更新するため、登録はすぐに上書きされ、表示されなくなります。
- セル範囲に影響を与える(色や画像など)プロパティにアクセスするため、マージしたセル範囲の左上を指定します。
- 他のオブジェクトの内容を変更するにはEA_Panel.SetValueを使用します。
- この関数は通常、ユーザインタラクティブオブジェクトを情報を交換するために適用し、*SetValue*、*SetColor*、*SetCaption*などはテーブルのセルに適用されない場合があります。

例

```
  Sub Energy()
    'Reads the current measurement of a force channel and a displacement channel,
computes the energy from them and enters it into the second column and third row of
the Table "Result" on the first panel.
    Dim MV1 As Double
    Dim MV2 As Double
    Dim Result As Double
    EA_DAQ.CurrentReading "Force", MV1
    EA_DAQ.CurrentReading "Displacement", MV2
    Result = MV1 * MV2
    EA_Panel.SetCell 1, "Result", 2, 3, Result, TYPE_CONTENT
  End Sub
```

EA_Panel.SetColor

オブジェクトの前景色または背景色を設定します。

```
  EA_Panel.SetColor(ByVal Panel As Variant, ByVal ObjName As String, ByVal ColorType
As Integer, ByVal Color As Long, Optional ByVal SubPartID As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。Panel = 0の場合、*すべてのパネル*を検索することを意味します。
ObjName	オブジェクト名。
ColorType	変更する色(オブジェクトの種類に応じて)を定義します:
	デジタル表示器: 0 = テキスト色、タイトル 1 = 背景、タイトル 2 = 文字色、計測値 3 = 背景、計測値 4 = グラデーション開始(グラデーション色を設定する前に、パラメータ *SubPartID*を使用してグラデーションのスタイルを指定) 5 = グラデーション停止(グラデーション色を設定する前に、パラメータ *SubPartID*を使用してグラデーションのスタイルを指定)
	バー表示器、アナログメータ: 0 = テキスト色、タイトル 1 = 背景色、外側領域 2 = 文字色、スケール 3 = 背景色、外側領域 4 = ポインタ色 5 = ポインタ塗りつぶし色 6 = ローレベル警報色 7 = ローレベル警告色 8 = ハイレベル警告色 9 = ハイレベル警報色
	ボタン、テキストボックス、チェックボックス、リストボックス: 0 = テキスト色 1 = 背景 ボタンの背景には特殊な色値を使用しています(*Color*を参照)
	LED、テキスト: 0 = テキスト色 1 = 背景 2 = 文字色、計測値 4 = グラデーション開始(グラデーション色を設定する前に、パラメータ *SubPartID*を使用してグラデーションのスタイルを指定) 5 = グラデーション停止(グラデーション色を設定する前に、パラメータ *SubPartID*を使用してグラデーションのスタイルを指定)

<table>
<tr><td>Color</td><td>RGB色相。
色相はVBAで定義された定数のいずれか(vbBlack、vbWhite、vbBlue、vbRed、vbGreen、vbCyan、vbMagenta、vbYellow)、または赤、緑および青の組み合わせRGB(R,G,B)になります。

<table>
<tr><td colspan="2">ボタンの背景:</td></tr>
<tr><td>0 = 黒色</td><td>8 = 灰色</td></tr>
<tr><td>1 = 青色</td><td>9 = 水色</td></tr>
<tr><td>2 = 緑色</td><td>10 = 薄緑色</td></tr>
<tr><td>3 = 青緑色(シアン)</td><td>11 = 薄水色</td></tr>
<tr><td>4 = 赤色</td><td>12 = 栗毛色</td></tr>
<tr><td>5 = 赤紫色(マゼンタ)</td><td>13 = 桃色</td></tr>
<tr><td>6 = 黄色</td><td>14 = 淡黄色</td></tr>
<tr><td>7 = 白色</td><td>15 = 明白色</td></tr>
</table>
</td></tr>
<tr><td>SubPartID</td><td>任意です。アナログメータおよびバー表示器にのみ適用されます。
ColorType = 0: キャプションテキストのインデックス
ColorType = 2: スケールテキストのインデックス
ColorType = 4 or 5: ポインタ値のインデックス

または、背景色のグラデーションスタイルを指定するため、テキスト、LED、および数字表示器に使用します。
LEDと数字表示器のグラデーション スタイルのインデックス:
0 = グラデーションなし
1 = 左右
2 = 上下
3 = 中央揃え
4 = 左斜め
5 = 右斜め
6 = 水平方向の中央揃え
7 = 垂直方向の中央揃え

テキストのグラデーション スタイルのインデックス:
0 = グラデーションなし
1 = 上から下
2 = 下から上
3 = 左から右
4 = 右から左
5 = 左上から右下
6 = 右下から左上
7 = 左下から右上
8 = 右上から左下
9 = 矩形外方向
10 = 矩形内方向
11 = 円形外方向
12 = 円形内方向</td></tr>
</table>

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはオブジェクトが存在しないなど。

注記

- 複数のオブジェクトが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったオブジェクトを使用します。

- グラフの色を変更するにはEA_Graph.SetPropertyを使用します。

例

```
  Sub SetDigitColors()
    'No gradient style is used.
    EA_Panel.SetColor 1, "DIGIT_1", COLOR_CAPTION, vbGreen 'Color of caption.
    EA_Panel.SetColor 1, "DIGIT_1", COLOR_BACKGROUND, vbBlack 'Color of caption
backgroud.
    EA_Panel.SetColor 1, "DIGIT_1", COLOR_VALUE, vbCyan 'Color of value.
    EA_Panel.SetColor 1, "DIGIT_1", COLOR_VALUEBACKGROUND, vbMagenta 'Color of value
backgroud.

  End Sub

  Sub SetDigitGradientColors()
    'Gradient style Left-right is used.
    EA_Panel.SetColor 1, "DIGIT_1", COLOR_GRADIENTSTART, vbRed, 1 'Gradient start
color.
    EA_Panel.SetColor 1, "DIGIT_1", COLOR_GRADIENTSTOP, vbYellow, 1 'Gradient stop
color.
  End Sub
```

EA_Panel.SetColWidth

テーブルの幅に関連して、列の幅を設定します。

```
  EA_Panel.SetColWidth(ByVal Panel As Variant, ByVal ObjectName As String, ByVal Col
As Long, ByVal Width As Double) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	テーブル名。
Col	セルの列番号。列の最初のインデックスは1です。
Width	表の幅と関連した列幅(1 = テーブルの幅、0.5 = 表の幅の半分、など)。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはテーブルが存在しないなど。

EA_Panel.SetFloatingPanelURL

フローティングパネル内の、WebBrowser項目のURLを設定します。

```
  EA_Panel.SetFloatingPanelURL(ByVal PanelName As Variant, ByVal LayerName As String,
ByVal URL As String) As Long
```

引数	説明
PanelName	パネルの名前または指標。最初のパネルの指標は1です。
LayerName	WebBrowserコントロールを含むレイヤ名。
URL	URL文字列(例えばwww.hbm.com)または、ブラウザで表示可能な任意のファイル。

EA_Panel.SetListboxItems

リストボックス完全な内容を設定します。

```
  EA_Panel.SetListboxItems(ByVal Panel As Variant, ByVal ObjectName As String, ByVal
ListItems As String) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	リストボックス名。
ListItems	リストボックスの新しい内容。個々の項目はCRLF (vbCrLf)で区切る必要があります。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはリストボックスが存在しないなど。

注記

- リストボックスの以前の内容は、この操作で上書きされます。
- 単一の項目を追加するにはEA_Panel.AddListboxItemを使用します。
- 複数のリストボックスが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったリストボックスを使用します。

例

```
  'Sets the two listbox entries with text "ListItem1" and "ListItem2" in this
example.
  EA_Panel.SetListboxItems(1, "COMBOBOX_1", "ListItem1" & vbCrLf & "ListItem2")
```

EA_Panel.SetListboxSelIndex

リストボックスの選択項目を指定した位置に設定します。

```
  EA_Panel.SetListboxSelIndex(ByVal Panel As Variant, ByVal ObjectName As String,
ByVal Index As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	リストボックス名。
Index	選択位置。リストの最初の項目は1です。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはリストボックスが存在しないなど。

注記

- 選択した項目の値テキストを検索するにはEA_Panel.GetValueを使用します。
- 複数のリストボックスが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったリストボックスを使用します。

EA_Panel.SetObjectFocus

指定したオブジェクトにフォーカスを設定します。

```
EA_Panel.SetObjectFocus(ByVal Panel As Variant, ByVal ObjectName As String) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。
ObjectName	オブジェクト名。サポートしているオブジェクトタイプは、入力ボックス、コンボボックス、チェックボックス/ラジオボタン、スライドスイッチです。

戻り値	説明
0	エラーなし。
<0	エラー。パネルが存在しないなど。

注記

- この関数は、ユーザの入力を要求するオブジェクトにフォーカスを設定することを目的としています。

EA_Panel.SetObjectProperty

アナログメータまたはバー表示器オブジェクトの最小/最大軸制限を設定します。

```
  EA_Panel.SetObjectProperty(ByVal Panel As Variant, ByVal ObjectName As String,
ByVal PropID As Integer, ByVal PropVal As Variant)
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	アナログメータまたはバー表示器オブジェクト名。
PropID	Flags = 0: アナログメータまたはバー表示器オブジェクトの最小軸制限を設定。 Flags = 1: アナログメータまたはバー表示器オブジェクトの最大軸制限を設定。
PropVal	アナログメータまたはバー表示器の最小/最大軸制限値。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはオブジェクトが存在しないなど。

EA_Panel.GetObjectPropertyも参照してください。

注記

- 複数のオブジェクトが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったオブジェクトを使用します。
- この関数はアナログ メータとバー表示器オブジェクトにのみ適用されます。

EA_Panel.SetObjectState

可視化オブジェクトの表示や有効状態を設定します。

```
  EA_Panel.SetObjectState(ByVal Panel As Variant, ByVal ObjectName As String, ByVal
Mode As eObjectStateType, ByVal Value As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	可視化オブジェクト名。
Mode	0 = 表示状態 1 = 有効状態
Value	Mode = 0: 表示状態 0 = 非表示 1 = 表示 Mode = 1: 有効状態 0 = 無効 1 = 有効

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはオブジェクトが存在しないなど。

注記

- 複数のオブジェクトが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったオブジェクトを使用します。

例

```
  'Disable the CheckBox on the first panel.
  EA_Panel.SetObjectState 1, "CHECKBOX_1", 1, 0
  'Keep this CheckBox visible.
  EA_Panel.SetObjectState 1, "CHECKBOX_1", 0, 1
```

EA_Panel.SetRTDisplayChannel

アナログメータ、バー、デジタル表示器、LED、テーブルに表示するチャネルの割り当てや、グラフに表示するチャネルを追加します。このメソッドはDAQモードでのみ使用可能です(DAQモード = リアルタイムモードのため、関数名に"RT"が付いています)。解析モードでは、EA_Test.PlotYT、EA_Test.PlotYXまたはEA_Panel.SetCellを使用します。

```
  EA_Panel.SetRTDisplayChannel(ByVal Panel As Variant, ByVal ObjectName As String,
ByVal Channel As Variant, Optional ByVal Location As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
ObjectName	オブジェクト名。
Channel	チャネル名または位置。
Location	任意です。 グラフ: y軸レイヤを示すチャネルを割り当てます: -1 = x軸 0 = 最初のレイヤ(デフォルト) 1 = 二番目のレイヤ 2 = 三番目のレイヤ 3 = 右軸を基準にした最初のレイヤ 4 = 右軸を基準にした二番目のレイヤ 5 = 右軸を基準にした三番目のレイヤ 6 - 14 = 第4レイヤから第12レイヤ 15 - 23 = 右軸を基準にした第4レイヤから第12レイヤ ⚠ グラフは適切なレイヤ数のために、対話的に設定する必要があります: EasySciptは現在、グラフのレイヤ構成をサポートしていません。 テーブル: チャンネルを表示する行を指定します。空または0の場合、チャネルはアクティブな行に割り当てられます。 極線図: -1 = 角度軸 0 = 半径軸

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはオブジェクトが存在しないなど。

EA_Panel.SetValueも参照してください。

注記

- 複数のオブジェクトが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったオブジェクトを使用します。

EA_Panel.SetState

パネル選択タブの表示や有効状態を設定します(ユーザがパネルを選択できるようにするかを決定します)。

```
  EA_Panel.SetState(ByVal Panel As Variant, ByVal Mode As Integer, ByVal State As
Integer) As Long
```

引数	説明
Panel	パネル名または位置。
Mode	0 = 表示状態 1 = 有効状態
State	0 = 非表示または無効 1 = 表示または有効

戻り値	説明
0	エラーなし。
<0	エラー。パネルが存在しないなど。

EA_Panel.SetTableProperty

指定したTableプロパティの値を設定します。テーブル設定のいくつかは解析モードのみ、またはDAQモードのみに関連するものがあります。

個々のセルのアクセスにはEA_Panel.SetCellおよびEA_Panel.GetCellメソッドの使用を推奨しています。

対話型モードでの、Tableの**Configure**ダイアログで行う設定に相当します。

```
EA_Panel.SetTableProperty(ByVal Panel As Variant, ByVal ObjectName As String, ByVal Col As Long, ByVal Row As Long, ByVal PropIndex As Integer, ByVal PropVal As Variant) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、*すべての*パネルを検索することを意味します。
ObjectName	テーブル名。
Col	変更を適用するテーブルの列。この設定はPropIndexが6、9、10、11および12にのみ適用されます。Col = -1の場合、*すべての列*に影響を与えます。
Row	変更を適用するテーブルの行。この設定はPropIndexが9、10、11および12にのみ適用されます。Row = -1の場合、*すべての行*に影響を与えます。
PropIndex	Tableプロパティのインデックス: 1 = 解析モードのTableのみ、ヘッダ列の表示(ProVal: 0 = 表示、1 = 非表示) 2 = ヘッダ行の表示(ProVal: 0 = 表示、1 = 非表示) 3 = 水平スクロールバーの表示(ProVal: 0 = 表示、1 = 非表示) 4 = 垂直スクロールバーの表示(ProVal: 0 = 表示、1 = 非表示) 5 = 解析モードのTableのみ、Tableの罫線の設定(PropVal: 0 = 水平および垂直罫線、1 = 水平罫線のみ、2 = 垂直罫線のみ、3 = 罫線なし)、 6 = ヘッダ列のテキスト 7 = DAQモードのTableのみ、min/max列の表示(ProVal: 0 = 表示、1 = 非表示)、 8 = モードのTableのみ、limit monitoring列の表示(ProVal: 0 = 表示、1 = 非表示) 9 = セルのテキスト色を設定 10 = セルの背景色を設定 11 = 解析モードのTableのみ、セルの種類を設定(PropVal: 1 = 編集可能テキスト、2 = 数値、3 = 編集不可テキスト) 12 = 解析モードのTableのみ、数値セルの小数点以下の桁数(PropVal: 0から最大6) 13 = 指定した操作のためのシート番号。最初のシート番号は1です。Tableに複数のシートが含まれている場合にみ、この引数が必要です。 14 = アクティブシート、すなわち、表示するシート。最初のシート番号は1です。Tableに複数のシートが含まれている場合にみ、この引数が必要です。
PropVal	Tableのプロパティ値。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはTableが存在しないなど。

注記

- EA_Panel.SetCellも参照してください。

EA_Panel.SetValue

オブジェクトの内容または状態を設定します。データ型は、オブジェクトの種類によって異なります。

```
  EA_Panel.SetValue(ByVal Panel As Variant, ByVal ObjectName as String, ByVal
NewValue As Variant, Optional ByVal SubPartID As Integer) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルは1です。Panel = 0の場合、*すべての*パネルでオブジェクトを検索することを意味します。パネルは通常のパネル、スコープパネル、およびフローティングパネルのいずれかになります。スコープおよびフローティングパネルは、常にパネル名から参照する必要があります!
ObjectName	オブジェクト名。スコープまたはフローティングパネルでは、グラフ、テーブルや表示オブジェクトを含む"layer"を指定します。
Value	オブジェクトの新規内容または状態。これは、オブジェクトの型により異なります。
	Checkbox: チェックボックスが一つのみの選択オプション(デフォルト)の場合、このオプションのステータス(0 = チェックしていない、1 = チェックしている)を設定します。 チェックボックスに複数の選択オプションがあり*ラジオボタン*として構成している場合、選択したオプションを設定します。最初のオプションのインデックスは1です。 チェックボックスに複数の選択オプションがあり*チェックボックス*として構成している場合、選択したオプションのステータスはビットマスクとして返します。選択したオプションのビットが設定されます。最初のオプションは最下位ビット(bit 0)に相当します。
	デジタル表示器および7セグ表示器: 表示値。
	LED: LEDの状態(0 = オフ、1 = 警報、2 = 警告)
	リストボックス: 表示するテキスト
	テキストボックス: テキストボックスに表示するテキスト。
	テキスト: 表示するテキスト。テキストはCRLF文字を含めることができます。
	アナログメータおよびバー表示器: ポインタ値。
	ノブおよびスライダ: ポインタ値。
	On/Offスイッチ: ONの場合Trueで、それ以外はFalse。
	スライドスイッチ: スイッチ位置(0、1、2....、一番下/左端の位置が0です)。
	7セグ時計: 日付/時刻(例えば"Now")またはストップウォッチの経過時間。
SubPartID	オプションです。値ポインタのインデックス。最初の値ポインタのインデックスは0です。 これはアナログメータおよびバー表示器ににのみ、適用されます。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはオブジェクトが存在しないなど。

EA_Panel.GetValueも参照してください。

注記

- 複数のオブジェクトが同じ名前で、すべてのパネルを検索(Panel = 0)する場合、最初に見つかったオブジェクトを使用します。
- ボタンやチェックボックスのキャプションを設定するにはEA_Panel.SetCaptionを使用します。
- テーブルのセル内容を変更するには、EA_Panel.SetCellを使用します。

例

```
  'Selects the 4th option of a checkbox whose operating mode is set to RadioButton
and which has at least 4 options in this example.
  EA_Panel.SetValue(1, "CHECKBOX_1", 4) 'NewValue = 4

  'Selects the 4th option of a checkbox whose operating mode is set to CheckBox and
which has at least 4 options in this example.
  EA_Panel.SetValue(1, "CHECKBOX_2", 8) 'NewValue = 2^(4-1) = 8
  'Selects both the 3rd and 4th options of the checkbox.
  EA_Panel.SetValue(1, "CHECKBOX_2", 12) 'NewValue = 2^(3-1) + 2^(4-1) = 12
```

EA_Panel.SetXAxisChannel

DAQモードで、グラフのx軸にプロットを表示するチャネルを割り当てます。

```
  EA_Panel.SetXAxisChannel(ByVal Panel As Variant, ByVal GraphName As String, ByVal
XChannel As Variant) As Long
```

引数	説明
Panel	パネル名または位置。最初のパネルの位置は1です。Panel = 0の場合、すべてのパネルを検索することを意味します。
GraphName	グラフ名。
XChannel	プロットを追加するチャネル名または位置。

戻り値	説明
0	エラーなし。
<0	エラー。パネル、グラフまたはチャネルが存在しないなど。

注記

- このメソッドはDAQモーでのみ使用します。解析モードでは、EA_Testを使用します。
- 現在、個々のプロットにxチャネルを割り当てることはできません。

EA_Panel.ShowFloatingPanel

フローティングパネルを表示(見えるように)します。

```
  EA_Panel.ShowFloatingPanel(ByVal PanelName As Variant, Optional ByVal ConfigTab As
Integer, Optional ByVal x As Long, Optional ByVal y As Long) As Long
```

引数	説明
PanelName	パネル名または位置。最初のパネルの位置は1です。
ConfigTab	1に設定すると、構成ウィンドウも表示します。
x,y	ウィンドウ左上からのTwip単位の位置。省略した場合、ウィンドウは最後の位置を保持します。-1,-1に設定した場合、ウィンドウは画面の中央に配置されます。

戻り値	説明
0	エラーなし。
<0	エラー。フローティングパネルが存在しないなど。

EA_Panel.ShowPanel

パネルや印刷ページを表示します。

```
EA_Panel.ShowPanel(ByVal Panel As Variant) As Long
```

引数	説明
Panel	パネル/印刷ページ名または位置。最初のパネルまたは印刷ページの位置は1です。パネルおよび印刷ページは連続的に番号付けされますが、印刷ページで番号付けを再開しません。

戻り値	説明
0	エラーなし。
<0	エラー。パネルが存在しないなど。

例

```
Sub Monitor()
  Dim MV
  EA_DAQ.CurrentReading "Valve pressure", MV
  If MV > 50 Then
    EA_Panel.ShowPanel "Panel 5"
  End If
End Sub
```

EA_Panel.Unload

パネルまたは印刷ページを削除します。

```
EA_Panel.Unload(ByVal Panel As Variant) As Long
```

引数	説明
Panel	パネル/印刷ページ名または位置。最初のパネルまたは印刷ページの位置は1です。

戻り値	説明
0	エラーなし。
<0	エラー。パネルまたはページが存在しないなど。

EA_Panel.ZoomPanelIn

現在の通常パネルまたは印刷ページ上の可視化オブジェクトを10パーセント縮小します。

```
EA_Panel.ZoomPanelIn()
```

注記

- ズームは、スコープパネルまたはフローティングパネル上の可視化オブジェクトには影響しません。

例

```
Sub ZoomInPanel()
  Dim i As Integer
  EA_Panel.ShowPanel 2
  'Zoom in 3 times (30% smaller)
  For i = 0 To 2
    EA_Panel.ZoomPanelIn
  Next
End Sub
```

EA_Panel.ZoomPanelOut

現在の通常パネルまたは印刷ページ上の可視化オブジェクトを10パーセント拡大します。

```
EA_Panel.ZoomPanelOut()
```

注記

- ズームは、スコープパネルまたはフローティングパネル上の可視化オブジェクトには影響しません。

例

```
Sub ZoomOutPanel()
  Dim i As Integer
  EA_Panel.ShowPanel 2
  ''Zoom out 3 times (30% larger)
  For i = 0 To 2
    EA_Panel.ZoomPanelOut
  Next
End Sub
```

EA_Ribbon

EA_Ribbonオブジェクトはカスタムのリボンユーザインタフェースの拡張機能と対話のためのメソッドを提供します。
EasyScriptアプリケーションのユーザインタフェースを作成も参照してください。

以下の関数を提供しています:

- ファイルからすべてのユーザインタフェースを読み込みます: EA_Ribbon.LoadExtensions
- リボンのカスタマイズ: EA_Ribbon.AddControl、EA_Ribbon.AddTab、EA_Ribbon.AddGroupToCustomTab、EA_Ribbon.AddGroupToSystemTab
- リボンタブを表示: EA_Ribbon.ShowTab、EA_Ribbon.ShowSysTab
- コントロールのキャプションを設定: EA_Ribbon.SetCaption
- リボンシステム全体の表示や有効状態を設定: EA_Ribbon.SetRibbonState
- リボンタブの表示や有効状態を設定: EA_Ribbon.SetTabState、EA_Ribbon.SetSysTabState
- リボンタブのグループの可視状態を設定: EA_Ribbon.SetGroupState
- コントロールの表示や有効状態を設定: EA_Ribbon.SetControlState
- リボンシステムメニューの可視状態を設定: EA_Ribbon.SetControlMenuState
- リボンの表示状態を取得: EA_Ribbon.Visible
- リボンが最小化かどうかを取得: EA_Ribbon.Minimized
- リボン バーを更新: EA_Ribbon.Refresh
- リボンシステム、メニューおよびコンポーネントウィンドウを含むcatmanEasy GUIを非表示に設定: EA_Ribbon.HideGUI
- コントロール(テキストボックス、リストボックス、チェックボックス)の内容や状態を取得/設定: EA_Ribbon.GetValue、EA_Ribbon.SetValue
- リストボックスコントロールのための関数: EA_Ribbon.GetListBoxItem、EA_Ribbon.GetListBoxSelIndex、EA_Ribbon.SetListBoxSelIndex、EA_Ribbon.ClearListbox、EA_Ribbon.AddListboxItem、EA_Ribbon.SetListboxItems、EA_Ribbon.RemoveListboxItem
- アイコンを読み込み、コントロールに割り当て: EA_Ribbon.LoadIcon、EA_Ribbon.SetIcon
- ポップアップメニューを表示: EA_Ribbon.ShowPopupMenu、EA_Ribbon.ShowTemporaryPopupMenu
- catman組み込みのコンテキストメニューに項目を追加: EA_Ribbon.ExtendContextMenu
- 組み込みのコンテキストメニューの可視化制御: EA_Ribbon.SetContextMenuMode
- メッセージバーの表示/非表示: EA_Ribbon.ShowMessageBar、EA_Ribbon.HideMessageBar
- メッセージバーに表示する新規メッセージを追加: EA_Ribbon.AddMessageBarMessage
- プログレスバーを設定: EA_Ribbon.SetProgress
- ステータスバーのステータステキストを設定: EA_Ribbon.SetStatus
- 組み込みコンポーネントウィンドウを表示/非表示: EA_Ribbon.ShowComponentWindow
- コンテナを基準とした相対座標を絶対画面座標に変換: EA_Ribbon.GetScreenCoordinates

このオブジェクトでは以下のメソッドが公開されています:

EA_Ribbon.AddControl ・・・

EA_Ribbon.HideMessageBar ・・・

EA_Ribbon.AddControl

リボンタブのグループ内に、新規コントロールを追加します。
EA_Ribbon.AddGroupToCustomTabおよびEA_Ribbon.AddGroupToSystemTabを使用して、リボンタブにグループを追加します。

```
  EA_Ribbon.AddControl(ByVal ParentGroupName As String, ByVal ControlName As
String, ByVal Caption As String, ByVal CallerType As eCallerTypes, ByVal MethodName
As String, Optional ByVal IsDropdown As Integer, Optional ByVal ParentControlName
As String, Optional ByVal IconFile As String, Optional ByVal ControlType As
eControlTypes, Optional ByVal ControlStyle As eControlStyles, Optional ByVal ToolTip
As String) As Long
```

引数	説明
ParentGroupName	コントロールをホストしているグループ名。
ControlName	コントロール名。他の関数のコントロールと識別するために使用します。名前は一意でなければなりません!
Caption	キャプション。
CallerType	0 = コントロールはEasyScriptで作成され、*MethodName*はスクリプトのプロシージャを示します。 1 = コントロールはAddInで作成され、*MethodName*はAddInのパブリックメソッドを示します。
MethodName	スクリプトのプロシージャまたはパブリックAddInメソッド。ユーザがコントロールをクリックした場合に呼び出されます。プロシージャ/メソッドを呼び出さない場合、空の文字列を指定します(例えばドロップダウンコントロールやラベル)。 CallerType = 1 (AddIn)の場合、名前はメソッドの完全修飾名である必要があり、AddInのクラス名の前に続きます(例えば"MyAddIn.clsMain.MyMethod")。
IsDropdown	0 = 標準コントロール 1 = ドロップダウンコントロール(サブメニューコントロールをホスト)
ParentControlName	ドロップダウンコントロール名。この場合、新規コントロールは親コントロール下のサブメニューコントロールとして追加されます。 最大2つのサブレベルが許可されます。
IconFile	アイコンファイル名。ファイル名にパスが含まれていない場合、デフォルトの画像フォルダ(User\MyDocuments\HBM\catmanEasy\Images)のアイコンを検索します。
ControlType	0 = ボタン 1 = チェックボックス 2 = テキストボックス 3 = コンボボックス 4 = ラベル 5 = ラジオボタン チェックボックスやテキストボックスコントロールへのアクセスは、EA_Ribbon.Get/SetValueメソッドを使用します。コンボボックスコントロールへのアクセスは、ListBoxメソッドを使用します。
ControlStyle	0 = 自動 1 = キャプションのみ 2 = アイコンのみ 3 = アイコン+キャプション 4 = アイコン + 下のキャプション
ToolTip	ツールチップテキスト。コントロールがサブメニューコントロールの場合、キャプションの下に説明テキストとして表示されます。

⚠ 現在、作成後のユーザインタフェース項目は削除できないことに注意してください。catmanEasy/AP終了後まで常駐します!

⚠ 追加ししたコントロールは、EA_Ribbon.Refreshメソッドを呼び出すまで表示されません!

EA_Ribbon.AddGroupToCustomTab

リボンタブに新規グループを追加します。
catmanEasy/APの組み込みリボンタブにグループを追加するにはEA_Ribbon.AddGroupToSystemTabを使用します。

```
  EA_Ribbon.AddGroupToCustomTab(ByVal ParentTabName As String, ByVal GroupName As
String, ByVal Caption As String) As Long
```

引数	説明
ParentTabName	グループをホストしているタブ名。
GroupName	グループ名。他の関数のグループと識別するために使用します。名前は一意でなければなりません!
Caption	キャプション。

⚠ 現在、作成後のユーザインタフェース項目は削除できないことに注意してください。catmanEasy/AP終了後まで常駐します!

⚠ 追加ししたグループは、EA_Ribbon.Refreshメソッドを呼び出すまで表示されません!

EA_Ribbon.AddGroupToSystemTab

組み込みリボンタブに新規グループを追加します。
*AddTab*で作成したタブににグループを追加するにはEA_Ribbon.AddGroupToCustomTabを使用します。

```
  EA_Ribbon.AddGroupToSystemTab(ByVal ParentTabID As Integer, ByVal GroupName As
String, ByVal Caption As String) As Long
```

引数	説明
ParentTabID	グループをホストしているタブのID。 TAB_COCKPIT = 0 TAB_CHANCONFIG = 1 TAB_SENSORDB = 2 TAB_OPTO = 3 TAB_REALTIMEMATH = 4 TAB_CHANNELCHECK = 5 TAB_DAQJOBS = 6 TAB_ANALYSIS_EXPLORER = 7 TAB_ANALYSIS_DATAVIEW = 8 TAB_VISUAL = 9 TAB_ANALYSIS_MATH = 10 TAB_ANALYSIS_EXPORT = 11 TAB_AUTOSEQ = 12 TAB_EASYSCRIPT = 13 TAB_VIDEOCAM = 14
GroupName	グループ名。他の関数のグループと識別するために使用します。名前は一意でなければなりません!
Caption	キャプション。

⚠ 現在、作成後のユーザインタフェース項目は削除できないことに注意してください。catmanEasy/AP終了後まで常駐します!

⚠ EA_Ribbon.Refreshメソッドを呼び出すまで、追加ししたグループは表示されません!

EA_Ribbon.AddListBoxItem

リストボックス内に項目を挿入します。項目は任意の位置、または最後尾に追加することができます。また項目(ItemData)に番号を割り当てることもできます。これにより、テキストと番号の組み合わせを管理することができます。

```
  EA_Ribbon.AddListboxItem(ByVal ControlName As String, ByVal ItemText As String,
ByVal ItemData As Long, Optional ByVal Index As Integer) As Long
```

引数	説明
ControlName	コントロール名。
ItemText	項目リストのテキスト。
ItemData	項目リストの番号。
Index	項目を挿入するインデックス。Index = 0または欠落している場合、項目はリストの最後に追加されます。

戻り値	説明
0	エラーなし。
<0	エラー。コントロールが存在しないなど。

注記

- 単一コマンドでリストを埋めるにはEA_Ribbon.SetListboxItemsを使用します。

EA_Ribbon.AddMessageBarMessage

メッセージバーに表示する新規メッセージを追加します。

```
  EA_Ribbon.AddMessageBarMessage(ByVal MsgBold As String, ByVal MsgPlain As String,
ByVal IconID As Long) As Long
```

引数	説明
MsgBold	条件や問題を説明するメインの指示テキスト。
MsgPlain	意味を説明する補足説明テキスト。
IconID	通知領域に表示するアイコンのID。事前定義された記号は次のとおりです: 1 = 情報記号 2 = 警告記号 3 = エラー記号

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Ribbon.AddTab

catmanEasy/APのリボンバーに新規タブを追加します。

```
  EA_Ribbon.AddTab(ByVal TabName As String, ByVal Caption As String, ByVal IconFile
As String, ByVal ProjectVisibility As eProjectTypes, Optional ByVal WorkingPane As
Integer) As Long
```

引数	説明
TabName	タブ名。他の関数のタブと識別するために使用します。名前は一意でなければなりません!
Caption	キャプション。
IconFile	アイコンファイル名。ファイル名にパスが含まれていない場合、デフォルトの画像フォルダ(User\MyDocuments\HBM\catmanEasy\Images)のアイコンを検索します。
ProjectVisibility	プロジェクトの種類によってタブを表示/非表示に指定します: 0=DAQプロジェクト 1=解析プロジェクト 2=両方のプロジェクト
WorkingPane	ユーザが新規作成したタブをクリックした場合に表示されるシステム(組み込み)のワークスペース: TAB_COCKPIT = 0 TAB_CHANCONFIG = 1 TAB_SENSORDB = 2 TAB_OPTO = 3 TAB_REALTIMEMATH = 4 TAB_CHANNELCHECK = 5 TAB_DAQJOBS = 6 TAB_ANALYSIS_EXPLORER = 7 TAB_ANALYSIS_DATAVIEW = 8 TAB_VISUAL = 9 TAB_ANALYSIS_MATH = 10 TAB_ANALYSIS_EXPORT = 11 TAB_AUTOSEQ = 12 TAB_EASYSCRIPT = 13 TAB_VIDEOCAM = 14 この引数を省略した場合、Visualisation workspace (9)を表示します。

⚠ 現在、作成後のユーザインタフェース項目は削除できないことに注意してください。catmanEasy/AP終了後まで常駐します!

⚠ EA_Ribbon.Refreshメソッドを呼び出すまで、追加ししたコントロールは表示されません!

EA_Ribbon.ClearListBox

リストボックス型のコントロールの内容を消去します。

```
EA_Ribbon.ClearListbox(ByVal ControlName As String) As Long
```

引数	説明
ControlName	コントロール名。

戻り値	説明
0	エラーなし。
<0	エラー。コントロールが存在しないなど。

注記

- 単一項目を削除するにはEA_Ribbon.RemoveListboxItemを使用します。

EA_Ribbon.ExtendContextMenu

catman組み込みのコンテキストメニューに項目を追加します。

このメソッドはEA_Ribbon.ShowPopupMenuでポップアップメニューを作成して表示させるよりも容易ですが、位置やアイコンの種類といった高度なデザインは使用できません。

```
  EA_Ribbon.ExtendContextMenu(ByVal MenuID As eContextMenuIDs, ByVal ID As Long,
ByVal Caption As String, ByVal HandlerProc As String, Optional ByVal IconID As Long,
Optional ByVal Description As String, Optional ByVal BeginGroup As Integer) As Long
```

引数	説明
MenuID	組み込みメニューのID。インテリセンス機能は、利用可能なIDのリストを提供します。
ID	メニュー項目の一意のID。 65256以上から始める必要がります。
Caption	メニュー項目のキャプション。
HandlerProc	メニュー選択を処理するスクリプト内のプロシージャの名前。このプロシージャは、特別な*命名規則*に従い、特別な*署名*を持つ必要があります: 命名規則:プロシージャ名は"MenuHandler_"で開始する必要があります(例えば"MenuHandler_MyPopupMenu")。 署名: `MenuHandler_xxxx(ByVal Index As Integer, ByVal Caption As String)` ユーザがメニュー項目を選択した時、catmanはハンドラ関数を呼び出し、選択した項目のインデックス(最初は1)とともにキャプションを渡します。
IconID	任意です。メニュー項目に割り当てられるアイコンのID。最初にEA_Ribbon.LoadIconで読み込みます。省略した場合、メニューにアイコンは表示されません。
Description	任意です。メニュー項目の下に表示させる説明(ツールチップ)を指定します。
BeginGroup	任意です。1に設定すると、メニュー項目の上に区切り線が表示されます。

引数	説明
ControlName	コントロール名。

戻り値	説明
0	エラーなし。
<0	エラー。HandlerProcが存在しないなど。

注記

- メニュー項目に同一**ID**がある場合、作成したメニュー項目に置き換えられます。

EA_Ribbon.GetListBoxItem

リストボックスコントロールのリストの任意の位置の項目を返します。

```
  EA_Ribbon.GetListboxItem(ByVal ControlName As String, ByVal Index As Integer,
ItemText As String, ItemData As Long) As Long
```

引数	説明
ControlName	コントロール名。
Index	項目の位置。リストの最初の項目の位置は1です。
ItemText	項目のテキスト。
ItemData	リストの項目番号。

戻り値	説明
0	エラーなし。
<0	エラー。コントロールが存在しないなど。

注記

- 選択した位置の検索にはEA_Ribbon.GetListboxSelIndexを使用します。
- 選択している項目のテキストを取得するにはEA_Ribbon.GetValueを使用します。

EA_Ribbon.GetListboxSelIndex

選択しているリストボックスの項目の位置(インデックス)を検索します。

```
  EA_Ribbon.GetListboxSelIndex( ByVal ControlName As String, Index As Integer) As
Long
```

引数	説明
ControlName	コントロール名。
Index	選択した項目の指標。リストの最初の項目の位置は1です。

戻り値	説明
0	エラーなし。
<0	エラー。リストボックスが存在しないなど。

注記

- 選択している項目のテキストを取得するにはEA_Ribbon.GetValueを使用します。
- リストの任意の要素の値(テキスト)を知るにはEA_Ribbon.GetListboxItemを使用します。

EA_Ribbon.GetScreenCoordinates

コンテナの相対座標(_*OnMouseDown/Up*プロシージャに渡します)を絶対画面座標に変換します。ポップアップメニューの表示に_*MouseXXX*プロシージャを使用する場合、最初にx、y値を絶対画面座標に変換します。

⚠ catmanはコンテナが最後に_*MouseDown*または*MouseUp*イベントを受け取ったコントロールであると前提としています! EA_Ribbon.ShowTemporaryPopupMenuを参照してください。

```
  EA_Ribbon.GetScreenCoordinates(ByVal xRel as Long, ByVal yRel As Long, xAbs as
Long, yAbs as Long) As Long
```

引数	説明
xRel	ピクセル単位のx位置(コンテナ座標)。
yRel	ピクセル単位のy位置(コンテナ座標)。
xAbs	ピクセル単位で返すx位置(絶対画面座標)。
yAbs	ピクセル単位で返すy位置(絶対画面座標)。

EA_Ribbon.GetValue

コントロールの内容や状態を読み出します。データ型はコントロールの型に応じて異なります。

```
EA_Ribbon.GetValue(ByVal ControlName As String, Value As Variant) As Long
```

引数	説明
ControlName	コントロール名。
Value	コントロールの内容や状態。
	チェックボックスまたはラジオボタン: 0 = チェックしていない 1 = チェックしている
	リストボックス: 選択している項目のテキスト(選択しているインデックスの取得にはEA_Ribbon.GetListBoxSelIndexを使用します)。
	テキストボックス: 入力フィールド内のテキスト。

戻り値	説明
0	エラーなし。
<0	エラー。オブジェクトが存在しないなど。

EA_Ribbon_SetValueも参照してください。

EA_Ribbon.HideGUI

リボンシステム、メニューおよびコンポーネントウィンドウを含むcatmanEasy GUIを非表示にします。

またコマンドラインの引数/HideGUIでも、すべてのGUI要素(リボン バー、メニュー、およびコンポーネント ウィンドウ)を非表示にすることができます。詳細はcatmanのGUIを非表示にする方法を参照してください。

```
EA_Ribbon.HideGUI
```

EA_Ribbon.HideMessageBar

メッセージバーを非表示にします。

```
EA_Ribbon.HideMessageBar() As Long
```

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Ribbon.LoadExtensions

ファイルからすべてのユーザインタフェース(タブ、グループおよびコントロール)を読み取ります。
EasyScriptアプリケーションのユーザインタフェースを作成も参照してください。

```
EA_Ribbon.LoadExtensions(ByVal FileName As String) As Long
```

引数	説明
FileName	ファイル名。ファイル名にパスが含まれていない場合、または指定したパスが見つからない場合、catmanEasy/APの\PROJECTディレクトリ内を検索します。 ファイルからインタフェースを読み取り後にデフォルトのユーザインタフェースに戻りたい場合、空の文字列を指定します。

戻り値	説明
0	エラーなし。
<0	エラー。ファイルが見つからない、または無効など。

注記

- リボンの拡張は一度に一つのみ配置することができます!
- スクリプトの終了後、インタフェースはデフォルトにリセットされます。

EA_Ribbon.LoadIcon

コントロールやメニュー項目を割り当てるため、アイコンをリボンバーシステムに読み込みます。

```
EA_Ribbon.LoadIcon(ByVal FileName As String, ByVal IconID as Long) As Long
```

引数	説明
FileName	アイコンファイル名。ファイル名にアイコンのパスが含まれていない場合、デフォルトの画像フォルダ(User\MyDocuments\HBM\catmanEasy\Images)を検索します。
IconID	アイコンのID。 後から*EA_Ribbon.SetIcon*、*EA_Ribbob.ShowTemporaryPopupMenu*や*EA_Ribbon.ExtendContextMenus*関数で参照するため、このIDが必要です。 ⚠ アイコンIDは20000から始めます!

戻り値	説明
0	エラーなし。
<0	エラー。ファイルが見つからない、または無効など。

注記

- IconIDは、EA_Ribbon.SetIconでコントロールにアイコンを割り当てるために必要です。

EA_Ribbon.Minimized

リボンが最小化かどうかを取得、または設定します。

```
Property EA_Ribbon.Minimized
```

EA_Ribbon.Refresh

このメソッドは、リボンバーを更新(再描画)します。追加したタブ、グループ、またはコントロールを表示する前に、必ずこのメソッドを呼び出す必要があります。

EA_Ribbon.AddTab、EA_Ribbon.AddGroupToCustomTab、EA_Ribbon.AddGroupToSystemTab、EA_Ribbon.AddControl も参照してください。

```
EA_Ribbon.Refresh()
```

戻り値	なし

注記

- 最初にすべてのカスタムユーザインタフェースを追加した後に、*Refresh*を一回呼び出すことを推奨しています。

EA_Ribbon.RemoveListboxItem

リストボックスから項目を削除します。

```
  EA_Ribbon.RemoveListboxItem(ByVal ControlName As String, ByVal Index As Integer)
As Long
```

引数	説明
ControlName	コントロール名。
Index	削除する項目のインデックス。

戻り値	説明
0	エラーなし。
<0	エラー。コントロールが存在しないなど。

注記

- リストボックスのすべての項目を削除するにはEA_Ribbon.ClearListboxを使用します。

EA_Ribbon.SetCaption

コントロールのキャプションを設定します。

```
  EA_Ribbon.SetCaption(ByVal ControlName As String, ByVal NewCaption As String) As
Long
```

引数	説明
ControlName	コントロール名。
NewCaption	オブジェクトの新規キャプション。

戻り値	説明
0	エラーなし。
<0	エラー。コントロールが存在しないなど。

EA_Ribbon.SetContextMenuMode

このメソッドは、パネルオブジェクトやパネル領域で右クリックしたとき、通常表示されるコンテキストメニューを非表示にすることができます。これはcatman組み込みのコンテキストメニューと、ユーザのポップアップメニュー(EA_Ribbon.ShowTemporaryPopupMenuやEA_Ribbon.ShowPopupMenu)との干渉を防ぎます。例えばスクリプト実行時にパネル領域にMouseUpイベントで、エクスポートやパネルの印刷機能を備えた小さなコンテキストメニューを呼び出します。このイベント(またはMouseDownやClick)に独自のポップアップを表示させることを想像してください: 両方のメニューが次々と表れます。これを防ぐには*EA_Ribbon.SetContextMenuMode*により、すべてのオブジェクトクラスの組み込みコンテキストメニューを無効にします。

```
EA_Ribbon.SetContextMenuMode(ByVal Parent As Integer, ByVal Hide As Integer)
```

引数	説明
Parent	オブジェクトクラス: 0 = パネルの背景領域 1 = Real-timeグラフ 2 = 概要グラフ 3 = DAQモードのテーブル 4 = 解析モードのテーブル 5 = デジタル表示器 6 = アナログメータ 7 = バー表示器 8 = LED 9 = テキスト 10 = 極座標グラフ 11 = 汎用テーブル 12 = すべてのオブジェクト
Hide	0 = メニューを表示 1 = メニューを非表示

EA_Ribbon.SetControlMenuState

メインメニュー("File")、リボン右側のメニュー(解析モード、Help、Windowsなどへの切り替え)、およびQuick Access Toolbarを含むリボンシステムメニューの*表示*状態を設定します。

```
EA_Ribbon.SetControlMenuState(ByVal State As Integer)
```

引数	説明
State	0 = 非表示 1 = 表示

戻り値	なし

EA_Ribbon.SetControlState

コントロールの*表示*や*有効*状態を設定します。

```
  EA_Ribbon.SetControlState(ByVal ControlName As String, ByVal Mode As Integer, ByVal
State As Integer) As Long
```

引数	説明
ControlName	コントロール名。
Mode	0 = 表示状態 1 = 有効状態
State	0 = 非表示または無効 1 = 表示または有効

戻り値	説明
0	エラーなし。
<0	エラー。コントロールが見つからないなど。

EA_Ribbon.SetGroupState

リボンタブのグループの*表示*状態を設定します。

```
  EA_Ribbon.SetGroupState(ByVal GroupName As String, ByVal Mode As Integer, ByVal
State As Integer) As Long
```

引数	説明
GroupName	グループ名。
Mode	0 = 表示状態
State	0 = 非表示または無効 1 = 表示または有効

戻り値	説明
0	エラーなし。
<0	エラー。コントロールが見つからないなど。

EA_Ribbon.SetIcon

コントロールにアイコンを割り当てます。

```
EA_Ribbon.SetIcon(ByVal ControlName As String, ByVal NewIconID as Long) As Long
```

引数	説明
ControlName	コントロール名。
IconID	割り当てるアイコンのID。

戻り値	説明
0	エラーなし。
<0	エラー。コントロールが見つからないなど。

注記

- アイコンは事前にリボン バー システムに、EA_Ribbon.LoadIconにより読み込まれている必要があります。

EA_Ribbon.SetListboxItems

リストボックスコントロールのすべての内容を設定します。

```
  EA_Ribbon.SetListboxItems(ByVal ControlName As String, ByVal ListItems As String)
As Long
```

引数	説明
ControlName	コントロール名。
ListItems	リストボックスの新規内容。個々の項目はCRLF (vbCrLf)で区切る必要があります。

戻り値	説明
0	エラーなし。
<0	エラー。コントロールが存在しないなど。

注記

- リストボックスの以前の内容は、この操作で上書きされます。
- 単一の項目を追加するにはEA_Ribbon.AddListboxItemを使用します。

EA_Ribbon.SetListBoxSelIndex

リストボックスの項目の位置(インデックス)を設定します。

```
  EA_Ribbon.SetListboxSelIndex(ByVal ControlName As String, ByVal Index As Integer)
As Long
```

引数	説明
ControlName	コントロール名。
Index	選択する項目のインデックス。リストの最初の項目の位置は1です。

戻り値	説明
0	エラーなし。
<0	エラー。リストボックスが存在しない、インデックスが無効など。

EA_Ribbon.SetProgress

catmanEasy/APのプログレスバーの値を設定します。

```
EA_Ribbon.SetProgress(ByVal NewValue As Variant, Optional ByVal Msg As String)
```

引数	説明
NewValue	プログレスバーの値。値の範囲: 0から100。
Msg	ステータスバーに表示させるメッセージ。

EA_Ribbon.SetRibbonState

リボンシステム全体の*表示*や*有効*状態を設定します。

```
EA_Ribbon.SetRibbonState(ByVal Mode As Integer, ByVal State As Integer) As Long
```

引数	説明
Mode	0 = 表示状態 1 = 有効状態
State	0 = 非表示または無効 1 = 表示または有効

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Ribbon.SetStatus

catmanのステータスバーにメッセージを書き込みます。

```
  EA_Ribbon.SetStatus(ByVal NewText As String, Optional ByVal IconID As Long,
Optional ByVal NewColor as Long)
```

引数	説明
NewText	ステータスメッセージのテキスト。
IconID	アイコンのID。省略またはゼロの場合、アイコンを表示しません。
NewColor	テキスト色。省略した場合、黒色。

注記

- アイコンの読み込みはEA_Ribbon.LoadIconを使用します。

EA_Ribbon.SetSysTabState

例えばCockpitやChannel Configurationタブなどの、システムのリボンタブの*表示*や*有効*状態を設定します。

```
  EA_Ribbon.SetSysTabState(ByVal TabIndex As Integer, ByVal Mode As Integer, ByVal
State As Integer) As Long
```

引数	説明
TabIndex	システムタブのインデックス: -1 = すべてのタブ 0 = Cockpit 1 = DAQ channels 2 = Sensor database 3 = Optical functions 4 = Computations 5 = Channel check 6 = DAQ jobs 7 = テストエクスプローラ(解析モード) 8 = DataViewer(解析モード) 9 = Visualization 10 = Computations(解析モード) 11 = Export(解析モード) 12 = AutoSequence editor 13 = EasyScript editor 14 = Video cameras
Mode	0 = 表示状態 1 = 有効状態
State	0 = 非表示または無効 1 = 表示または有効

戻り値	説明
0	エラーなし。
<0	エラー。タブが見つからないなど。

例

```
  Sub CustomizeRibbon()
    'Hide the Visualization tab.
    EA_Ribbon.SetSysTabState(TAB_VISUAL, ST_VISIBLE, ST_FALSE) 'EA_Ribbon.
SetSysTabState(9, 0, 0)
    'Show the Visualization tab again.
    EA_Ribbon.SetSysTabState(TAB_VISUAL, ST_VISIBLE, ST_TRUE) 'EA_Ribbon.
SetSysTabState(9, 0, 1)

    'Disable the Visualization tab.
    EA_Ribbon.SetSysTabState(TAB_VISUAL, ST_ENABLED, ST_FALSE) 'EA_Ribbon.
SetSysTabState(9, 1, 0)
    'Enable the Visualization tab.
    EA_Ribbon.SetSysTabState(TAB_VISUAL, ST_ENABLED, ST_TRUE) 'EA_Ribbon.
SetSysTabState(9, 1, 1)
  End Sub
```

EA_Ribbon.SetTabState

ユーザ定義のリボンタブの*表示*や*有効*状態を設定します。

```
  EA_Ribbon.SetTabState(ByVal TabName As String, ByVal Mode As Integer, ByVal State
As Integer) As Long
```

引数	説明
TabName	タブ名。
Mode	0 = 表示状態 1 = 有効状態
State	0 = 非表示または無効 1 = 表示または有効

戻り値	説明
0	エラーなし。
<0	エラー。タブが見つからないなど。

EA_Ribbon.SetValue

コントロールの内容や状態を設定します。データ型はコントロールの種類によって異なります。

```
EA_Ribbon.SetValue(ByVal ControlName as String, ByVal vNewValue As Variant) As Long
```

引数	説明
ControlName	コントロール名。
vNewValue	コントロールの新規内容や状態。これはコントロールの種類によって異なります。
	チェックボックスまたはラジオボタン: 0 = チェックしていない 1 = チェックしている
	リストボックス: 現在選択している項目のテキスト
	テキストボックス: 入力フィールド内のテキスト。

戻り値	説明
0	エラーなし。
<0	エラー。オブジェクトが存在しないなど。

注記

- コントロールのキャプションの変更にはEA_Ribbon.SetCaptionを使用します。

EA_Ribbon.ShowComponentWindow

このメソッドを使用すると、組み込みコンポーネントウィンドウの一つを表示、または非表示にすることができます。

```
EA_Ribbon.ShowComponentWindow(ByVal Component As Integer, ByVal Flag As Integer)
```

引数	説明
Component	0 = センサーウィンドウ 1 = チャンネル情報ウィンドウ 2 = DAQ モード チャネル 選択 3 = 可視化オブジェクトの構成 4 = 解析プロジェクトツリー
Flag	0= ウィンドウを非表示 1= ウィンドウを表示

EA_Ribbon.ShowMessageBar

詳細なメッセージを含むメッセージバーを表示します。

```
  EA_Ribbon.ShowMessageBar(ByVal MsgBold As String, ByVal MsgPlain As String, ByVal
IconID As Long) As Long
```

引数	説明
MsgBold	状態や問題を説明するメインの指示テキスト。
MsgPlain	意味を説明する補足説明テキスト。
IconID	通知領域に表示されるアイコンのID。事前定義されているシンボルは次のとおりです: 1 = 情報シンボル 2 = 警告記号 3 = エラー記号

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Ribbon.ShowPopupMenu

バージョン3.0.1以降では、最初にユーザインタフェースをカスタマイズする必要はなく、最初からポップアップメニューを表示させることができます: *EA_Ribbon.ShowTemporaryPopup*

ポップアップメニュー(コンテキストメニュー)を表示します。ポップアップメニューを作成するには最初にサブコントロールを許可するリボン コントロール、つまり、"parent"コントロールを作成する必要があります。そして、このコントロールにサブコントロールを追加します - これがポップアップメニューの項目になります。各項目はスクリプトの動作(例えばプロシージャ)を選択します。これは"Customize user interface"ダイアログで行います(EasyScriptアプリケーションのユーザインタフェースを作成を参照)。親コントロールをリボンバーに表示させたくない場合、"Invisible"プロパティをtrueに設定します。

```
  EA_Ribbon.ShowPopupMenu(ByVal ControlName As String, ByVal x as Long, ByVal y As
Long) As Long
```

引数	説明
ControlName	親コントロール名。
x	ピクセル単位のx位置(画面の絶対座標)。
y	ピクセル単位のy位置(画面の絶対座標)。

戻り値	説明
0	エラーなし。
<0	エラー。コントロールが存在しないなど。

注記

- _MouseXXXプロシージャのいずれかを使用してメニューを表示するには、最初に関数に渡されたx、y値をEA_Ribbon.GetScreenCoordinates画面の絶対座標に変換します。そして変換された値をx、y引数として*ShowPopupMenu*関数へ渡します。

- 作成したポップアップメニューは、catman組み込みのコンテキストメニューと競合する可能性があることに注意してください。例えばスクリプト実行時にパネル領域にMouseUpイベントで、エクスポートやパネルの印刷機能を備えた小さなコンテキストメニューを呼び出します。このイベント(またはMouseDownやClick)に独自のポップアップを表示させることを想像してください: 両方のメニューが次々と表れます。これを防ぐにはEA_Ribbon.SetContextMenuModeにより、すべてのオブジェクトクラスの組み込みコンテキストメニューを無効にします。

EA_Ribbon.ShowSysTab

例えばCockpitやChannel Configurationタブなどの、システムのリボンタブを表示します。

```
EA_Ribbon.ShowSysTab(ByVal TabIndex As Integer) As Long
```

引数	説明
TabIndex	システムタブのインデックス: -1 = すべてのタブ 0 = Cockpit 1 = DAQ channels 2 = Sensor database 3 = Optical functions 4 = Computations 5 = Channel check 6 = DAQ jobs 7 = テストエクスプローラ(解析モード) 8 = DataViewer(解析モード) 9 = Visualization 10 = Computations(解析モード) 11 = Export(解析モード) 12 = AutoSequence editor 13 = EasyScript editor 14 = Video cameras

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Ribbon.ShowTab

ユーザ定義のリボンタブを表示します。

```
EA_Ribbon.ShowTab(ByVal TabName As String) As Long
```

引数	説明
TabName	タブ名。

戻り値	説明
0	エラーなし。
<0	エラー。タブが見つからないなど。

EA_Ribbon.ShowTemporaryPopupMenu

文字列のリストに基づいてポップアップメニュー(コンテキストメニュー)を表示します。

このメソッドはEA_Ribbon.ShowPopupMenuでポップアップメニューを作成して表示させるよりも容易ですが、位置やアイコンの種類といった高度なデザイン機能は使用できません。

```
EA_Ribbon.ShowTemporaryPopupMenu(ByVal ItemList As String, ByVal HandlerProc As
String, ByVal x as Long, ByVal y As Long, Optional ByVal IconList As String, Optional
ByVal DescriptionList As String, Optional ByVal GroupList As String) As Long
```

引数	説明
ItemList	"|"で区切られたメニュー項目のリスト。例えば、"Default|Slow|Fast"。
HandlerProc	メニュー選択を処理するスクリプト内のプロシージャ名。このプロシージャは特別な命名規則に従う、特別な署名を持っている必要があります。 命名規則: プロシージャ名は"**MenuHandler_**"で開始する必要があります(例えば"MenuHandler_MyPopupMenu")。 署名: `HandlerProc(ByVal Index As Integer, ByVal Caption As String)` ユーザがメニュー項目を選択した時、catmanはハンドラ関数を呼び出し、選択した項目のインデックス(最初は1)とともにキャプションを渡します。
x	ピクセル単位のx位置(画面の絶対座標)。
y	ピクセル単位のy位置(画面の絶対座標)。
IconList	任意です。"|"で区切られたメニュー項目に割り当てるアイコンのIDのリストを指定します。(例えば"20001|20002|20003")。最初にEA_Ribbon.LoadIconで読み込みます。省略した場合、メニューにアイコンが表示されません。
DescriptionList	任意です。"|"で区切られたメニュー項目の下に表示させる説明(ツールチップ)のリスト。
GroupList	任意です。"0"または"1"を"|"で区切り、メニュー項目の上に区切り線を表示させます。区切り線を表示させるには1に設定します(例えば "0|0|1")。

戻り値	説明
0	エラーなし。
<0	エラー。コントロールが存在しないなど。

注記

- _MouseXXXプロシージャのいずれかを使用してメニューを表示するには、最初に関数に渡されたx、y値をEA_Ribbon.GetScreenCoordinates画面の絶対座標に変換します。そして変換された値をx、y引数として*ShowPopupMenu*関数へ渡します。

- 作成したポップアップメニューは、catman組み込みのコンテキストメニューと競合する可能性があることに注意してください。例えばスクリプト実行時にパネル領域にMouseUpイベントで、エクスポートやパネルの印刷機能を備えた小さなコンテキストメニューを呼び出します。このイベント(またはMouseDownやClick)に独自のポップアップを表示させることを想像してください: 両方のメニューが次々と表れます。これを防ぐにはEA_Ribbon.SetContextMenuModeにより、すべてのオブジェクトクラスの組み込みコンテキストメニューを無効にします。

- 事前にユーザインタフェースをカスタマイズして事前に作成したポップアップメニューを表示させる別の方法はEA_Ribbon.ShowPopupMenuを参照してください。

例

```
Sub Panel_OnMouseDown(ByVal Button As Integer, ByVal x As Long, ByVal y As Long)
  'Automatically created procedure: Do not change or delete name or signature!
  Dim ItemList As String
  Dim sX As Long
  Dim sY As Long
  ItemList = "1|2|3"
  EA_Ribbon.GetScreenCoordinates x, y, sX, sY
  EA_Ribbon.ShowTemporaryPopupMenu ItemList, "MenuHandler_1", sX, sY
End Sub

Sub MenuHandler_1(ByVal Id As Integer, ByVal Text As String)
  MsgBox Id & "," & Text
End Sub
```

EA_Ribbon.Visible

リボンを表示するかどうかを設定、または取得します。

```
Property EA_Ribbon.Visible
```

注記

- リボンシステム全体の表示または非表示は、EA_Ribbon.SetRibbonStateを使用します。

EA_Test

解析モードでは、EA_Testよりテストデータにアクセスします。

- EA_Testメソッドは、テストのチャネルに含まれるデータを操作し、すなわち、チャネルの引数を要求します。catmanはしかし、変数や配列に働きかける関数と同様のEA_Mathオブジェクトも提供しています。

- *EA_Test*メソッドは、解析モードでのみ呼び出すことができます!

このオブジェクトでは以下が行えます:

- テストの読み込みや削除: EA_Test.Load、EA_Test.Remove
- 解析プロジェクトにロードされたテストの数を取得: EA_Test.TestCount
- テストデータのディレクトリ関数: EA_Test.DataDir、EA_Test.LastTestExported、EA_Test.ResolvePlaceholdersInFileName
- 複数のcatman標準のデータファイルを一つのデータファイルにマージ: EA_Test.MergeFiles
- テストからチャネルデータを配列に読み取り: EA_Test.GetChannelData、EA_Test.GetChannelSection
- チャネルの単一の値を取得/設定: EA_Test.GetSample、EA_Test.SetSample
- チャネルから最小/最大圧縮データの配列を取得: EA_Test.CompressChannel
- テストデータおよび演算をグラフ表示: EA_Test.PlotYT、EA_Test.PlotYX
- テストデータのスペクトログラムを表示: EA_Test.PlotSpectrogram
- チャネルの統計やトレースデータテーブルのデータを追加/削除: EA_Test.AddTableDatasource、EA_Test.RemoveTableDatasources
- 入力チャネルのACおよびDCコンテンツの推定値を計算: EA_Test.GetACDCLevels
- チャネル上で代数計算を実行: EA_Test.Calculator、EA_Test.Recalc
- 周波数解析およびデジタルフィルタ: EA_Test.AutoPowerSpectrumEx、EA_Test.JointTimeFrequencySpectrum、EA_Test.AmplitudeSpectrum、EA_Test.PhaseSpectrum、EA_Test.DigFilt
- ロゼットひずみゲージのひずみチャネルの応力解析: EA_Test.StrainAnalysis
- チャネル内の指定したイベント(例えば5 Vより大きい最初の振幅)の位置を検索: EA_Test.FindEvent
- チャネルの一部を抽出: EA_Test.CurveSection
- データセット(チャネル)のX軸を左または右方向に一定量シフト: EA_Test.CurveShift
- チャネルデータを補間: EA_Test.Interpolate
- チャネルの統計データを取得: EA_Test.GetStatistics
- チャネルデータから外れ値を削除: EA_Test.RemoveOutliers
- チャネルから特定のビットを抽出: EA_Test.ExtractBitChannel
- 空の演算チャネルを作成し、データを追加: EA_Test.CreateChannel、EA_Test.SetChannelData
- 現在の解析プロジェクトから、演算チャネルを削除(複数可): EA_Test.RemoveChannel、EA_Test.RemoveAllComputations
- 解析プロジェクトの演算チャネルの名前を変更: EA_Test.RenameComputation
- 計測チャネルの値間の時間間隔を変更: EA_Test.SetChannel_dt
- 時間間隔や角度セクタごとにピーク値を含む新規のデータセットを作成: EA_Test.CreatePeakValues
- 等高線グラフに表示する行列を作成: EA_Test.CreateMatrix、EA_Test.PlotMatrix
- テストプロパティの読み取り: EA_Test.TestAttribute
- チャネル情報(トレーサビリティデータ)の読み取り: EA_Test.ChannelAttribute
- OLEオートメーション日付からUnix日付形式へ、またはその逆に変換: EA_Test.ConvertTimeChannel
- 異なる形式の計測データをエクスポート: EA_Test.AddToExport、EA_Test.ClearExport、EA_Test.Export
- エクスポート リスト内のチャネルを取得: EA_Test.GetExportListCount、EA_Test.GetExportListItem
- チャネルのデータを異なる形式にエクスポート: EA_Test.ExportChannelData、EA_Test.ExportMultipleChannelData
- クラスカウントの関数: EA_Test.Counting_RainflowFromTo、EA_Test.Counting_RainflowRangeMean、EA_Test.Counting_SpanPairs、EA_Test.Counting_TimeAtLevel、EA_Test.Counting.Counting_JointTimeAtLevel
- ASCII/ConvertFastStream/SIEファイルをcatman標準形式に変換: EA_Test.ConvertASCIIFile、EA_Test.ConvertFastStream、EA_Test.ConvertSIEFile
- ASCIIファイルから、データの再生ファイルを作成: EA_Test.MakePlaybackFileFromASCII

- CANの生データをASCIIファイルに変換: EA_Test.CANRawConvertToASCII
- CANの生データからデコードされたデータを作成: EA_Test.CANRawDecodeSignal
- CANの生トレースファイルを読み込み: EA_Test.LoadCANTraceFile
- catman BIN形式への直接アクセス: EA_Test.GetDataFileInfo、EA_Test.GetDataFileChanCount、EA_Test.GetDataFileData、EA_Test.ShowDataFileChanPreview

このオブジェクトでは以下のメソッドが公開されています:

EA_Test.AddTableDatasource ・・・

EA_Test.Interpolate ・・・

EA_Test.AddTableDatasource

チャネルに統計やトレースデータテーブルのデータを追加します。

```
  EA_Test.AddTableDatasource(ByVal Test As Variant, ByVal Channel As Variant, ByVal
Panel As Variant, ByVal TableName As String) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 Testに空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネル位置は1です。Test = 0の場合、最初の演算チャネルの位置は1です。
Panel	パネル名または位置。テストの最初のパネルは1です。Panel = 0の場合、すべてのパネルのテーブルを検索することを意味します。
TableName	テーブル名。

戻り値	説明
>=0	データソースは正常に追加されました。
<0	エラー。すなわち、テーブルが見つからないか型が違います。

EA_Panel.SetValueも参照してください。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。
- テーブルからすべてのチャネルを削除するには、EA_Test.RemoveTableDatasourcesを使用します。

例

```
Sub Plot()
  EA_Test.Load "C:\MYDATA\Test1.TST"
  EA_Test.RemoveTableDatasources 1, "DATATABLE_1"
  EA_Test.AddTableDatasource 1, "pressure_1", 1, "DATATABLE_1"
  EA_Test.AddTableDatasource 1, "pressure_2", 1, "DATATABLE_1"
End Sub
```

EA_Test.AddToExport

エクスポートチャネルのリストに、テストの単一またはすべてのチャネルまたは、演算を追加します。リスト内のチャネルのエクスポートには、EA_Test.Exportを使用します。

```
EA_Test.AddToExport(ByVal Test As Variant, ByVal SourceType As Integer, Optional
ByVal Channel As Variant, Optional ByVal Aliasname As String) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 Testに空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
SourceType	0 = すべてのテストチャネル 1 = Channelパラメータで指定した単一チャネル。これは演算チャネルにすることもできます。 2 = すべての演算
Channel	チャネル名または位置。テストの最初のチャネル位置は1です。Test = 0の場合、最初の演算チャネルの位置は1です。
AliasName	エクスポートしたファイルで使用するチャネル名。これは、異なるテストから同名のチャネルが同じファイルにエクスポートされる場合に必要です。そうしないと、後でチャネルを区別できなくなります。

戻り値	説明
0	エラーなし。
<0	エラー。例えばチャンネルやテストが存在しないなど。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。

例

```
Sub ExportData()
  'Add single channels to export list.
  EA_Test.Load "C:\MYDATA\Test1.TST"
  EA_Test.AddToExport 1, 1, "Valve pressure"
  EA_Test.AddToExport 1, 1, "Temperature"
  EA_Test.AddToExport 1, 1, "Braking_force_left"

  'Add all channels to export list at once.
  EA_Test.Load "C:\MYDATA\Test2.TST"
  EA_Test.AddToExport 1, 0
  'Export in EXCEL format.
  EA_Test.Export("C:\MYDATA\Test2.xls", 3, "A comment")
End Sub
```

EA_Test.AmplitudeSpectrum

チャネルの振幅スペクトルを計算します。Δfがチャネル情報に保存されるため、周波数に対する結果を直接チャネルに表示することができます。

ジョイント時間周波数スペクトルの計算(EA_Test.JointTimeFrequencySpectrum)も参照してください。

```
  EA_Test.AmplitudeSpectrum(ByVal Test As Variant, ByVal Channel As Variant,
ByVal FFTPoints As Long, ByVal PercentOverlap As Single, ByVal AvgMode As Integer,
ByVal RemoveDCComponent As Integer, ByVal WindowType As Integer, ByVal ScaleType
As Integer, ByVal OutputUnit As Integer, ByVal ResultName As String, ByVal
CreateFreqChan as Integer) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 Testに空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネル位置は1です。Test = 0の場合、最初の演算チャネルの位置は1です。
FFTPoints	単一FFTのポイント数。
PercentOverlap	単一FFTオーバーラップ(%)。
AvgMode	単一FFT平均化の種類: 0 = リニア 1 = ピークホールド
RemoveDCComponent	0 = DC成分を削除しない 1 = DC成分を削除
WindowType	窓関数: 0 = なし 1 = 三角 2 = ハニング 3 = ハミング 4 = ブラックマン 5 = 完全ブラックマン 6 = ブラックマンハリス 7 = フラットトップ 8 = コサインテーパ 9 = スケーリング: ユニフォーム 10 = スケーリング: ハニング 11 = スケーリング: ハミング 12 = スケーリング: ブラックマンハリス 13 = スケーリング: 完全ブラックマン 14 = スケーリング: ブラックマン 15 = スケーリング: フラットトップ 16 = スケーリング: 4項ブラックマンハリス 17 = スケーリング: 7項ブラックマンハリス

ScaleType	スペクトルのスケール: 0 = スケーリングしない 1 = リニア 2 = dB 3 = dBm
OutputUnit	スペクトル出力単位: 0 = Amp. rms 1 = Amp. peak 2 = Amp. rms^2 3 = Amp. peak^2 4 = Amp. rms/Hz平方根 5 = Amp. peak/Hz平方根 6 = Amp. 二乗rms/Hz
ResultName	結果のチャネル名。
CreateFreqChan	該当する周波数値を保持する追加チャネルを作成: 0 = 周波数チャネルを作成しない 1 = 周波数チャネルを作成

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- *ResultName*という名前の演算が存在している場合、既存の計算はこのメソッドの引数に応じて変更されます。
- グラフやDataViewウィンドウは、計算後に自動では更新しないことに注意してください! 単一演算後に時間のかかる更新を行わないことにより、多くの計算が行えます。計算後の効果を表示するには、EA_Panel.RefreshAll、EA_Graph.RefreshAllやEA_DataView.RefreshAllを使用します。
- 選択した周波数分解能およびサンプリングレートに応じて、多くの部分スペクトルを計算した後、平均化します。
- このメソッドは、数百万サンプルの非常に大規模なデータセットに最適化されています。

例

```
  Sub ComputeSpectrum()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.AmplitudeSpectrum 1, "pressure_1", 1024, 67, 1, 0, 2, 0, 0, "Amplitude
spectrum of pressure", 0
  End Sub
```

EA_Test.AutoPowerSpectrum

チャネルの自動パワースペクトルを計算します。Δfがチャネル情報に保存されるため、周波数に対する結果を直接チャネルに表示することができます。

このメソッドは廃止されました。代わりにEA_Test.AutoPowerSpectrumExを使用してください。

```
  EA_Test.AutoPowerSpectrum(ByVal Test As Variant, ByVal Channel As Variant, ByVal
FreqRes As Double, ByVal WindowType As Integer, ByVal ScaleType As Integer, ByVal
OutputUnit As Integer, ByVal ResultName As String) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 最初にテストを読み込む必要があります。 Testに空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネル位置は1です。Test = 0の場合、最初の演算チャネルの位置は1です。
FreqRes	必要な周波数分解能(Hz)。
WindowType	窓関数: 0 = なし 1 = 三角 2 = ハニング 3 = ハミング 4 = ブラックマン 5 = 完全ブラックマン 6 = ブラックマンハリス 7 = フラットトップ 8 = コサインテーパ 9 = スケーリング: ユニフォーム 10 = スケーリング: ハニング 11 = スケーリング: ハミング 12 = スケーリング: ブラックマンハリス 13 = スケーリング: 完全ブラックマン 14 = スケーリング: ブラックマン 15 = スケーリング: フラットトップ 16 = スケーリング: 4項ブラックマンハリス 17 = スケーリング: 7項ブラックマンハリス
ScaleType	スペクトルのスケール: 0 = スケーリングしない 1 = リニア 2 = dB 3 = dBm

OutputUnit	スペクトル出力単位: 0 = Amp. rms 1 = Amp. peak 2 = Amp. rms^2 3 = Amp. peak^2 4 = Amp. rms/Hz平方根 5 = Amp. peak/Hz平方根 6 = Amp. 二乗rms/Hz
ResultName	結果のチャネル名。

戻り値	説明
0	エラーなし。
<0	エラー。データバッファが作成できないなど。

注記

- 選択した周波数分解能およびサンプリングレートに応じて、多くの部分スペクトルを計算した後、平均化します。例: サンプルレート9600Hzおよび必要な周波数分解能1Hzでは、基本的なFFTで少なくとも9600 x 2 = 19200計測ポイントが必要になります(結果には周波数の正の半分のみが観測されるため、式には係数2が生じることに注意してください)。FFTは2のべき乗に形成されるため、catmanEasy/APは最も近い2のべき乗、すなわち32768を要求します。元のデータレコードが小さい場合、データレコード内で収まる最大の2のべき乗を使用します。その場合、必要な周波数で実現できなくなります。次のステップでは、元のデータレコードで使用する値の数(32768)に合わせて、多くの部分スペクトルを計算します。最後に、計算した部分スペクトルを平均化します。そのため平均化を行わない場合、それに応じた高い周波数分解能を選択する必要があります。

- *ResultName*という名前の演算が存在している場合、既存の計算はこのメソッドの引数に応じて変更されます。

- グラフやDataViewウィンドウは、計算後に自動では更新しないことに注意してください! 単一演算後に時間のかかる更新を行わないことにより、多くの計算が行えます。計算後の効果を表示するには、EA_Panel.RefreshAll、EA_Graph.RefreshAllやEA_DataView.RefreshAllを使用します。

例

```
  Sub ComputeSpectrum()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.AutoPowerSpectrum 1, "pressure_1", 0.01, 0, 0, 0, "Pressure frequency
spectrum"
  End Sub
```

EA_Test.AutoPowerSpectrumEx

チャネルの自動パワースペクトルを計算します。Δfがチャネル情報に保存されるため、周波数に対する結果を直接チャネルに表示することができます。

ジョイント時間周波数スペクトルの計算(EA_Test.JointTimeFrequencySpectrum)も参照してください。

```
  EA_Test.AutoPowerSpectrumEx(ByVal Test As Variant, ByVal Channel As Variant,
ByVal FFTPoints As Long, ByVal PercentOverlap As Single, ByVal AvgMode As Integer,
ByVal RemoveDCComponent As Integer, ByVal WindowType As Integer, ByVal ScaleType
As Integer, ByVal OutputUnit As Integer, ByVal ResultName As String, ByVal
CreateFreqChan as Integer) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 Testに空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネル位置は1です。Test = 0の場合、最初の演算チャネルの位置は1です。
FFTPoints	単一FFTのポイント数。
PercentOverlap	単一FFTオーバーラップ(%)。
AvgMode	単一FFT平均化の種類: 0 = リニア 1 = ピークホールド
RemoveDCComponent	0 = DC成分を削除しない 1 = DC成分を削除
WindowType	窓関数: 0 = なし 1 = 三角 2 = ハニング 3 = ハミング 4 = ブラックマン 5 = 完全ブラックマン 6 = ブラックマンハリス 7 = フラットトップ 8 = コサインテーパ 9 = スケーリング: ユニフォーム 10 = スケーリング: ハニング 11 = スケーリング: ハミング 12 = スケーリング: ブラックマンハリス 13 = スケーリング: 完全ブラックマン 14 = スケーリング: ブラックマン 15 = スケーリング: フラットトップ 16 = スケーリング: 4項ブラックマンハリス 17 = スケーリング: 7項ブラックマンハリス

ScaleType	スペクトルのスケール: 0 = スケーリングしない 1 = リニア 2 = dB 3 = dBm
OutputUnit	スペクトル出力単位: 0 = Amp. rms 1 = Amp. peak 2 = Amp. rms^2 3 = Amp. peak^2 4 = Amp. rms/Hz平方根 5 = Amp. peak/Hz平方根 6 = Amp. 二乗rms/Hz
ResultName	結果のチャネル名。
CreateFreqChan	該当する周波数値を保持する追加チャネルを作成: 0 = 周波数チャネルを作成しない 1 = 周波数チャネルを作成

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- *ResultName*という名前の演算が存在している場合、既存の計算はこのメソッドの引数に応じて変更されます。
- グラフやDataViewウィンドウは、計算後に自動では更新しないことに注意してください! 単一演算後に時間のかかる更新を行わないことにより、多くの計算が行えます。計算後の効果を表示するには、EA_Panel.RefreshAll、EA_Graph.RefreshAllやEA_DataView.RefreshAllを使用します。
- 選択した周波数分解能およびサンプリングレートに応じて、多くの部分スペクトルを計算した後、平均化します。
- このメソッドは、数百万サンプルの非常に大規模なデータセットに最適化されています。

例

```
  Sub ComputeSpectrum()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.AutoPowerSpectrumEx 1, "pressure_1", 1024, 67, 1, 0, 2, 0, 0, "Power
spectrum of pressure", 0
  End Sub
```

EA_Test.Calculator

```
EA_Test.ComputeAcrossChannels (old naming style)
```

バージョン3.1以降このメソッドは、より意味のある短い名前に引き継がれました。*EA_Test.ComputeAcrossChannels* の代わりに *EA_Test.Calculator* の使用を推奨しています。

数式に現れる、すべての引数チャネルの代数計算を行ごとにを行います。結果として、参加しているチャネルの中で最も小さい長さのチャネルを作成します(他チャネルの超過値は破棄されます)。

```
EA_Test.Calculator(ByVal Expression As String, ByVal ResultName As String, ByVal
ResultUnit As String) As Long
```

引数	説明
Expression	式。チャネル名を式で使用することができます。解析プロジェクトに複数のテストがあり、同じ名前のチャネル名を計算する場合、チャネル名にかぎ括弧で囲んでテスト位置を追加する必要があります。 例: *1/3*(pressure_1[01] + pressure_1[02] + pressure_1[03])* チャネル名を混同する恐れがない場合、例えばプロジェクト内にテストが一つのみの場合、テスト位置は省略することができます。
ResultName	結果のチャネル名。
ResultUnit	結果チャネルの単位。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- 結果チャネルは、自動的に最初の引数チャネルの時間情報(サンプルレート)を引き継ぎます。そのため、すべて同じサンプルレートで取得したチャネルのみで計算する必要があります。

- 別の方法として、配列に基づいた純粋なVBA コードで計算を実行することもできます: EA_Test.GetChannelData でチャネルの内容を読み取り、すべてのデータポイントをループで計算を実行し、結果をEA_Test.CreateChannelで新規チャネルとして保存します。

- 以下の演算子および関数式を数式に使用することができます:
  +、-、/、、^*
  AND、OR、>=、<=、=
  sqrt、log、ln、exp
  sin、cos、tan、arctan
  ceil、floor、min、max
  特殊関数は微分および積分です:
  deriv(y channel、x channel)
  integral(y channel、x channel)

- グラフやDataViewウィンドウは、計算後に自動では更新しないことに注意してください！単一演算後に時間のかかる更新を行わないことにより、多くの計算が行えます。計算後の効果を表示するには、EA_Panel.RefreshAll、EA_Graph.RefreshAllやEA_DataView.RefreshAllを使用します。

- *ResultName*という名前の演算が存在している場合、既存の計算はこのメソッドの引数に応じて変更されます。

例

```
  Sub Avg()
    'Compute mean of pressure over many series of measurements.
    Dim Formula As String
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.Load "C:\MYDATA\Test2.TST"
    EA_Test.Load "C:\MYDATA\Test3.TST"
    EA_Test.Load "C:\MYDATA\Test4.TST"
    Formula = "1/4 * (Valve pressure[01] + Valve pressure[02] + Valve pressure[03] +
Valve pressure[04])"
    EA_Test.Calculator Formula, "Averaged pressure", "bar"
  End Sub
```

EA_Test.CANRawConvertToASCII

CANの生データをASCIIファイルに変換します。

```
  EA_Test.CANRawConvertToASCII(ByVal Test As Variant, ByVal CANRawChannel As Variant,
ByVal FileName As String) As Long
```

引数	説明
Test	テストのファイル名または位置。最初に読み込まれたテストの位置は1です。
CANRawChannel	CAN rawチャネルの名前または位置。テストの最初のチャネルの位置は1です。
FileName	変換するASCII ファイル名。

- CANの生データのサイズによっては、変換にかなりの時間がかかる場合があります。

EA_Test.CANRawDecodeSignal

CAN生データからデコードされたデータを作成します。

```
  EA_Test.CANRawDecodeSignal(ByVal Test As Variant, ByVal CANRawChannel As Variant,
ByVal dbcFile As String, ByVal SignalName As String, ByVal MsgName As String, ByVal
CreateTimechan As Integer, Optional ByVal TimeFormat As Integer) As Long
```

引数	説明
Test	テストのファイル名または位置。最初に読み込まれたテストの位置は1です。
CANRawChannel	CAN rawチャネルの名前または位置。テストの最初のチャネルの位置は1です。
dbcFile	CANデータベースのファイル名。
SignalName	信号名。
MsgName	メッセージ名。
CreateTimeChan	メッセージのタイムスタンプから時間チャネルを作成するかどうかを示します(0 = 作成しない、1 = 作成する)。
TimeFormat	オプションです。タイムスタンプの形式: 0 = 秒単位で、最初のメッセージは0から始まります 1 = 絶対NTPタイムスタンプ

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Test.ChannelAttribute

チャネルプロパティを読み込みます。チャネルは、テスト内のチャネル、または解析モードで作成した演算チャネルのいずれかです(この場合は`Test = 0` に設定します)。

```
  EA_Test.ChannelAttribute(ByVal Test As Variant, ByVal Channel As Variant, ByVal
AttribType As Integer, Attrib As Variant) As Long
```

引数	説明
`Test`	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 Testに空の文字列または0を指定した場合、チャネルは演算チャネルで検索します。最初の演算チャネルの位置は1です。演算チャネルの有効な`AttribType`は `1`、`2`、`3`、`5`、および`6`のみです。
`Channel`	チャネル名または位置。テストの最初のチャネルは1です。Test = 0の場合、最初の演算チャネルの位置は1です。
`AttribType`	属性の種類: `1` = チャネル内のデータ点数 `2` = チャネル名(位置からチャネルを指定している場合のみ) `3` = 単位 `4` = チャネルのコメント `5` = dt(ミリ秒) `6` = VBAの時刻形式(Now、DateSerial、TimeSerialを参照) `7` = アンプの種類 `8` = 接続プレートのタイプ(MGCplusのみ) `9` = 変換器回路 `10` = フィルタ特性 `11` = フィルタ周波数(Hz) `12` = アンプ + 接続ボードのシリアル番号 `13` = センサ名 `14` = センサID `15` = ゼロバランス値 `16` = 励起電圧 `17` = ゲージ率(kファクタ) `18` = スロット(プラグインの位置) `19` = サブチャネル(MGCplusのマルチチャネル、MX840のCANまたはMX471およびeDAQのアンプのみ) `20` = チャネル設定のXML記述(QuantumXのみ) `21` = センサの校正有効期限 `22` = センサの校正日 `23` = センサのシリアル番号 `24` = チャネル タイプ(テストのみ)。`0` = ハードウェア チャネル、`1` = catman時間チャネル、`2` = オンライン計算チャネル、`3` = エクスポートしたポストプロセス計算
`Attrib`	属性値。

戻り値	説明
`0`	エラーなし。
`<0`	エラー。

注記

- 演算チャネルにこのメソッドは使用できません。

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。
  EA_Test.DataDirを参照してください。

例

```
  Sub RemoveOutliers()
    'The example reads a channel from a test and executes a computation on the
channel values (elimination of freak values). The result is saved in a new channel.
    Dim lc As Long
    Dim Data() As Double
    Dim Count As Variant, Unit As Variant, dt As Variant, T0 As Variant
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.ChannelAttribute 1, "Valve pressure", 1, Count
    EA_Test.ChannelAttribute 1, "Valve pressure", 3, Unit
    EA_Test.ChannelAttribute 1, "Valve pressure", 5, dt
    EA_Test.ChannelAttribute 1, "Valve pressure", 6, T0
    ReDim Data(Count) As Double
    EA_Test.GetChannelData 1, "Valve pressure", Data
    For lc=1 To Count-1
      If Data(lc) > 1000 Then
        Data(lc)=Data(lc-1)
      End If
    Next
    EA_Test.CreateChannel "Valve pressure corrected", Unit, dt, T0, Count, Data
  End Sub
```

EA_Test.ClearExport

エクスポートするチャネルのリストを削除します。

```
EA_Test.ClearExport() As Long
```

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Test.CompressChannel

この関数は、チャネルから最小/最大圧縮データの配列を返します。この関数は、解析プロジェクトのチャネル(解析モード)、または一時ストア(DAQモード)のチャネルを操作できます。返されるx()配列には、圧縮された各y値に属するタイムスタンプ(秒単位)が含まれます。

注: この関数は、時間的に等距離にあるチャンネル、つまり固定サンプルレートでサンプリングされたチャンネルにのみ適用されます。

```
EA_Test.CompressChannel(ByVal Test As Variant, ByVal Channel As Variant, ByVal
Algorithm As Integer, ByVal CompressionFactor As Integer, ByVal StartIndex As Long,
ByVal Count As Long, NofPointsCompressed As Long, x() As Double, y() As Double) As
Long
```

引数	説明
Test	解析モード: テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 DAQモード: 無視されます。0に設定
Channel	チャネル名または位置。テストの最初のチャネルは1です。
Algorithm	適用する圧縮アルゴリズム: 0 = 等距離データ圧縮アルゴリズム その他 = Douglas-Peucker圧縮アルゴリズム
CompressionFactor	圧縮量を決定します。例: 元のデータ量(StartIndexとCountによって決定される)が100000サンプルの場合、*CompressionFactor*が100にすると、1000サンプルのブロックごとに1つの最小/最大値のペアが構築されます。つまり、圧縮サンプル数は200になります。
StartIndex	元のデータセット内の最初のサンプルのインデックス。最初のサンプルのインデックスは0から始まります。
Count	圧縮するサンプルの数。*Count* = 0の場合、データセット内のすべてのサンプルを考慮します。
NofPointsCompressed	生成した最小/最大値のペアの数を返します。
x	各yポイントのタイムスタンプ。配列は、*2*Count/CompressionFactor*サンプルを保持できる大きさにする必要があります。
y	圧縮されたデータ。配列は、*2*Count/CompressionFactor*サンプルを保持できる大きさにする必要があります。

戻り値	説明
0	エラーなし。
<0	エラー。テストまたはチャネルが存在しないなど。

EA_Test.ConvertASCIIFile

ASCIIデータファイルをcatman標準形式に変換します。

```
  EA_Test.ConvertASCIIFile(ByVal InFile As String, OutFile As String, ByVal
Separator As String, ByVal dt As Double, SpecialLines() As Integer, Optional ByVal
LineTerminator As Integer) As Long
```

引数	説明
InFile	ASCIIファイル名。
OutFile	変換後のファイル名(拡張子.BINの使用を推奨しています)。
Separator	データ列の区切り文字。例えばvbTabや";"。
dt	1/サンプルレート(ms)。不明の場合は0に設定します。
SpecialLines	線に特別な意味を割り当てる配列。すべての配列のインデックスは、それを適用する必要がある行番号を指定します。 インデックス0: すべての行番号配列要素をヘッダ情報として扱い、結果を.TSTファイルに書き出します。 インデックス1: この行番号の配列要素にはチャネル名が含まれています。 インデックス2: この行番号の配列要素にはチャネルの単位が含まれています。 インデックス3: この行番号の配列要素にはチャネルのコメントが含まれています。 インデックス4: この行番号の配列要素にはチャネルのサンプルレートが含まれています。 インデックス5: この行番号の配列要素にはファイルのコメントが含まれています。
LineTerminator	終了行。 LineTerminator = 0: 復帰文字と改行文字を組み合わせ(CR/LF)。それ以外は改行(LF)文字。

注記

- ASCIIファイルの大きさにより、変換に時間がかかることがあります。

EA_Test.ConvertFastStream

*FastStream*ファイルをcatman標準形式に変換します。

```
EA_Test.ConvertFastStream(ByVal InFile As String, OutFile As String) As Long
```

引数	説明
InFile	*FastStream*ファイル名。
OutFile	変換後のファイル名(拡張子.BINの使用を推奨しています)。catmanバイナリデータファイルとともに、テストファイル(.TST)を作成します。

注記

- *FastStream*ファイルの大きさにより、変換に時間がかかることがあります。

EA_Test.ConvertPNRFFile

PNRFデータファイルをcatman標準形式に変換します。

```
EA_Test.ConvertPNRFFile(ByVal InFile As String, OutFile As String) As Long
```

引数	説明
InFile	PNRFファイル名。
OutFile	変換後のファイル名(拡張子.BINの使用を推奨しています)。catmanバイナリデータファイルとともに、テストファイル(.TST)を作成します。

注記

- *FastStream*ファイルの大きさにより、変換に時間がかかることがあります。

EA_Test.ConvertSIEFile

eDAQ SIEデータファイルをcatman標準形式に変換します。

```
  EA_Test.ConvertSIEFile(ByVal InFile As String, OutFile As String, Optional
ByVal LongDescriptionAsChannelName As Integer, Optional ByVal ExtractVideoFrames
As Integer, Optional ByVal ExtractNMEALog As Integer, Optional ByVal
ExtractPlainTextData As Integer) As Long
```

引数	説明
InFile	SIEファイル名。
OutFile	変換後のファイル名(拡張子.BINの使用を推奨しています)。catmanバイナリデータファイルとともに、テストファイル(.TST)が作成されます。
LongDescriptionAsChannelName	オプションです。引数が1に設定されている場合、チャネル名に長い説明を使用します。デフォルト = 0。
ExtractVideoFrames	オプションです。引数が1に設定されている場合、ビデオフレームを別のファイルに保存します。デフォルト = 0。
ExtractNMEALog	オプションです。引数が1に設定されている場合、NMEAメッセージを別のテキストファイルに保存します。デフォルト = 0
ExtractPlainTextData	オプションです。引数が1に設定されている場合、テキストメッセージを別のテキストファイルに抽出します。デフォルト = 0

注記

- SIEファイルの大きさにより、変換に時間がかかることがあります。
- 変換により、TCE設定を含むファイルと、SIEファイルに含まれるログブックも作成されます。これらのファイルの名前はSIEファイルと同じですが、拡張子は.TCEおよび.LOGです。 これらは、*OutFile*で指定したcatmanバイナリデータファイルと同じディレクトリに存在します。どちらのファイルもテストファイル(.TST) 内で参照されます。

EA_Test.ConvertTimeChannel

時間チャネルのタイムスタンプを、OLEオートメーション日付からUnix日付形式へ、またはその逆に変換します。

```
  EA_Test.ConvertTimeChannel(ByVal Test As Variant, ByVal Channel As Variant, ByVal
FromFormat As eTimeFormats, ByVal ToFormat As eTimeFormats, ByVal ResultName As
String) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。
Channel	チャネル名または位置。テストの最初のチャネルは1です。
FromFormat	変換元の日付形式: 0 = Unix日付形式 1 = OLEオートメーション日付形式
ToFormat	変換後の日付形式: 0 = Unix日付形式 1 = OLEオートメーション日付形式
ResultName	結果のチャネル名。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルまたはテストが存在しないなど。

注記

- Unix時刻は、1970年1月1日(深夜0時UTC/GMT)からの秒です。
- OLEオートメーション時刻は、1899年12月30日深夜0時からの日数を値とする浮動小数点として実装されています。

EA_Test.Counting_JointTimeAtLevel

二つの時系列信号のレベル計数でジョイント時間を実行します。結果は、両方の信号が各クラスで費やした秒数を含む行列となります。

```
  EA_Test.Counting_JointTimeAtLevel(ByVal Test As Variant, ByVal Channel As Variant,
ByVal RefChannel As Variant, ByVal nClasses As Integer, ByVal ClassMin As Double,
ByVal ClassMax As Double, ByVal ClassMin2 As Double, ByVal ClassMax2 As Double, ByVal
ResultName As String, Optional ByVal ClassMinMaxFromChannel as Integer) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。
Channel	信号1のチャネル名または演算名(例えば速度)。
RefChannel	信号2のチャネル名または演算名(例えばRPM)。"reference"信号として表します。
nClasses	クラス数(4...255)。両方の信号は同数のクラスに分割されます。
ClassMin	信号1のクラス下限値。
ClassMax	信号1のクラス上限値。
ClassMin2	信号2のクラス下限値。
ClassMax2	信号2のクラス上限値。
ResultName	結果のチャネル名。
ClassMinMaxFromChannel	任意です。*ClassMin* および*ClassMax* を1に設定すると、*ClassMin2* および*ClassMax2* は無視され、代わりにチャネル値の最小値と最大値からクラスの下限値と上限値が導出されます。デフォルト = 0。

例

```
  Sub JointTimeAtLevel()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.Counting_JointTimeAtLevel 1, "pressure_1", "displacement_1", 64, -1, 1,
-1, 1, "PresDisp JointTimeAtLevel", 1
  End Sub
```

EA_Test.Counting_RainflowFromTo

時系列入力信号からレインフローのFromTo行列を計算します。

```
  EA_Test.Counting_RainflowFromTo(ByVal Test As Variant, ByVal Channel As Variant,
ByVal nClasses As Integer, ByVal ClassMin As Double, ByVal ClassMax As Double,
ByVal MinAmpl As Double, ByVal ClorSeg As Integer, ByVal SymFlag As Integer, ByVal
ResultName As String, Optional ByVal ClassMinMaxFromChannel as Integer, Optional
ByVal CloseOpenResiduals as Integer) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。
Channel	チャネル名または演算名。
nClasses	クラス数(4...255)。
ClassMin	クラス下限値。
ClassMax	クラス上限値。
MinAmpl	peak-valley検出のために考慮される最小振幅スパン(ゲート)(全クラス範囲の%)。
ClorSeg	1の場合、コールマンシーガ補正を適用します。デフォルトは0です。
SymFlag	1に設定した場合、同じクラス間の両方向においての遷移を集計します。デフォルトは0です。
ResultName	結果のチャネル名。
ClassMinMaxFromChannel	任意です。1に設定した場合、*ClassMin*および*ClassMax*は無視され、チャネル値の最小値および最大値を代わりに導出します。デフォルトは0です。
CloseOpenResiduals	任意です。1に設定した場合、不完全な剰余も集計します。デフォルトは0です。

例

```
  Sub Rainflow_FT()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.Counting_RainflowFromTo 1, "pressure_1", 64, -1, 1, 1, 0, 0, "Pressure
Rainflow_FT", 1, 1
  End Sub
```

EA_Test.Counting_RainflowRangeMean

時系列入力信号からレインフロー範囲平均行列を計算します。

```
  EA_Test.Counting_RainflowRangeMean(ByVal Test As Variant, ByVal Channel As Variant,
ByVal nClasses As Integer, ByVal ClassMin As Double, ByVal ClassMax As Double,
ByVal MinAmpl As Double, ByVal ClorSeg As Integer, ByVal SymFlag As Integer, ByVal
ResultName As String, Optional ByVal ClassMinMaxFromChannel as Integer, Optional
ByVal CloseOpenResiduals as Integer) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。
Channel	チャネル名または演算名。
nClasses	クラス数(4...255)。
ClassMin	クラス下限値。
ClassMax	クラス上限値。
MinAmpl	peak-valley検出のために考慮される最小振幅スパン(ゲート)(全クラス範囲の%)。
ClorSeg	1の場合、コールマンシーガ補正を適用します。デフォルトは0です。
SymFlag	1に設定した場合、同じクラス間の両方向においての遷移を集計します。デフォルトは0です。
ResultName	結果のチャネル名。
ClassMinMaxFromChannel	任意です。1に設定した場合、*ClassMin*および*ClassMax*は無視され、チャネル値の最小値および最大値を代わりに導出します。デフォルトは0です。
CloseOpenResiduals	任意です。1に設定した場合、不完全な剰余も集計します。デフォルトは0です。

例

```
  Sub Rainflow_RM()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.Counting_RainflowRangeMean 1, "pressure_1", 64, -1, 1, 1, 0, 0, "Pressure
Rainflow_RM", 1, 1
  End Sub
```

EA_Test.Counting_SpanPairs

時系列入力信号から、レンジペアの分類を計算します。

```
  EA_Test.Counting_SpanPairs(ByVal Test As Variant, ByVal Channel As Variant,
ByVal nClasses As Integer, ByVal ClassMin As Double, ByVal ClassMax As Double,
ByVal MinAmpl As Double, ByVal ClorSeg As Integer, ByVal SymFlag As Integer, ByVal
ResultName As String, Optional ByVal ClassMinMaxFromChannel As Integer, Optional
ByVal CloseOpenResiduals As Integer) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。
Channel	チャネル名または演算名。
nClasses	クラス数(4...255)。
ClassMin	クラス下限値。
ClassMax	クラス上限値。
MinAmpl	peak-valley検出のために考慮される最小振幅スパン(ゲート)(全クラス範囲の%)。
ClorSeg	1の場合、Clormann-Seegerを適用します。デフォルトは0です。
SymFlag	1に設定した場合、同じクラス間の両方向においての遷移を集計します。デフォルトは0です。
ResultName	結果のチャネル名。
ClassMinMaxFromChannel	任意です。1に設定した場合、*ClassMin*および*ClassMax*は無視され、チャネル値の最小値および最大値を代わりに導出します。デフォルトは0です。
CloseOpenResiduals	任意です。1に設定した場合、不完全な剰余も集計します。デフォルトは0です。

例

```
  Sub SpanPairs()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.Counting_SpanPairs 1, "pressure_1", 64, -1, 1, 1, 0, 0, "Pressure
SpanPairs", 1, 1
  End Sub
```

EA_Test.Counting_TimeAtLevel

時系列信号のレベル集計を実行します。結果は、信号が各クラスで費やした秒数を含む線形ベクトルとなります。

```
  EA_Test.Counting_TimeAtLevel(ByVal Test As Variant, ByVal Channel As Variant,
ByVal nClasses As Integer, ByVal ClassMin As Double, ByVal ClassMax As Double, ByVal
ResultName As String, Optional ByVal ClassMinMaxFromChannel as Integer) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。
Channel	チャネル名または演算名。
nClasses	クラス数(4...255)。
ClassMin	クラス下限値。
ClassMax	クラス上限値。
ResultName	結果のチャネル名。
ClassMinMaxFromChannel	任意です。1に設定した場合、*ClassMin*および*ClassMax*は無視され、チャネル値の最小値および最大値を代わりに導出します。デフォルトは0です。

例

```
  Sub TimeAtLevel()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.Counting_TimeAtLevel 1, "pressure_1", 64, -1, 1, "Pressure TimeAtLevel",
1
  End Sub
```

EA_Test.CreateChannel

演算グループ内に新規チャネルを作成し、渡されたデータで埋めます。その後、そのチャネルを視覚的な表示やさらなる計算に使用できます。この関数を使用して、既存のデータセットに新しいデータを設定することもできます。

このメソッドを使用すると、グラフでの可視化などのために、(EA_Mathなどを使用して)自分で作成(計算)したデータをcatmanEasy/AP標準関数で利用できるようにすることができます。

```
  EA_Test.CreateChannel(ByVal Name As String, ByVal Unit As String, ByVal dt As
Double, ByVal T0 As Double, ByVal SampleCount As Long, Data() As Double) As Long
```

引数	説明
Name	チャネル名 チャネル名は一意でなければなりません。
Unit	チャネルの単位。
dt	測定ポイント間の時間間隔値(ms)。この情報は時間に対するチャネルを表示するために使用されます。
T0	絶対日付/時刻形式のチャネル内の最初のデータポイントの時刻(VBA関数: Now)。この値は、日付/時刻形式でグラフの時間軸をスケーリングするために必要です。必要がない場合は、この値を0に設定してください。
SampleCount	*Data*内のデータポイント数。
Data	チャネルデータを含む配列。少なくとも*Count*要素が含まれている必要があります。最初の要素のインデックスは0です。 VBAは任意のサイズの配列を割り当てる(Dim)ことができないことに注意してください。配列は通常、64MBを超えると割り当てに失敗します。これは800万以上のサンプルが読み取れないことを意味します。EA_Test.SetChannelDataを使用して、より多くのサンプルをチャネルに設定します -このメソッドを使用すると、データを区分的に入力できます。

戻り値	説明
0	エラーなし。
<0	エラー。データバッファが作成できないなど。

注記

- *Name*という名前のチャネル(データセット)がすでに存在する場合、既存のデータセットは新規データセットに上書きされます。

- グラフやDataViewウィンドウは、計算後に自動では更新しないことに注意してください! 単一演算後に時間のかかる更新を行わないことにより、多くの計算が行えます。計算後の効果を表示するには、EA_Panel.RefreshAll、EA_Graph.RefreshAllやEA_DataView.RefreshAllを使用します。

例

```
  Sub RemoveOutliers()
    'The example reads a channel from a test and executes a computation on the
channel values (elimination of freak values). The result is saved in a new channel.
    Dim lc As Long
    Dim Data() As Double
    Dim Count As Variant, Unit As Variant, dt As Variant, T0 As Variant
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.ChannelAttribute 1, "Valve pressure", 1, Count
    EA_Test.ChannelAttribute 1, "Valve pressure", 3, Unit
    EA_Test.ChannelAttribute 1, "Valve pressure", 5, dt
    EA_Test.ChannelAttribute 1, "Valve pressure", 6, T0
    ReDim Data(Count) As Double
    EA_Test.GetChannelData 1, "Valve pressure", Data
    For lc=1 To Count-1
      If Data(lc) > 1000 Then
        Data(lc)=Data(lc-1)
      End If
    Next
    EA_Test.CreateChannel "Valve pressure corrected", Unit, dt, T0, Count, Data
  End Sub
```

EA_Test.CreateMatrix

演算グループ内に新規行列を作成し、渡されたデータで埋めます。その後、等高線グラフ(EA_Test.PlotMatrix)で視覚表示するために行列を使用することができます。

```
  EA_Test.CreateMatrix(ByVal Name As String, ByVal xUnit As String, ByVal yUnit As
String, ByVal zUnit As String, ByVal XCount As Long, ByVal YCount As Long, XData() As
Single, YData() As Single, ZData() As Single) As Long
```

引数	説明
Name	行列名。EA_Test.PlotMatrixで必要。 ⚠ 行列名は一意である必要があります。
xUnit	行列の列データ単位。等高線グラフで使用します。
yUnit	行列の行データ単位。等高線グラフで使用します。
zUnit	行列のセルデータ単位。等高線グラフで使用します。
XCount	列数。
YCount	行数。
XData	等高線グラフにX軸のラベルに使用するデータ配列。
YData	等高線グラフにY軸のラベルに使用するデータ配列。
ZData	Zデータ("標高"データ)の配列。この配列は最初の次元が行で、次の次元が列を示す二次元配列でなければなりません(例えばDim ZData(MaxRows,MaxCols)。

戻り値	説明
0	エラーなし。
<0	エラー。データバッファを作成できないなど。

注記

- *Name*が行列に存在する場合、既存の行列はこのメソッドの引数により変更されます。

- グラフやDataViewウィンドウは、計算後に自動では更新しないことに注意してください! 単一演算後に時間のかかる更新を行わないことにより、多くの計算が行えます。計算後の効果を表示するには、EA_Panel.RefreshAll、EA_Graph.RefreshAllやEA_DataView.RefreshAllを使用します。

EA_Test.CreatePeakValues

時間間隔や角度セクタごとにピーク値を含む新規のデータセットを作成します。

```
  EA_Test.CreatePeakValues(ByVal Test As Variant, ByVal Channel As Variant, ByVal
PeakMode As ePeakMode, ByVal AngleChannel As Variant, ByVal PeakType As ePeakType,
ByVal DownSampleRate As Single, ByVal TimeInterval As Single, ByVal StartAngle As
Integer, ByVal StopAngle As Integer, ByVal Interpolate As Integer, ByVal ResultName
As String) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。
Channel	チャネル名または位置。テストの最初のチャネルは1です。
PeakMode	ピーク値の計算方法: 0 = 角度セクタ。ピーク値は指定した角度範囲内を計算します。 1 =時間間隔。ピーク値は選択した時間間隔で計算します。
AngleChannel	角度計測のチャネル名または位置。これは*PeakMode* = 0の場合に必要です。
PeakType	ピークの種類: 0 = 最小値 1 = 最大値
DownSampleRate	結果のデータセットのために縮小したサンプルレート。
TimeInterval	ピーク値を計算する時間間隔。これはPeakMode = 1の場合に必要です。
StartAngle	角度範囲の開始。これは*PeakMode* = 0の場合に必要です。
StopAngle	角度範囲の終了。これは*PeakMode* = 0の場合に必要です。
Interpolate	1の場合、ソースチャネルのサンプルレートが信号帯域幅に対して低すぎると、より正確なピーク値を決定するため補間を使用します。
ResultName	結果のチャネル名。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルまたはテストが存在しないなど。

EA_Test.CurveSection

チャネルの一部を抽出し、抽出したセクションのみの新規データセットを作成します。セクションはポイントインデックス、時間値または周波数値のいずれかの、二つのX座標により定義されます。

```
  EA_Test.CurveSection(ByVal Test As Variant, ByVal Channel As Variant, ByVal
SectionType As Integer, ByVal X1 As Double, ByVal X2 As Double, ByVal ResultName As
String) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 Testに空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネル位置は1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
SectionType	セクションの定義: 0 = ポイントインデックス 1 = 時間値(チャネル開始からの相対秒) 2 = 周波数値(Hz)(チャネルに周波数スペクトルが含まれる場合にのみ妥当)
X1	Xセクションの下位座標。
X2	Xセクションの上位座標。
ResultName	結果のチャネル名。

戻り値	説明
0	エラーなし。
<0	エラー。結果チャネルが作成できないなど。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。
- EA_Test.CurveShiftも参照してください。

EA_Test.CurveShift

データセット(チャネル)をX軸に対してデータポイントを左または右方向に一定量シフトします。この関数は、非同期の共通のイベント(例えばデジタル回線状態の変化)が両方のデータセットにある場合に、二つのデータセットを同期させるために使用します。量は時間間隔で指定します。

```
  EA_Test.CurveShift(ByVal Test As Variant, ByVal Channel As Variant, ByVal
ShiftType As Integer, ByVal ShiftValue As Double, ByVal PaddingType As Integer, ByVal
PaddingValue As Double, ByVal ResultName As String) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 Testに空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネル位置は1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。(ポイント数によるシフト)
ShiftType	0 = SHIFTBYNUMBEROFPOINTS: ポイント数によるシフト 1 = SHIFTBYTIME: 時間によるシフト 2 = SHIFTBYFREQUENCY: 周波数によるシフト
ShiftValue	シフト量は*ShiftType*によります: *ShiftType*=0: ポイント数 *ShiftType*=1: 時間(s) *ShiftType*=2: 周波数(Hz) 量の値が負の場合、データセットは左に、そうでない場合は右にシフトします。
PaddingType	得られたデータセット長は、入力データセットと同じです。先頭および末尾のセクション(シフト方向による)は、二つの方法で埋めることができます。 R_NULLVALUX = 0 : グラフにNullValueを使用 R_MANUALVALUE = 1 : 指定した値を使用
PaddingValue	*PaddingType* = 1の場合、パディング値
ResultName	結果のチャネル名。

注記

- EA_Test.CurveSectionも参照してください。

例

```
  Sub ShiftSignal()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.CurveShift(1, "pressure_1", 0, 1000, 0, 0, "Pressure R_Shift") 'Shift in
the right direction.
    EA_Test.CurveShift(1, "pressure_1", 0, -1000, 0, 0, "Pressure L_Shift") 'Shift in
the left direction.
  End Sub
```

EA_Test.DataDir

DAQまたはテストデータのデフォルトディレクトリの設定または読み込み。パス無しでファイルを指定した場合、このディレクトリ内のファイルを探索します。

対話型モードでは、**Options**ダイアログボックスの**Directories**タブにある、DAQおよびテストデータディレクトリに相当します。

```
Property EA_Test.DataDir
```

EA_Test.Load、EA.ProjectDirも参照してください。

注記

- ここで指定したディレクトリは、ユーザにより変更することができます: **Data storage after DAQ job**が**Prompt**に設定されている場合、ジョブの終了ごとに保存するファイルダイアログを開きます。*EA_Test.DataDir*には、このダイアログで設定したパスが含まれます。

- ディレクトリが存在しない、またはディレクトリ名が無効の場合、古いディレクトリ名は変更せず、*EA.LastError*がエラーコード-27(不正なディレクトリ)を報告します。

例

```
Sub ExportData()
  EA_Test.DataDir = "C:\MYDATA\VALVE TESTS"
  EA_Test.Load "Test1.TST"
  EA_Test.AddToExport 1, 1, "Valve pressure"
  EA_Test.AddToExport 1, 1, "Temperature"
  EA_Test.AddToExport 1, 1, "Braking_force_left"
  'Export in EXCEL format
  EA_Test.Export("Test1.xls", 3, "A comment")
End Sub
```

EA_Test.DigFilt

チャネルを平滑化またはフィルタリングします。

```
EA_Test.DigFilt(ByVal Test As Variant, ByVal Channel As Variant, ByVal FiltType As Integer, ByVal F1 As Single, ByVal F2 As Single, ByVal ResultName As String, Optional ByVal Order As Integer, Optional ByVal PhaseCorr As Integer, Optional ByVal Ripple As Double, Optional ByVal Attenuation As Double) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 Testに空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネル位置は1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。(ポイント数によるシフト)
FiltType	フィルタの種類: 0 = 移動平均 1 = サビツキー・ゴーレイ平滑化 2 = バターワースローパスフィルタ 3 = バターワースハイパスフィルタ 4 = バターワースバンドパスフィルタ 5 = バターワースバンドストップフィルタ 6 = ベッセルローパスフィルタ(8次) 7 = 時間窓の平均値(*F1*でmsを指定)。さらに平均値の算出後に、*F2*(Hz)で指定したカットオフ周波数を持つローパスフィルタが適用されます。 8 = 時間窓のRMS値(*F1*でmsを指定)。さらにRMSの算出後に、*F2*(Hzで指定したカットオフ周波数を持つローパスフィルタが適用されます。 9 = チェビシェフローパスフィルタ 10=チェビシェフハイパスフィルタ 11=チェビシェフバンドパスフィルタ 12=チェビシェフバンドストップフィルタ 13= 楕円ローパスフィルタ 14=楕円ハイパスフィルタ 15=楕円バンドパスフィルタ 16=楕円バンドストップフィルタ 17 = dbA音圧レベルフィルタ 18 = ベッセルハイパスフィルタ 19 = ベッセルバンドパスフィルタ 20 = ベッセルバンドストップフィルタ 21 = 人体振動フィルタ Wb 22 = 人体振動フィルタ Wc 23 = 人体振動フィルタ Wd 24 = 人体振動フィルタ We 25 = 人体振動フィルタ Wf 26 = 人体振動フィルタ Wh 27 = 人体振動フィルタ Wj

	28 = 人体振動フィルタ Wk 29 = 人体振動フィルタ Wm 30 = 衝突解析CFC 60フィルタ(CFCはChannel Frequency Classの略称) 31 = 衝突解析CFC 180フィルタ 32 = 衝突解析CFC 600フィルタ 33 = 衝突解析CFC 1000フィルタ
F1	*FiltType* = 0または1: 平滑化ウィンドウ内のポイント数。デフォルト= 32 *FiltType* = 2: 低域カットオフ周波数(Hz) *FiltType* = 2: 低域カットオフ周波数(サンプルレートの%) *FiltType* = 3: 高域カットオフ周波数(Hz) *FiltType* = 4または5: 低帯域周波数(Hz) *FiltType* = 7または8: 時間窓(ms) *FiltType* = 9または13: 低域カットオフ周波数(Hz) *FiltType* = 10または14: 高域カットオフ周波数(Hz) *FiltType* = 11、12、15、16: 低帯域周波数(Hz)
F2	*FiltType* 4、5、11、12、15、16: 高帯域周波数(Hz) *FiltType* 7、8: ローパスカットオフ周波数(Hz)
ResultName	結果のチャネル名。
Order	フィルタの次数。有効範囲は4から32です。省略した場合、4を使用します。
PhaseCorr	1に設定した場合、フィルタによる位相シフトは逆フィルタリングにより除去されます。これにより、フィルタの次数を二倍にします!
Ripple	通貨帯域リップル(dB)。*チェビシェフ*および*楕円*フィルタにのみ必要。
Attenuation	阻止帯域減衰(dB)。*楕円*フィルタにのみ必要。

戻り値	説明
0	エラーなし。
<0	エラー。データバッファを作成できないなど。

注記

- *ResultName*が演算に存在する場合、既存の演算はこのメソッドの引数により変更されます。

- グラフやDataViewウィンドウは、計算後に自動では更新しないことに注意してください! 単一演算後に時間のかかる更新を行わないことにより、多くの計算が行えます。計算後の効果を表示するには、EA_Panel.RefreshAll、EA_Graph.RefreshAllやEA_DataView.RefreshAllを使用します。

- 配列ベースの*EA_Math.Filter*メソッドは、*チェビシェフ*や*楕円*フィルタなどの追加フィルタを提供します。

- 平均およびRMS関数は、1ステップの反復式の際に平均値およびRMS値を計算します:

  ```
  rms: Y(n)=sqrt( (1-a) * X(n) * X(n) + a * Y(n-1))
  Mean: Y(n)= (1-a) * X(n) + a * Y(n-1))
  ```

  ここでは`a = exp(-1/(FsInHz * TimeWinInSeconds))`としています。平均値やRMS値の算出後、*F2*(Hz)で指定したカットオフ周波数のローパスフィルタが、データセットに適用されます。

例

```
  Sub FilterSignal()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.DigFilt 1, "pressure_1", FILT_RUNNINGAVERAGE, 32, 0.01, "Pressure
filtered"
  End Sub
```

EA_Test.Export

エクスポートリストにある、すべてのチャネルをエクスポートします。EA_Test.AddToExportでリストを生成します。

```
  EA_Test.Export(ByVal FileName As String, ByVal Format As Integer, ByVal Comment As
String) As Long
```

引数	説明
FileName	エクスポートするファイル名。ファイルに拡張子が含まれていない場合、形式に応じて自動的に追加します。例えばExcel形式で.xls。 *Format* = 14(音声ファイル)および*Format* = 15(catman再生ファイル)の場合、チャネルごとに一つのファイルをチャネル名に基づいて作成します。*FileName*にはエクスポートフォルダのみを指定してください。
Format	ファイル形式: 0 = catman 1 = ASCII 2 = ASCII + チャネル情報 3 = Excel (Office XP/2003) 4 = Diadem 5 = nSoft DAC 6 = Vector MDF 3 7 = Excel (Office 2007 XLSX XML) 8 = Excel (Office 2007 XLSB Binary) 9 = Matlab 5.0 10 = RPC III 11 = nCode s3t 12 = ASAM MDF 4.1 13 = UFF58 14 = 音声ファイル(Wav) この形式は、チャネルごとに一つのファイルを作成します。*FileName*にはエクスポートフォルダのみを指定してください。 15 = catman再生ファイル(catwfm) この形式は、チャネルごとに一つのファイルを作成します。*FileName*にはエクスポートフォルダのみを指定してください。 16 = NI TDMS
Comment	コメント。 すべてのファイル形式がファイルのコメントをサポートしているわけではありません。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。

例

```
Sub ExportData()
  'Add single channels to export list.
  EA_Test.Load "C:\MYDATA\Test1.TST"
  EA_Test.AddToExport 1, 1, "Valve pressure"
  EA_Test.AddToExport 1, 1, "Temperature"
  EA_Test.AddToExport 1, 1, "Braking_force_left"
  'Add all channels to export list at once.
  EA_Test.Load "C:\MYDATA\Test2.TST"
  EA_Test.AddToExport 1, 0
  'Export in EXCEL format.
  EA_Test.Export("C:\MYDATA\Test2.xls", 3, "A comment")
End Sub
```

EA_Test.ExportChannelData

このメソッドは、単一チャネルまたは演算の数値データを、さまざまな形式にエクスポートします:

- ASCIIテキストファイル(.txt)
- 再生ファイル(.catwfm)
- 音声ファイル(.wav)
- Vector CANalyzerログ(.asc)
- PCAN-Trace(.trc)
- Vector BLF(.blf)

catmanは、上記のようにファイル拡張子に応じて形式を決定しています!

結果のファイル構造は、形式およびチャネルや演算の種類に依存しており、この関数の説明の後半で説明します。
複数チャネルをエクスポートするEA_Test.ExportMultipleChannelDataも参照してください。

インタラクティブモード(解析モード)では、これは解析プロジェクトリストのコンテキストメニュー"Export data"に対応します。

```
  EA_Test.ExportChannelData(ByVal ChanName As String, ByVal TestIndex As Integer,
ByVal FileName As String, Optional ByVal NumFormat as String, Optional ByVal
Delimiter As String)
```

引数	説明
Channel	チャネルまたは演算名。
TestIndex	チャネル内のテストのインデックス(最初のテストの位置は1です)。メソッドを演算に適用する場合、この引数は無視され、0に設定されます。
FileName	結果のテキスト ファイルの名前(パスを含む)。ファイル拡張子によって形式が決定: .txt = ASCIIテキストファイル .catwfm = catman再生ファイル .wav = 音声ファイル .asc = Vector CANalyzerログ .trc = PCAN Trace .blf = Vector BLF
NumFormat	オプションです。数値のASCII変換で使用される数値精度(小数点以下の桁数)。例: "0.00" "0.0000" "0"(整数) "Auto"(小数点以下の桁数は値の大きさから決定します) "Full"(フル64ビット精度、14桁) 省略した場合は小数点以下3桁を使用します。
Delimiter	オプションです。出力ファイルに複数の列 (クラス数行列、ジョイント周波数スペクトル行列、レベルでの時間、スパンペア)が含まれる場合、列を区切るために使用される区切り文字。省略した場合のデフォルトはTABです。

ASCII Text

チャネルや演算によるファイルの構造は以下の通りです:

データ型	ファイル構造
計測チャネル	CRLFで区切った数値の線形ベクトル。
オンライン演算チャネル(電卓、フィルタ、ロゼット)	CRLFで区切った数値の線形ベクトル。
演算: 電卓	CRLFで区切った数値の線形ベクトル。
演算: 周波数スペクトル	CRLFで区切った数値の線形ベクトル。
演算: ジョイント時間周波数スペクトル	複数列。列は*Delimiter*で区切り、行はCRLFで終了。 行1: 周波数 列1: 時間 セル(列、行)=振幅スペクトル
演算: フィルタ	CRLFで区切った数値の線形ベクトル。
演算: 外れ値の除去	CRLFで区切った数値の線形ベクトル。
演算: 区間(cut)	CRLFで区切った数値の線形ベクトル。
演算: レインフロー(FromTo)	複数列。列は*Delimiter*で区切り、行はCRLFで終了。 行1: クラス制限 (最も低いクラス順) 列1: クラス制限 (最も高いクラス順) セル(列、行)=カウント
演算: レインフロー(RangeMean)	複数列、列は*Delimiter*で区切り、行はCRLFで終了。 行1: 範囲値 列1: 平均値 セル(列、行)=カウント
演算: スパンペア	複数列、列は*Delimiter*で区切り、行はCRLFで終了。 列1: スパン幅 列2: カウント
演算: レベルでの時間	複数列、列は*Delimiter*で区切り、行はCRLFで終了。 列1: 振幅区分 列2: 振幅間隔に費やした時間(秒)
演算: レベルでのジョイント時間	複数列、列は*Delimiter*で区切り、行はCRLFで終了。 行1: クラス制限 列1: クラス制限参照チャネル セル(列、行)=秒単位の時間

catman再生ファイル

最適な再生能力を確保する再生ファイルの構造は、以下の通りです:

4 Byte DWORD	ファイルID (35014414)
4 Byte DWORD	最初のサンプルのバイト単位のオフセット
4 Byte DWORD	サンプル数
8 Byte DOUBLE	サンプルレート(Hz)
2 Byte WORD	チャネル名の文字数N
N Bytes CHAR	チャネル名
2 Byte WORD	チャネルの単位の文字数N
N Bytes CHAR	チャネルの単位
8 Byte DOUBLE	1. サンプル
8 Byte DOUBLE	2. サンプル
.....	.....

音声ファイル

このファイルの構造は、.WAV標準(RIFF)準拠です。

Vector CANalyzerログ, PCAN TraceおよびVector BLF

これらのファイル形式にエクスポートできるのは、CAN rawチャネルのみです。
標準のVector CANalyzer ASCIIログ形式では、各行は次のように構成されます。

```
Timestamp Busnumber CanId "Rx|Tx" "d" Number-of-bytes Byte1, Byte2, ..., Byte8
```

空白で区切ります。このファイル形式は、多くのCANツールおよびHBK GlyphWorksで読み取ることができます。
PCAN Traceログ形式は、いくつかの解析ツールでも読み取ることができます。
Vector BLFの場合、このバイナリログ形式のファイル圧縮は現在サポートされていません。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

EA_Test.ExportMultipleChannelData

このメソッドは、チャネルまたは演算の数値データを、ASCIIテキストファイルにエクスポートします。

このメソッドは信号ベクトル(つまり時系列)にのみ適用できますが、クラス集計チャネルや行列には適用できません。

このメソッドは、catman再生ファイル(.catwvfm)または音声ファイル(.wav)の作成には使用できません。

チャネルや行列の分類集計は、EA_Test.ExportChannelDataを参照してください。

インタラクティブモード(解析モード)では、これは解析プロジェクトリストのコンテキストメニュー"Export data"に対応

```
EA_Test.ExportMultipleChannelData(ByVal DataSource As String, ByVal FileName As String, Optional ByVal NumFormat as String, Optional ByVal Delimiter As String)
```

引数	説明
DataSource	以下のように"|"文字で区切ります。"Channel name|Test index|Channel name|Test index....." (例: "Pressur|1|Oil temperature|1|Wheel force|1.....") テストインデックスは、チャネルの位置を示します(最初のテストのインデックスは1)。
FileName	結果のテキストファイル名(パスを含む)。
NumFormat	オプションです。数値のASCII変換で使用される数値精度(小数点以下の桁数)。例: "0.00" "0.0000" "0"(整数) "Auto"(小数点以下の桁数は値の大きさから決定します) "Full"(フル64ビット精度、14桁) 省略した場合はAutoを使用します。
Delimiter	オプションです。出力ファイルに複数の列が含まれる場合、列を区切るために使用される区切り文字。省略した場合のデフォルトはTABです。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルまたは演算チャネルが存在しないなど。

EA_Test.ExtractBitChannel

入力チャネルの特定ビットを調査し、新規チャネルを作成します。結果チャネルにはビットが設定されていない場合は0、そうでない場合は1 が含まれます。

```
  EA_Test.ExtractBitChannel(ByVal Test As Variant, ByVal Channel As Variant, ByVal
BitPos As Integer, ByVal ResultName As String) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
BitPos	0...31
ResultName	結果のチャネル名。

戻り値	説明
0	エラーなし。
<0	エラー。データバッファが作成できないなど。

注記

- 入力チャネルの値は、最初に長整数型(LONG INTEGER)に変換されます。

EA_Test.FindEvent

チャネル内の指定したイベント(例えば5Vより大きい最初の振幅)の位置を検索します。位置はインデックスとして返します。チャネルの*dt*を使用して(EA_Test.ChannelAttributeを参照)時間のポイントを計算することができます。

```
  EA_Test.FindEvent(ByVal Test As Variant, ByVal YChannel As Variant, ByVal XChannel
As Variant, ByVal StartIndex As Long, ByVal ConditionID As Integer, Param() As
Double, Position As Long, Optional ByVal StopIndex As Long) As Long
```

引数	説明
Test	テスト名または位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
YChannel	チャネル名またはyチャネルの位置。テストの最初のチャネルの位置は1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
XChannel	xデータが必要なイベントのみに必要(傾斜チェックなど)。他の場合は0または""を指定します。
StartIndex	検索を開始する場所のインデックス。最初のデータポイントは1です。
ConditionID	0 = *Param(0)*で指定したレベルよりも大きい最初の値の位置 1 = *Param(0)*で指定したレベルよりも小さい最初の値の位置) 2 = 前の値よりも*Param(0)*だけ大きい最初の値の位置 3 = 前の値よりも*Param(0)*だけ小さい最初の値の位置 4 = 傾きが*Param(0)*以上の最初の位置 5 = 傾きが*Param(0)*未満の最初の位置) 6 = 指定したレベルで一定の偏差以上にならなくなる(平坦)、連続したサンプル数(n)を検索。返す位置のインデックスは平坦の始まりです。 7 = 最大値を検索。最大値は*Param(0)*に返します。 8 = 最小値を検索。最小値は*Param(0)*に返します。
Param	*ConditionID*により異なります: *ConditionID* 0-3: Param(0) = レベル *ConditionID* 4-5: Param(0) = 傾き *ConditionID* 6: Param(0) = レベル、Param(1) = 偏差、Param(2) = サンプル数 *ConditionID* 7および8: Param(0) 最小および最大値を含む
Position	イベント位置のインデックスを返します。最初のデータポイントは1です。イベントが検出されなかった場合、*Position*はゼロです。 返すインデックスは、常にチャネルデータの先頭を基準とし、*StartIndex*を基準とするものではありません。
StopIndex	任意です。指定した場合、検索は*StopIndex*のポイント番号で終了します。0または省略した場合、チャネルの最後まで検索を実行します。指定する場合は、*StartIndex*より大きくなければなりません。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルまたはテストが存在しないなど。

注記

- このメソッドは、数百万サンプルの非常に大規模なデータセットに最適化されています。
- このメソッドはEA_Test.CurveSection(大規模なデータセットから一部を切り取り)や、EA_Test.GetChannelSectionと組み合わせて使用すると便利です。

例

```
Dim vPar
Dim dt As Double
Dim t As Double
Dim Pos As Long
Dim param(0) As Double
EA_Test.Load "C:\MYDATA\Test1.TST"
EA_Test.ChannelAttribute 1, "Shock", CHAN_DT, vPar
dt=vPar/1000
param(0) = 1500
EA_Test.FindEvent 1, "Shock", "", 1, EV_FIRST_GREATER, param, Pos
t=Pos*dt
MsgBox CStr(t)
```

EA_Test.GetACDCLevels

入力チャネルのACおよびDC推定値を計算します。

```
  EA_Test.GetACDCLevels(ByVal Test As Variant, ByVal Channel As Variant, ACLevel As
Double, DCLevel As Double) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。
Channel	チャネル名または位置。テストの最初のチャネルは1です。
ACLevel	入力チャネルのACレベル
DCLevel	入力チャネルのDCレベル

戻り値	説明
0	エラーなし。
<0	エラー。チャネルまたはテストが存在しないなど。

EA_Test.GetChannelData

読み込まれたテストのチャネルや演算チャネルからバッファにデータを読み取ります。

```
  EA_Test.GetChannelData(ByVal Test As Variant, ByVal Channel As Variant, Data() As
Double) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
Data	チャネルデータを収容する配列。データ配列の最初の値はインデックス0で始まります。 ⚠ 配列は、チャネルのすべての値を受け入れられるよう、十分な大きさでなければなりません。まずEA_Test.ChannelAttributeで値数を検索します。 ⚠ VBAは配列の任意のサイズを割り当てる(Dim)ことができないことに注意してください。配列は通常、64MBを超えると割り当てに失敗します。これは800万以上のサンプルが読み取れないことを意味します。データブロックの処理にEA_Test.GetChannelSectionを使用します。

戻り値	説明
>=0	読み取られたデータポイント数。
<0	エラー。データバッファが作成できないなど。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。
- このメソッドは常に完全なチャネルの内容を読み取ります。EA_Test.GetChannelSectionを使用するとデータセットの一部のみを取得できます。非常に大規模なデータセットを扱う場合に便利です。
- このメソッドは、例えばJTFスペクトルのような行列を提供する計算に適用させることはできません。

例

```
  Sub RemoveOutliers()
    'The example reads a channel from a test and executes a computation on the
channel values (elimination of freak values). The result is saved in a new channel.
    Dim lc As Long
    Dim Data() As Double
    Dim Count As Variant, Unit As Variant, dt As Variant, T0 As Variant
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.ChannelAttribute 1, "Valve pressure", 1, Count
    EA_Test.ChannelAttribute 1, "Valve pressure", 3, Unit
    EA_Test.ChannelAttribute 1, "Valve pressure", 5, dt
    EA_Test.ChannelAttribute 1, "Valve pressure", 6, T0
    ReDim Data(Count) As Double
    EA_Test.GetChannelData 1, "Valve pressure", Data
    For lc=1 To Count-1
      If Data(lc) > 1000 Then
        Data(lc)=Data(lc-1)
      End If
    Next
    EA_Test.CreateChannel "Valve pressure corrected", Unit, dt, T0, Count, Data
  End Sub
```

EA_Test.GetChannelSection

EA_Test.GetChannelDataと似ていますが、チャネルや演算データの一部のみを取得します。

```
  EA_Test.GetChannelSection(ByVal Test As Variant, ByVal Channel As Variant, ByVal
StartIndex As Long, ByVal NofPoints As Long, Data() As Double) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
StartIndex	セクションの最初のポイントのインデックス。チャネルまたは計算の最初のポイントのインデックスは1です。
NofPoints	*StartIndex*から始まる、取得するポイント数。
Data	チャネルデータを収容する配列。 ⚠ 配列は、*NofPoints*の値を受け入れられるよう、十分な大きさでなければなりません。

戻り値	説明
>=0	読み取られたデータポイント数。
<0	エラー。データバッファが作成できないなど。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。
- このメソッドは、例えばJTFスペクトルのような行列を提供する計算に適用させることはできません。

例

```
  Sub RemoveOutliers()
    'The example reads a channel data section from a test and executes a computation
on the channel section values (elimination of freak values). The result is saved in a
new channel.
    Dim lc As Long
    Dim Data() As Double
    Dim Num_Values As Long, Start_Index As Long
    Num_Values = 1000 'Section length.
    Start_Index = 500 'Start index of section.
    Dim Count As Variant, Unit As Variant, dt As Variant, T0 As Variant
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.ChannelAttribute 1, "Valve pressure", 1, Count
    EA_Test.ChannelAttribute 1, "Valve pressure", 3, Unit
    EA_Test.ChannelAttribute 1, "Valve pressure", 5, dt
    EA_Test.ChannelAttribute 1, "Valve pressure", 6, T0
    If Count >= Start_Index+Num_Values Then 'Check that enough points are available.
      'Read channel data section of a fixed length of 1000 values, starting at the
500th value.
      ReDim Data(Num_Values)As Double
      EA_Test.GetChannelSection 1, "Valve pressure", Start_Index, Num_Values, Data
      For lc=1 To Num_Values-1
        If Data(lc) > 1000 Then
             Data(lc)=Data(lc-1)
        End If
      Next
      EA_Test.CreateChannel "Valve pressure corrected_section", Unit, dt, T0, Num_
Values, Data
    End If
  End Sub
```

EA_Test.GetDataFileChanCount

このメソッドは、最初に解析プロジェクトを読み込まなくても、catman標準の.BINファイルに直接アクセスできるメソッドのグループに属しています。ファイルへの直接アクセス方法を使用すると、DAQモードで既存の計測データを解析することもできます。
参照: EA_Test.GetDataFileInfo

この関数は、.BINファイルに含まれているチャネル数を返します。

```
  EA_Test.GetDataFileChanCount(ByVal FileName As String, ChanCount As Integer) As
Long
```

引数	説明
FileName	パスを含む.BINファイル名。ファイルには拡張子.BIN が必要であることに注意してください。そうでない場合、メソッドはエラーを返します。
ChanCount	ファイルに含まれているチャネル数。

戻り値	説明
0	エラーなし。
<0	エラー。.BINファイルが存在しないなど。

例

```
  Sub DirectAccessData()
    Dim RetVal As Long
    Dim FileName As String
    Dim ChanCount As Integer
    Dim ChanName As String
    Dim ChanOffset As Double
    Dim SampleCount As Long
    Dim Buf() As Double

    FileName = "C:\Users\HBM\Documents\HBM\catmanEasy\DATA\Job1.bin"
    RetVal = EA_Test.GetDataFileChanCount(FileName, ChanCount)

    ReDim ParamList(ChanCount-1, 64) As Variant
    RetVal = EA_Test.GetDataFileInfo(FileName, ParamList)

    ChanName = ParamList(1,0)
    RetVal = EA_Test.ShowDataFileChanPreview(FileName, ChanName)

    ChanOffset = ParamList(1,5)
    SampleCount = ParamList(1,2)
    ReDim Buf(SampleCount-1)
    RetVal = EA_Test.GetDataFileData(FileName, ChanOffset, 0, SampleCount, Buf)

  End Sub
  tDataFileData(FileName, ChanOffset, 0, SampleCount, Buf)
  End Sub
```

EA_Test.GetDataFileData

このメソッドは、最初に解析プロジェクトを読み込まなくても、catman標準の.BINファイルに直接アクセスできるメソッドのグループに属しています。ファイルへの直接アクセス方法を使用すると、DAQモードで既存の計測データを解析することもできます。
参照: EA_Test.GetDataFileChanCount 、EA_Test.GetDataFileInfo

この関数は、.BINデータファイルから1つのチャンネルのサンプルを取得します。すべてのサンプルを取得することも、サンプルの一部だけを取得することもできます(チャネル長が長い場合など)。

```
  EA_Test.GetDataFileData(ByVal FileName As String, ByVal ChanOffset As Double, ByVal
FirstSampleIndex As Long, ByVal SampleCount As Long, Buf() As Double) As Long
```

引数	説明
FileName	パスを含む.BINファイル名。ファイルには拡張子.BIN が必要であることに注意してください。そうでない場合、メソッドはエラーを返します。
ChanOffset	チャネルデータが始まる位置へのバイト単位のオフセット。このオフセットは、EA_Test.GetDataFileInfoを介して取得されます。
FirstSampleIndex	取得する最初のサンプルのインデックス。チャネルの最初のサンプルのインデックスは0です。
SampleCount	*FirstSampleIndex*から取得するサンプル数。チャネルのサンプルの合計数は、EA_Test.GetDataFileInfoを介して取得されます。
Buf	サンプル データを受け取る倍精度浮動小数点数型(Double)の配列。*SampleCount*要素を保持できる十分な大きさである必要があります。

戻り値	説明
0	エラーなし。
<0	エラー。.BINファイルが存在しないなど。

例

```
  Sub DirectAccessData()
    Dim RetVal As Long
    Dim FileName As String
    Dim ChanCount As Integer
    Dim ChanName As String
    Dim ChanOffset As Double
    Dim SampleCount As Long
    Dim Buf() As Double

    FileName = "C:\Users\HBM\Documents\HBM\catmanEasy\DATA\Job1.bin"
    RetVal = EA_Test.GetDataFileChanCount(FileName, ChanCount)

    ReDim ParamList(ChanCount-1, 64) As Variant
    RetVal = EA_Test.GetDataFileInfo(FileName, ParamList)

    ChanName = ParamList(1,0)
    RetVal = EA_Test.ShowDataFileChanPreview(FileName, ChanName)

    ChanOffset = ParamList(1,5)
    SampleCount = ParamList(1,2)
    ReDim Buf(SampleCount-1)
    RetVal = EA_Test.GetDataFileData(FileName, ChanOffset, 0, SampleCount, Buf)
  End Sub
```

EA_Test.GetDataFileInfo

このメソッドは、最初に解析プロジェクトを読み込まなくても、catman標準の.BINファイルに直接アクセスできるメソッドのグループに属しています。ファイルへの直接アクセス方法を使用すると、DAQモードで既存の計測データを解析することもできます。
参照: EA_Test.GetDataFileChanCount、EA_Test.GetDataFileData

この関数は、.BINファイルに含まれるすべてのチャネルのヘッダ情報を取得します。

```
EA_Test.GetDataFileInfo(ByVal FileName As String, ParamList As Variant) As Long
```

<table>
<tr><th>引数</th><th>説明</th></tr>
<tr><td>FileName</td><td>パスを含む.BINファイル名。ファイルには拡張子.BIN が必要であることに注意してください。そうでない場合、メソッドはエラーを返します。</td></tr>
<tr><td>ParamList</td><td>各チャネルのヘッダ情報を返すバリアントの2次元配列。配列は、各チャネルごとに64個の要素を保持できる十分な大きさでなければなりません。例:
<code>Dim ParamList(500,64) As Variant</code>

通常、最初の次元にはEA_Test.GetDataFileChanCounで取得したチャネル数を使用します。チャネルインデックスは0から始まります。つまり、最初のチャネル情報は ParamList(0,n) にあります。

<table>
<tr><th>ParamListのインデックス</th><th>内容</th></tr>
<tr><td>0</td><td>チャンネル名</td></tr>
<tr><td>1</td><td>チャンネル単位</td></tr>
<tr><td>2</td><td>サンプル数。この値はEA_Test.GetDataFileDataに必要です。</td></tr>
<tr><td>5</td><td>チャネルデータのバイトオフセット。この値はEA_Test.GetDataFileDataに必要です。この値を必ず Double 型の変数にコピーしてください。</td></tr>
<tr><td>6</td><td>サンプルレートからのdt(ミリ秒単位)。Fs = 1/dt * 1000 Hz。</td></tr>
<tr><td>7</td><td>OLE 時間形式での計測開始時刻(1900年1月1日からの日数)。</td></tr>
<tr><td>8</td><td>センサ名。</td></tr>
<tr><td>9</td><td>センサID。</td></tr>
<tr><td>14</td><td>ひずみゲージ計測に使用するkファクタ。</td></tr>
<tr><td>18</td><td>ゼロ調整バッファの現在値。</td></tr>
<tr><td>19</td><td>無負荷 バッファの現在値。</td></tr>
<tr><td>20</td><td>アンプのシリアル番号。</td></tr>
<tr><td>21</td><td>ハードウェアスロット番号。</td></tr>
<tr><td>22</td><td>サブチャネル、単一チャネルスロットの場合は0。</td></tr>
<tr><td>23</td><td>入力特性x1。</td></tr>
<tr><td>24</td><td>入力特性y1。</td></tr>
<tr><td>25</td><td>入力特性x2。</td></tr>
<tr><td>26</td><td>入力特性y2。</td></tr>
<tr><td>27</td><td>物理単位(ユーザスケーリングが有効な場合はユーザ単位)。</td></tr>
<tr><td>28</td><td>固有単位。</td></tr>
<tr><td>30</td><td>スケーリングタイプ。0=エンジニアリングユニット、1=電気。</td></tr>
<tr><td>36</td><td>ひずみゲージ計測で使用するブリッジ係数。</td></tr>
<tr><td>43</td><td>校正日。</td></tr>
<tr><td>44</td><td>校正有効期限。</td></tr>
</table>
</td></tr>
</table>

戻り値	説明
0	エラーなし。
<0	エラー。.BINファイルが存在しないなど。

例

```
Sub DirectAccessData()
  Dim RetVal As Long
  Dim FileName As String
  Dim ChanCount As Integer
  Dim ChanName As String
  Dim ChanOffset As Double
  Dim SampleCount As Long
  Dim Buf() As Double

  FileName = "C:\Users\HBM\Documents\HBM\catmanEasy\DATA\Job1.bin"
  RetVal = EA_Test.GetDataFileChanCount(FileName, ChanCount)

  ReDim ParamList(ChanCount-1, 64) As Variant
  RetVal = EA_Test.GetDataFileInfo(FileName, ParamList)

  ChanName = ParamList(1,0)
  RetVal = EA_Test.ShowDataFileChanPreview(FileName, ChanName)

  ChanOffset = ParamList(1,5)
  SampleCount = ParamList(1,2)
  ReDim Buf(SampleCount-1)
  RetVal = EA_Test.GetDataFileData(FileName, ChanOffset, 0, SampleCount, Buf)

End Sub
```

EA_Test.GetExportListCount

エクスポートリスト内のチャネル数を取得します。

```
EA_Test.GetExportListCount(Count As Integer)
```

引数	説明
Count	エクスポートリスト内のチャネル数。

注記

- エクスポートリスト内の各チャネルの詳細(チャネル名、テストインデックス)を取得するには、EA_Test.GetExportListItemを使用します。

例

```
Sub GetExportChanInfo()
  Dim i As Integer
  Dim chanCount As Integer
  Dim chanName As String, testIndex As Integer
  EA_Test.GetExportListCount chanCount
  For i = 1 To chanCount
    EA_Test.GetExportListItem i, chanName, testIndex
  Next
End Sub
```

EA_Test.GetExportListItem

エクスポートリスト内のチャネル名およびテストインデックスを取得します。

```
  EA_Test.GetExportListItem(ByVal Index As Integer, CName As String, TestIndex As
Integer)
```

引数	説明
Index	エクスポートリスト内のチャネルインデックス。
CName	チャネル名。
TestIndex	テストインデックス。

注記

- エクスポートリスト内のチャネル数を取得するには、EA_Test.GetExportListCountを使用します。

例

```
  Sub GetExportChanInfo()
    Dim i As Integer
    Dim chanCount As Integer
    Dim chanName As String, testIndex As Integer
    EA_Test.GetExportListCount chanCount
    For i = 1 To chanCount
      EA_Test.GetExportListItem i, chanName, testIndex
    Next
  End Sub
```

EA_Test.GetSample

チャネルの単一の値を返します。

```
  EA_Test.GetSample(ByVal Test As Variant, ByVal Channel As Variant, ByVal
SampleIndex As Long, Sample As Double) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
SampleIndex	値を取得する位置。チャネル内の最初の値の位置は1です。
Sample	値。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。
- チャネルの単一サンプルの*変更*にはEA_Test.SetSampleを使用します。

例

```
  Sub ShowMax()
  'Retrieves the index of the maximum in one channel and displays the value of
another channel at this position.
    Dim StatData(8) As Double
    Dim MaxIndex As Long
    Dim Sample As Double
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.GetStatistics 1, "pressure_1", StatData
    MaxIndex = StatData(4)
    EA_Test.GetSample 1, "temperature_1", MaxIndex, Sample
    MsgBox CStr(Sample)
  End Sub
```

EA_Test.GetStatistics

チャネルの統計データを返します。

```
  EA_Test.GetStatistics(ByVal Test As Variant, ByVal Channel As Variant, Results()
As Double) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネルは1です。Test = 0の場合、最初の演算チャネルの位置は1です。
Results	少なくとも9要素の配列が必要です。この関数は配列要素内の統計データを返します: Results(0) = 最小値 Results(1) = 最小値のインデックス Results(2) = 最小値の時間(チャネルの開始値からの相対秒) Results(3) = 最大値 Results(4) = 最大値のインデックス Results(5) = 最大値の時間(チャネルの開始値からの相対秒) Results(6) = 平均値 Results(7) = 標準偏差 Results(8) = RMS

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。

例

```
Sub ShowMax()
  Dim StatData(8) As Double
  EA_Test.Load "C:\MYDATA\Test1.TST"
  EA_Test.GetStatistics 1, "pressure_1", StatData
  MsgBox CStr(StatData(3))
End Sub
```

EA_Test.Interpolate

補間点数に基づいて、ソースチャネル(yチャネル)から新規のデータセットを作成します。補間点は、新規サンプルレートまたはxチャネルを参照する開始値、終了値およびステップ幅を指定することによって決定します。

```
  EA_Test.Interpolate(ByVal Test As Variant, ByVal Mode As Integer, ByVal Method As
Integer, ByVal YChannel As Variant, ByVal XChannel As Variant, ByVal NewSampleRate
As Single, ByVal XStart As Double, ByVal XEnd As Double, ByVal XStepWidth As Double,
ByVal ResultName As String) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Mode	補間点のモードを選択: 0 = 新規サンプルレート 1 = xチャネル
Method	補間方法: 1 = 線形 2 = 多項式 3 = 有理 4 = スプライン
YChannel	yチャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
XChannel	xチャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。 この引数は*Mode* = 0の場合、この引数は無視されます!
NewSampleRate	*Mode* = 0の場合のみ有効: 補間点を決定する新規サンプルレート。
XStart	*Mode* = 1の場合のみ有効: 補間点の開始値。
XEnd	*Mode* = 1の場合のみ有効: 補間点の終了値。
XStepWidth	*Mode* = 1の場合のみ有効: 補間点のステップ幅。
ResultName	結果のチャネル名。

戻り値	説明
0	エラーなし。
<0	エラー。データバッファが作成できないなど。

注記

- 演算は*ResultName*がすでに存在する場合、このメソッドの引数に応じて変更されます。

- グラフやDataViewウィンドウは、自動的に更新されないことに注意してください! 単一の計算後に時間のかかる更新を行わなことにより、多くの計算が行えます。計算後の効果を表示するには、EA_Panel.RefreshAll、EA_Graph.RefreshAllやEA_DataView.RefreshAllを使用します。

- 補間に基づいた新規サンプルレートは、異なるサンプルレートでチャネルを計算/比較して記録するために必要です。典型的な例: 角度信号の高速オーバーサンプリング、圧力や力チャネルの低速記録、その後、角度対圧力のy(x)をプロット。二つ目は補間点のx開始値、x終了値およびステップ幅により、等距離補間点を作成します。この方法は、さらにxチャネルが必要になります。典型的な例: 角度(x)および圧力(y)を記録。その結果、圧力値を含む新規データセットが作成されます(元の角度データは、回転サイクルと同期しておらず、固定サンプルレートで記録されるため、これらの値を正確に含むことはありません)。

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。

例

```
  Sub AngleWithNewSampleRate()
    'Creates a new angle value channel dataset. The new dataset corresponds to the
signal being sampled with a rate of 200 Hz.
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.Interpolate 1, 0, 1, "Angle", 0, 200, 0, 0, 0, "Angle_200"
  End Sub
```

EA_Test.JointTimeFrequencySpectrum

チャネルのジョイント時間周波数スペクトルを計算します。ジョイント時間周波数領域解析は、周波数成分の時間の経過に伴う変化に関する情報を取得します。結果はEA_Test.PlotMatrixメソッドにより、等高線や3Dグラフに表示することができます。

```
  EA_Test.JointTimeFrequencySpectrum(ByVal Test As Variant, ByVal Channel As Variant,
ByVal TimeWindow As Single, ByVal FFTPoints As Long, ByVal PercentOverlap As Single,
ByVal AvgMode As Integer, ByVal RemoveDCComponent As Integer, ByVal WindowType As
Integer, ByVal ScaleType As Integer, ByVal OutputUnit As Integer, ByVal ResultName As
String) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
TimeWindow	単一のスペクトルを計算する時間。元の時系列信号を分割して、時間間隔数を決定します。各間隔の単一スペクトルを計算します。この引数がゼロ(デフォルト)の場合、時間間隔数は*FFTPoints*数から決定します。
FFTPoints	単一FFTのポイント数。
PercentOverlap	単一FFTオーバーラップの%。
AvgMode	単一FFT平均化の種類: 0 = リニア 1 = ピークホールド
RemoveDCComponent	0 = DC成分を削除しない 1 = DC成分を削除
WindowType	窓関数: 0 = なし 1 = 三角 2 = ハニング 3 = ハミング 4 = ブラックマン 5 = 完全ブラックマン 6 = ブラックマンハリス 7 = フラットトップ 8 = コサインテーパ 9 = スケーリング: ユニフォーム 10 = スケーリング: ハニング 11 = スケーリング: ハミング 12 = スケーリング: ブラックマンハリス 13 = スケーリング: 完全ブラックマン 14 = スケーリング: ブラックマン 15 = スケーリング: フラットトップ 16 = スケーリング: 4項ブラックマンハリス 17 = スケーリング: 7項ブラックマンハリス

ScaleType	スペクトルのスケール: 0 = スケーリングしない 1 = リニア 2 = dB 3 = dBm
OutputUnit	スペクトル出力単位: 0 = Amp. rms 1 = Amp. peak 2 = Amp. rms^2 3 = Amp. peak^2 4 = Amp. rms/Hz平方根 5 = Amp. peak/Hz平方根 6 = Amp. 二乗rms/Hz 7 = Amp. 二乗peak/Hz
ResultName	結果のチャネル名。

戻り値	説明
0	エラーなし。
<0	エラー。データバッファが作成できないなど。

注記

- 選択した時間ウィンドウ、周波数分解能とサンプルレート、多くの部分スペクトルに応じて計算されます(ウィンドウあたり)。
- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。
- 演算は*ResultName*がすでに存在する場合、このメソッドの引数に応じて変更されます。
- これらスペクトルは、列が周波数で行が時間に相当する行列を形成します。
- グラフやDataViewウィンドウは、自動的に更新されないことに注意してください! 単一の計算後に時間のかかる更新を行わなことにより、多くの計算が行えます。計算後の効果を表示するには、EA_Panel.RefreshAll、EA_Graph.RefreshAllやEA_DataView.RefreshAllを使用します。
- このメソッドは、数百万サンプルの非常に大規模なデータセットに最適化されています。

例

```
  Sub ComputeJTF()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.JointTimeFrequencySpectrum 1, "pressure_1", 0, 2048, 67, 0, 0, 0, 0, 0,
"Pressure_JTF"
    EA_Test.PlotMatrix "Pressure_JTF", 1, "CONTOURGRAPH_1"
  End Sub
```

EA_Test.LastTestExported (Read only)

最後に保存したテストパラメータファイルのフルパスを返します。

```
Property EA_Test.LastTestExported
```

注記

- 最後のDAQジョブの計測データを未だ保存していない場合、空の文字列を返します。

EA_Test.Load

解析プロジェクトでテストを読み込みます。データにアクセスする前にテストを読み込む必要があります。

```
EA_Test.Load(ByVal FileName As String) As Long
```

引数	説明
FileName	テスト名(ファイル拡張子.TST)。*FileName*が空の文字列の場合、最後のDAQジョブのデータ(利用可能な場合)を読み込みます。

戻り値	説明
0	エラーなし。
<0	エラー。ファイルが存在しないなど。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。

- 自動的に最後のDAQジョブのデータを読み込むには、*FileName*に空の文字列を指定します。したがって、DAQジョブの完了後にEA.OperationModeで、DAQモードから解析モードに切り替えるとデータがすぐに利用可能になります。

例

```
Sub ExportData()
  'Add single channels to export list.
  EA_Test.Load "C:\MYDATA\Test1.TST"
  EA_Test.AddToExport 1, 1, "Valve pressure"
  EA_Test.AddToExport 1, 1, "Temperature"
  EA_Test.AddToExport 1, 1, "Braking_force_left"

  'Add all channels to export list at once.
  EA_Test.Load "C:\MYDATA\Test2.TST"
  EA_Test.AddToExport 1, 0
  'Export in EXCEL format.
  EA_Test.Export("C:\MYDATA\Test2.xls", 3, "A comment")
End Sub
```

EA_Test.LoadCANTraceFile

CANの生トレースファイルを読み込みます。

```
EA_Test.LoadCANTraceFile(ByVal FileName As String) As Long
```

引数	説明
FileName	の生トレースファイルのファイル名。

戻り値	説明
0	エラーなし。
<0	エラー。ファイルが存在しないなど。

EA_Test.MakePlaybackFileFromASCII

ASCIIテキストファイルから、データの再生ファイルを作成します。このファイルには、以下の構造である必要があります。

```
Channel name CRLF
Channel unit CRLF
Sample rate in Hz CRLF
Sample 1 CRLF
Sample 2 CRLF
Sample 3 CRLF
.......
........
```

ピリオド同様、コンマも小数点記号として許可されます。空白行は無視されます。

```
EA_Test.MakePlaybackFileFromASCII(ByVal ASCIIFile As String, ByVal PlaybackFile As String) As Long
```

引数	説明
ASCIIFile	ASCIIファイル名。
PlaybackFile	拡張子.catwfmの再生ファイル名。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- 再生ファイルには、単一チャネルのサンプルが含まれています。これらのサンプルは、DAQジョブで取得したデータと並行して再生することができます。
- ファイル拡張子に.catwfmを指定することで、EA_Test.ExportChannelDataより、再生ファイルを作成することもできます。

EA_Test.MathChanCount

解析プロジェクトで現在利用可能な計算の合計数を返します。

```
Property EA_Test.MathChanCount
```

このプロパティは、プロジェクトのすべての計算を反復処理する場合に役立ちます:

```
Dim Attrib As Variant
Dim i As integer
Dim NofMathChannels As integer
Dim NofSamples As integer

NofMathChannels = EA_Test.MathChanCount

For i=1 To NofMathChannels
  EA_Test.ChannelAttribute 0,i,CHAN_SRCLEN,Attrib
  NofSamples=Attrib
  Redim Data(NofSamples) As Double
  EA_Test.GetChannelData 0,i,Data()
  .....

  .....
Next
```

EA_Test.MergeFiles

複数のcatman標準のデータファイルを一つのデータファイルにマージします。

このメソッドは、チャネル名およびサンプルレートが同一で、チャネル数が同数のマージされるファイルが必要です。

```
  EA_Test.MergeFiles(ByVal MergeFileList As String, ByVal ResultFile As String) As
Long
```

引数	説明
MergeFileList	セミコロンで区切ったファイル名のリスト(すなわち、"C:\My Documents\HBM\Data\Test1.bin;C:\My Documents\HBM\Data\Test2.bin;C:\My Documents\HBM\Data\Test3.bin")。
ResultFile	結果のファイル名。

戻り値	説明
0	エラーなし。
<0	エラー。
-102	少なくとも二つのファイルでチャネル数が異なります。
-103	少なくとも二つのファイルでチャネル名が異なります。
-104	少なくとも二つのファイルの同一チャネルでサンプルレートが異なります。
-105	ファイルからデータの読み込み時にエラーが発生しました。
-106	マージされたチャネルの少なくとも一つが長すぎます(> 2000000000値)。
-107	マージ後のファイルサイズがターゲットドライブの空き用容量を超える可能性があります。
-108	チャネルの種類が異なります(例えば信号ベクトルとレインフロー行列)。
-109	行列の長さが異なります。
-110	結果のファイル名が指定されていません。

EA_Test.PhaseSpectrum

チャネルの位相スペクトルを計算します。Δfがチャネル情報に保存されるため、結果チャネルを周波数と直接比較して表示することができます。

ジョイント時間周波数スペクトルの計算(EA_Test.JointTimeFrequencySpectrum)も参照してください。

```
  EA_Test.PhaseSpectrum(ByVal Test As Variant, ByVal Channel As Variant, ByVal
FFTPoints As Long, ByVal PercentOverlap As Single, ByVal AvgMode As Integer, ByVal
RemoveDCComponent As Integer, ByVal WindowType As Integer, ByVal ResultName As
String, ByVal CreateFreqChan as Integer) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
FFTPoints	単一FFTのポイント数。
PercentOverlap	単一FFTオーバーラップの%。
AvgMode	単一FFT平均化の種類: 0 = リニア 1 = ピークホールド
RemoveDCComponent	0 = DC成分を削除しない 1 = DC成分を削除
WindowType	窓関数: 0 = なし 1 = 三角 2 = ハニング 3 = ハミング 4 = ブラックマン 5 = 完全ブラックマン 6 = ブラックマンハリス 7 = フラットトップ 8 = コサインテーパ 9 = スケーリング: ユニフォーム 10 = スケーリング: ハニング 11 = スケーリング: ハミング 12 = スケーリング: ブラックマンハリス 13 = スケーリング: 完全ブラックマン 14 = スケーリング: ブラックマン 15 = スケーリング: フラットトップ 16 = スケーリング: 4項ブラックマンハリス 17 = スケーリング: 7項ブラックマンハリス

ResultName	結果のチャネル名。
CreateFreqChan	該当する周波数値を保持する追加チャネルを作成: 0 = 周波数チャネルを作成しない 1 = 周波数チャネルを作成

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- 演算は*ResultName*がすでに存在する場合、このメソッドの引数に応じて変更されます。
- グラフやDataViewウィンドウは、自動的に更新されないことに注意してください! 単一の計算後に時間のかかる更新を行わなことにより、多くの計算が行えます。計算後の効果を表示するには、EA_Panel.RefreshAll、EA_Graph.RefreshAllやEA_DataView.RefreshAllを使用します。
- 選択した周波数分解能およびサンプリングレートに応じて、多くの部分スペクトルを計算した後、平均化します。
- このメソッドは、数百万サンプルの非常に大規模なデータセットに最適化されています。

例

```
  Sub ComputeSpectrum()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.AmplitudeSpectrum 1, "pressure_1", 1024, 67, 1, 0, 2, "Amplitude spectrum
of pressure", 0
  End Sub
```

EA_Test.PlotMatrix

等高線グラフに行列をプロットします(EA_Test.CreateMatrixやEA_Test.JointTimeFrequencySpectrumで作成)。

```
  EA_Test.PlotMatrix(ByVal MatrixName As String, ByVal Panel As Variant, ByVal
GraphName As String) As Long
```

引数	説明
MatrixName	行列名。
Panel	パネル名または位置。テストの最初のパネルは1です。*Panel* = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	等高線グラフ名。

戻り値	説明
0	エラーなし。
<0	エラー。データバッファが作成できないなど。

例

```
  Sub ComputeJTF()
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.JointTimeFrequencySpectrum 1, "pressure_1", 0, 2048, 67, 0, 0, 0, 0, 0,
"Pressure_JTF"
    EA_Test.PlotMatrix "Pressure_JTF", 1, "CONTOURGRAPH_1"
  End Sub
```

EA_Test.PlotSpectrogram

チャネルまたは演算を、スペクトログラムの新規プロットとして追加します。

```
  EA_Test.PlotSpectrogram(ByVal Test As Variant, ByVal Channel As Variant, ByVal
Panel As Variant, ByVal GraphName As String) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。
Channel	チャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
Panel	パネル名または位置。テストの最初のパネルは1です。*Panel* = 0の場合、*すべてのパネル*のグラフを検索することを意味します。
GraphName	スペクトログラム名。

戻り値	説明
0	エラーなし。
<0	エラー。

EA_Test.PlotYT

グラフのプロットにチャネルを追加します。チャネル値は、チャネル情報に格納されている時間情報(サンプルレート)に対して表示されます。

```
EA_Test.PlotYT(ByVal Test As Variant, ByVal Channel As Variant, ByVal Panel As Variant, ByVal GraphName As String, ByVal Axis As Integer, Optional ByVal Color As Long) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
Panel	パネル名または位置。テストの最初のパネルは1です。*Panel* = 0の場合、*すべて*のパネルのグラフを検索することを意味します。
GraphName	グラフ名。
Axis	チャネルを表示するy軸レイヤを示します: 0 = 最初のレイヤ(デフォルト) 1 = 二番目のレイヤ 2 = 三番目のレイヤ 3 = 右軸を基準にした最初のレイヤ 4 = 右軸を基準にした二番目のレイヤ 5 = 右軸を基準にした三番目のレイヤ ⚠ グラフは適切なレイヤ数のために、対話的に設定する必要があります: EasyScriptは現在、グラフのレイヤ構成をサポートしていません。
Color	RGBの色相。 色相はVBAで定義された定数のいずれか(vbBlack、vbWhite、vbBlue、vbRed、vbGreen、vbCyan、vbMagenta、vbYellow)、または赤、緑および青の組み合わせRGB(R,G,B)になります。

戻り値	説明
>=0	読み込んだデータ点数。
<0	エラー。データバッファが作成できないなど。

EA_Panel.SetValueも参照してください。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。
- 他のチャネルに対して表示する場合、EA_Test.PlotYXを使用します。
- プロットの更なる属性の定義(凡例や線種など)にはEA_Graph.SetPlotPropertyを使用します。

例

```
Sub Plot()
  EA_Test.Load "C:\MYDATA\Test1.TST"
  EA_Graph.ClearPlots 1, "PPGRAPH_1"
  EA_Test.PlotYT 1, "pressure_1", 1, "PPGRAPH_1", 0, vbRed
End Sub
```

EA_Test.PlotYX

y(x)グラフまたは極座標のプロットにチャネルまたは演算を追加します。極座標の場合、xチャネルのデータは角度軸に使用されます。

```
  EA_Test.PlotYX(ByVal YTest As Variant, ByVal XTest As Variant, ByVal YChannel
As Variant, ByVal XChannel As Variant, ByVal Panel As Variant, ByVal GraphName As
String, ByVal Axis As Integer, Optional ByVal Color As Long) As Long
```

引数	説明
YTest	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 YTestに空きの文字列または0を指定した場合、演算チャネルyを検索します。最初の演算チャネルの位置は1です。
XTest	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 XTestに空きの文字列または0を指定した場合、演算チャネルxを要求します。最初の演算チャネルの位置は1です。
YChannel	yチャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
XChannel	xチャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
Panel	パネル名または位置。テストの最初のパネルは1です。*Panel* = 0の場合、すべてのパネルのグラフを検索することを意味します。
GraphName	グラフ名。
Axis	チャネルを表示するy軸レイヤを示します: 0 = 最初のレイヤ(デフォルト) 1 = 二番目のレイヤ 2 = 三番目のレイヤ 3 = 右軸を基準にした最初のレイヤ 4 = 右軸を基準にした二番目のレイヤ 5 = 右軸を基準にした三番目のレイヤ ⚠ グラフは適切なレイヤ数のために、対話的に設定する必要があります: EasyScriptは現在、グラフのレイヤ構成をサポートしていません。
Color	RGBの色相。 色相はVBAで定義された定数のいずれか(vbBlack、vbWhite、vbBlue、vbRed、vbGreen、vbCyan、vbMagenta、vbYellow)、または赤、緑および青の組み合わせRGB(R,G,B)になります。

戻り値	説明
>=0	読み込んだデータ点数。
<0	エラー。データバッファが作成できないなど。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。
- 時間に対するチャネルを表示する場合、EA_Test.PlotYTを使用します。
- プロットの更なる属性の定義(凡例や線種など)にはEA_Graph.SetPlotPropertyを使用します。

例

```
Sub Plot()
  EA_Test.Load "C:\MYDATA\Test1.TST"
  EA_Graph.ClearPlots 1, "PPGRAPH_1"
  EA_Test.PlotYX 1, 1, "pressure_1", "force_1", 1, "PPGRAPH_1", 0, vbRed
  EA_Graph.SetAxisProperty 1, "PPGRAPH_1", -1, AX_CAPTION, "Force" 'Axis caption.
  EA_Graph.SetAxisProperty 1, "PPGRAPH_1", -1, AX_UNIT , "N" 'Unit
End Sub
```

EA_Test.Recalc

すべて、または個々の演算を再計算します。これは例えば、新規テストの読み込み後に実行します。

```
EA_Test.Recalc(Optional ByVal CompName As String) As Long
```

引数	説明
CompName	任意です。演算名。引数を省略した場合、すべての演算を再計算します。

戻り値	説明
0	エラーなし。
<0	エラー。演算が存在しないなど。

例

```
Sub ComputeSpectrum()
  EA_Test.Load "C:\MYDATA\Test1.TST"
  EA_Test.AutoPowerSpectrum 1, "pressure_1", 0.01, 0, 0, 0, "Pressure frequency
spectrum"
  'Load new test and repeat computation.
  EA_Test.Load "C:\MYDATA\Test2.TST"
  EA_Test.Recalc
End Sub
```

EA_Test.Remove

解析プロジェクトからテストを削除します。

```
  EA_Test.Remove(ByVal Test As Variant, Optional ByVal NoDispSourceAndCompRemoving
As Integer) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。
NoDispSourceAndCompRemoving	オプションです。 0(デフォルト) = テストを参照する表示ソースと演算も削除します。 それ以外のすべて = テストを参照する表示ソースと演算は削除されません。

戻り値	説明
0	エラーなし。
<0	エラー。テストが存在しないなど。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。
- テストを削除すると、そのデータは使用できなくなります。このテストの演算やチャネルが含まれていたグラフは、無効となります。
- オプションの引数*NoDispSourceAndCompRemoving*は、同様のチャネルを含む多数のテストを分析する必要がある場合に役立ちます。可視化と演算はテンプレートとして残り、EA_Test.Load経由で新しいテストを読み込むと、それらが自動的に更新されます。

例

```
'Remove the specified test from the analysis project.
EA_Test.Remove "C:\MYDATA\Test1.TST"
```

EA_Test.RemoveAllComputations

現在の解析プロジェクトから、すべての演算チャネルを削除します。

```
EA_Test.RemoveAllComputations() As Long
```

戻り値	説明
0	エラーなし。
<0	エラー。解析モードでのみ操作が許可されているなど。

EA_Test.RemoveChannel

現在の解析プロジェクトから、演算チャネルを削除します。

```
EA_Test.RemoveChannel(ByVal Name As String) As Long
```

引数	説明
Name	演算チャネル名。

戻り値	説明
0	エラーなし。
<0	エラー。演算が存在しないなど。

注記

- 対話的、またはEA_Test.CreateChannelで作成した、新規演算チャネルのみを削除することができます。テストファイルに含まれている演算は、削除できません。
- チャネルが削除されると、データは失われます。
- ロゼットゲージの計算では、メイン項目のみを削除できます(すべてのサブ項目を含む)。対話型モードの場合、単一の計算、例えば主公称応力を含まない角度のみなどは削除できない場合があります。

EA_Test.RemoveOutliers

データセットから外れ値を削除し、選択した値に置き換えます。外れ値は、前後の値が指定した閾値よりも大きな値です。また、外れ値として扱う閾値を上回る値数を指定することができます。結果として得られるチャネルは、ソースチャネルと同じ長さになります。

```
  EA_Test.RemoveOutliers(ByVal Test As Variant, ByVal Channel As Variant, ByVal
Threshold As Double, ByVal NofOutlierValues As Integer, ByVal ReplaceValueType, ByVal
ResultName As String, Optional ByVal RemoveMethod As Integer) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
Threshold	外れ値の閾値を指定。
NofOutlierValues	外れ値として扱う閾値を上回る数(通常は1)
ReplaceValueType	置き換える値の種類: 0 = グラフのNull値(グラフに表示されません) 1 = 最後の有効値
ResultName	結果のチャネル名。
RemoveMethod	任意です。外れ値の決定方法: 0 = 前値との差が一定値を超える(閾値) 1 = すべての値で一定以上 2 = すべての値で一定未満

戻り値	説明
0	エラーなし。
<0	エラー。データバッファが作成できないなど。

注記

- *ResultName*という名前の演算が存在している場合、既存の計算はこのメソッドの引数に応じて変更されます。
- グラフやDataViewウィンドウは、自動的に更新されないことに注意してください! 単一の計算後に時間のかかる更新を行わなことにより、多くの計算が行えます。計算後の効果を表示するには、EA_Panel.RefreshAll、EA_Graph.RefreshAllやEA_DataView.RefreshAllを使用します。
- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。

例

```
  Sub OutlierRemoval()
    'Removes all values with distance greater then 10 to preceding and succeeding
value and replaces them with the last valid value.
    EA_Test.Load "C:\MYDATA\Test1.TST"
    EA_Test.RemoveOutliers 1, "pressure_1", 10, 1, 1, "Adjusted pressure"
  End Sub
```

EA_Test.RemoveTableDatasources

データ、統計、トレースデータやメタデータテーブルからすべてのチャネルを削除します。

```
  EA_Test.RemoveTableDatasources(ByVal Panel As Variant, ByVal TableName As String)
As Long
```

引数	説明
Panel	パネル名または位置。テストの最初のパネルは1です。*Panel* = 0の場合、*すべてのパネルのテーブル*を検索することを意味します。
TableName	テーブル名。

戻り値	説明
>=0	データソースは正常に削除されます。
<0	エラー。すなわち、テーブルが見つからないか型が違います。

EA_Panel.SetValueも参照してください。

注記

- テーブルにチャネルを追加するには、EA_Test.AddTableDatasourceを使用します。

例

```
Sub Plot()
  EA_Test.Load "C:\MYDATA\Test1.TST"
  EA_Test.RemoveTableDatasources 1, "DATATABLE_1"
  EA_Test.AddTableDatasource 1, "pressure_1", 1, "DATATABLE_1"
  EA_Test.AddTableDatasource 1, "pressure_2", 1, "DATATABLE_1"
End Sub
```

EA_Test.RenameComputation

解析プロジェクト内の演算チャネル名を変更します。

```
EA_Test.RenameComputation(ByVal Name As String, ByVal NewName As String) As Long
```

引数	説明
Name	変更したい演算チャネルの現在の名前。
NewName	演算チャネルに割り当てる新しい名前。

戻り値	説明
0	エラーなし。
<0	エラー。演算チャネルが存在しないか、同じ名前のチャネルがすでに存在します。

EA_Test.ResolvePlaceholdersInFileName

このメソッドは、データを保存する前、テストファイル名に%Job%のようにプレースフォルダで置換して使用してすることができます。catmanは通常、組み込みの保存ルーチンで自動的にこれを実行します(ジョブ後自動的に)。しかし、スクリプト内で独自のデータを保存する目的として、テストファイル名が必要になるシナリオが発生する可能性があります。この場合、ファイルを開く/作成する前にプレースフォルダを解消するように注意する必要があります。

```
  EA_Test.ResolvePlaceholdersInFileName(ByVal InFile As String, OutFile As String)
As Long
```

引数	説明
InFile	プレースフォルダを含む元のファイル名。
OutFile	実際の内容(例えばジョブ名のような)によってプレースフォルダで置換したファイル名。

戻り値	説明
0	エラーなし。
<0	エラー。無効なInFileなど。

EA_Test.SelectedFiles

解析モードのData Explorerで、選択したファイルのリストを取得します。

```
EA_Test.SelectedFiles() As String
```

戻り値	このメソッドは、カンマ区切り(;)でファイル名を返し、ファイルが選択されていない場合、空の文字列を返します。

例

```
Sub GetSelectedTestFiles()
  Dim i As Integer, nofFiles As Integer
  Dim s As String, files() As String
  s=EA_Test.SelectedFiles
  files=Split(s, ";") 'Semicolon is used as separator.
  nofFiles=UBound(files)
  For i=0 To nofFiles
    If files(i) <> "" Then MsgBox files(i)
  Next
End Sub
```

EA_Test.SetChannel_dt

計測チャネルの値間の時間間隔を変更します。

⚠ この関数は、計測チャネルのみに適用されます。

```
  EA_Test.SetChannel_dt(ByVal Test As Variant, ByVal Channel As Variant, ByVal dt As
Double) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。
Channel	チャネル名または位置。テストの最初のチャネルは1です。
dt	計測点間の時間間隔の新しい値(ms)。この情報は時間に対するチャネルを表示するために使用されます。

戻り値	説明
0	エラーなし。
<0	エラー。テストまたはチャネルが存在しないなど。

EA_Test.SetChannelData

EA_Test.CreateChannelで作成しておいた演算チャネル(すなわちデータセット)にデータを入力します。視覚化表示や演算にチャネルを使用することができます。

このメソッドを使用することにより、グラフでの視覚化などのため(EA_Mathなどで)、作成(計算)したデータを catmanEasy/AP標準関数で利用できるようにすることができます。

```
  EA_Test.SetChannelData(ByVal Name As String, ByVal SampleCount As Long, Data() As
Double, Optional ByVal StartIndex as Double) As Long
```

引数	説明
Name	チャネル名 チャネルが存在している必要があります。EA_Test.CreateChannelを使用して、新規のデータセットを作成します。
SampleCount	*Data*内のデータポイント数。
Data	チャネルデータの配列。少なくとも*Count*要素を含める必要があります。最初の要素のインデックスは0です。 VBAは配列の任意のサイズを割り当てる(Dim)ことができないことに注意してください。配列は通常、64MBを超えると割り当てに失敗します。これは800万以上のサンプルが読み取れないことを意味します。この目的のために*SetChannelData*は、引数*StartIndex*で巨大なデータセットを区分して実装させます。
StartIndex	任意です(省略した場合0)。データセットを区分して実装します。*Data()*バッファの最初の項目は、データセットの*StartIndex*の位置が書き込まれます。*StartIndex*はサンプル数(バイトではなく)です。StartIndex+Countが、現在のデータセット長を超える場合、自動的に長さを拡張します。 データセット内の最大サンプル数が、2^32を越えられないことに注意してください。

戻り値	説明
0	エラーなし。
<0	エラー。データバッファが作成できないなど。

注記

- グラフやDataViewウィンドウは、計算後に自動では更新しないことに注意してください! 単一演算後に時間のかかる更新を行わないことにより、多くの計算が行えます。計算後の効果を表示するには、EA_Panel.RefreshAll、EA_Graph.RefreshAllやEA_DataView.RefreshAllを使用します。

EA_Test.SetSample

チャネルの単一の値を修正(上書き)します。これは例えば、外れ値の除去に使用します。

⚠ このメソッドは、テストの*元データ*も変更します。

```
  EA_Test.SetSample(ByVal Test As Variant, ByVal Channel As Variant, ByVal
SampleIndex As Long, ByVal NewValue As Double) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
Channel	チャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
SampleIndex	上書きする位置。チャネル内の最初の値は1です。
NewValue	新規の値。

戻り値	説明
0	エラーなし。
<0	エラー。チャネルが存在しないなど。

注記

- このメソッドは、チャネルにサンプルを追加することはできません - チャネル長は計測により決定します。このメソッドは例えば、外れ値の除去に使用します。
- チャネルから単一サンプルを読み取るにはEA_Test.GetSampleを使用します。
- この関数は、2バイト整数型で保存しているチャネルには適用されません!

EA_Test.ShowDataFileChanPreview

このメソッドは、最初に解析プロジェクトを読み込まなくても、catman標準の.BINファイルに直接アクセスできるメソッドのグループに属しています。ファイルへの直接アクセス方法を使用すると、DAQモードで既存の計測データを解析することもできます。
参照: EA_Test.GetDataFileInfo

この機能は、チャネルプレビューダイアログを表示します。チャネルプレビューウィンドウには、信号のグラフィック表示とチャネルの最も重要なトレーサビリティ データが含まれています。

💡 この関数を繰り返し呼び出すと、複数のプレビューを同時に開くことができます。プレビューウィンドウを閉じるかどうかはユーザの判断です。

```
  EA_Test.ShowDataFileChanPreview(ByVal FileName As String, ByVal ChanName As String)
As Long
```

引数	説明
FileName	パスを含む .BIN ファイルの名前。ファイルには拡張子.BINが必要であることに注意してください。そうでない場合、メソッドはエラーを返します。
ChanName	チャンネル名。これは、EA_Test.GetDataFileInfoから取得できます。

戻り値	説明
0	エラーなし。
<0	エラー。.BINが存在しないなど。

例

```
  Sub DirectAccessData()
    Dim RetVal  As Long
    Dim FileName As String
    Dim ChanCount As Integer
    Dim ChanName As String
    Dim ChanOffset As Double
    Dim SampleCount As Long
    Dim Buf() As Double

    FileName = "C:\Users\HBM\Documents\HBM\catmanEasy\DATA\Job1.bin"
    RetVal = EA_Test.GetDataFileChanCount(FileName, ChanCount)

    ReDim ParamList(ChanCount-1, 64) As Variant
    RetVal = EA_Test.GetDataFileInfo(FileName, ParamList)

    ChanName = ParamList(1,0)
    RetVal = EA_Test.ShowDataFileChanPreview(FileName, ChanName)

    ChanOffset = ParamList(1,5)
    SampleCount = ParamList(1,2)
    ReDim Buf(SampleCount-1)
    RetVal = EA_Test.GetDataFileData(FileName, ChanOffset, 0, SampleCount, Buf)
  End Sub
```

EA_Test.StrainAnalysis

例えばロゼットタイプのひずみゲージのひずみチャネルから、公称応力やせん断応力などの様々な量を計算します。

```
  EA_Test.StrainAnalysis(ByVal Test As Variant, ByVal ChannelA As Variant, ByVal
ChannelB As Variant, ByVal ChannelC As Variant, ByVal RosType As Integer, ByVal YMod
As Double, ByVal PoisRatio As Double, ByVal StressUnit As String, ByVal TSensA As
Double, ByVal TSensB As Double, ByVal TSensC As Double, ByVal ResMask As Integer,
ByVal BaseName As String) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
ChannelA	チャネル名または位置。テストの最初のチャネルは1です。*Test* = 0の場合、最初の演算チャネルの位置は1です。
ChannelB	二番目のひずみチャネル名または位置。
ChannelC	三番目のひずみチャネル名または位置。
RosType	ロゼットの種類: 45 = 45°/90° 60 = 60°/120° 90 = 90° 二軸 1 = 単一ひずみゲージ
YMod	ヤング率。
PoisRatio	ポアソン比
StressUnit	応力チャネルの単位。
TSensA	計測グリッドaの横感度(%)
TSensB	計測グリッドbの横感度(%)
TSensC	計測グリッドcの横感度(%)
ResMask	生成する演算チャネルのビットマスク。生成したい結果チャネルの整数を追加します。: 1 = 角度 2 = 主公称応力1 4 = 主公称応力2 8 = 応力X 16 = 応力Y 32 = せん断応力 64 = 参照応力(フォン・ミーゼスによる) 128 = 主ひずみ1 256 = 主ひずみ2 512 = ひずみX 1024 = ひずみY 2048 = せん断ひずみ 例: 71(=1+2+4+64)を指定すると、角度、主公称応力1、主公称応力2、および参照応力(フォン・ミーゼスによる)を計算します。
BaseName	演算チャネルのベース名。

戻り値	説明
0	エラーなし。
<0	エラー。ひずみチャネルが存在しない、またはロゼットの種類が存在しないなど。

注記

- 計算は*ResultName*という名前の演算が存在している場合、既存の計算はこのメソッドの引数に応じて変更されます。
- グラフやDataViewウィンドウは、計算後に自動では更新しないことに注意してください！ 単一演算後に時間のかかる更新を行わないことにより、多くの計算が行えます。計算後の効果を表示するには、EA_Panel.RefreshAll、EA_Graph.RefreshAllやEA_DataView.RefreshAllを使用します。
- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。
- *90° 二軸*ロゼットは3番目のひずみチャネルを無視します。*単一ひずみゲージ*の計算では、2番目と3番目のチャネルは無視されます。一方で、計算を機能させるには、有効なチャネルを指定する必要があります。
- *単一ひずみゲージ*の計算では、結果として主公称応力1のみが提供されます。*ResMask*は無視されます。

EA_Test.TestAttribute

テストプロパティを読み取ります。

```
  EA_Test.TestAttribute(ByVal Test As Variant, ByVal AttribType As Integer, Attrib
As Variant, Optional ByVal ParIndex As Integer) As Long
```

引数	説明
Test	テスト名(ファイル拡張子.TST)またはチャネルが始まるテストの位置。最初に読み込まれたテストの位置は1です。 ⚠ 最初にテストを読み込む必要があります。 *Test*に空きの文字列または0を指定した場合、演算チャネルを検索します。最初の演算チャネルの位置は1です。
AttribType	属性の種類: 1 = テスト名 2 = 計測データのファイル名 3 = ファイル内のチャネル数 4 = 計測ファイルのコメント 10 = テストパラメータ数 11 = 位置*ParIndex*のテストパラメータ名 12 = 位置*ParIndex*のテストパラメータ値
Attrib	属性値。
ParIndex	任意です。*AttribType* = 11または12のときの、テストパラメータのインデックスを指定します。最初のテストパラメータのインデックスは1です。

戻り値	説明
0	エラーなし。
<0	エラー。

注記

- ファイル名にパスが含まれていない場合、DAQおよびテストデータのデフォルトディレクトリを使用します。EA_Test.DataDirを参照してください。

例

```
Sub ListTestDescription()
  Dim Attrib
  Dim TestParCount As Integer, i As Integer
  Dim ParName As String, ParVal As String
  EA_Test.Load("C:\MYDATA\Test1.TST")
  EA_Test.TestAttribute(1, 10, Attrib)
  TestParCount=CInt(Attrib)
  For i=1 To TestParCount
    EA_Test.TestAttribute(1, 11, Attrib, i)
    ParName=CStr(Attrib)
    EA_Test.TestAttribute(1, 12, Attrib, i)
    ParVal=CStr(Attrib)
  Next
End Sub
```

EA_Test.TestCount(Read only)

解析プロジェクトに現在ロードされているテストの総数を返します。

```
Property EA_Test.TestCount
```

このプロパティは、プロジェクトのすべてのテストとすべてのチャネルを反復処理する場合に役立ちます。

```
Sub Attrib As Variant
Sub i As Integer
Sub NofChannels As Integer
Sub NofSamples As Integer
NofTests=EA_Test.TestCount

For i=1 To NofTests
  EA_Test.TestAttribute i, TEST_NOFCHANNELS, Attrib
  NofChannels = Attrib
    For j=1 To NofChannels
      EA_Test.ChannelAttribute i,j,CHAN_NAME,Attrib
      .....
    .....
Next
```

EA_Video

EA_Videoクラスは、ビデオカメラで使用するメソッドのセットを公開しています。スクリプト毎にキャプチャの開始/停止、解像度の設定、圧縮などを制御することができます。

⚠ これらスクリプトの主な目的は、ビデオキャプチャの開始/停止を制御することです。カメラの構成は通常の場合、対話形式で行われます。

⚠ ビデオ キャプチャ関数を使用するには、EasyVideocamまたはEasyRoadloadモジュール、またはcatmanAPのライセンスが必要です。

概要

上記の関数は、おそらく最も頻繁に使用する関数です。

EA_Video.SetCaptureFile
EA_Video.SetTempCaptureFile
EA_Video.SetCaptureMode
EA_Video.ReadyCapture
EA_Video.StartCapture
EA_Video.StopCapture
EA_Video.StartPreview
EA_Video.StopPreview
EA_Video.CaptureFrame
EA_Video.CaptureFrameTo

高度なコマンド

スクリプト経由でカメラとエンコーダを設定することを選択した場合は、次のコマンドのいずれかを使用します。

EA_Video.RefreshVideoDeviceList
EA_Video.GetVideoDeviceList
EA_Video.SelectVideoDevice
EA_Video.RefreshAudioDeviceList
EA_Video.GetAudioDeviceList
EA_Video.SelectAudioDevice
EA_Video.GetSupportedResolutions
EA_Video.SetResolution
EA_Video.GetVideoCompressionFormats
EA_Video.SetVideoCompressionFormat
EA_Video.GetAudioCompressionFormats
EA_Video.SetAudioCompressionFormat
EA_Video.SetFramesPerSecond
EA_Video.ShowCameraDialog
EA_Video.ShowVideoCompressionDialog

このオブジェクトでは以下のメソッドが公開されています:

EA_Video.CaptureFrame ・・・

EA_Video.CaptureFrame

"manual frames"(手動フレーム)モードで実行しているフレームをキャプチャします。ビデオストリームに単一のフレームを追加します(一連の静止画像をストリームとして想像することができます)。

```
EA_Video.CaptureFrame(ByVal CameraIndex As Integer) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- *EA_Video.CaptureFrame*を呼び出す前に、EA_Video.SetCaptureModeでキャプチャモードを"manual frames"に設定し、EA_Video.StartCaptureでキャプチャを開始する必要があります。

- "manual frames"モードはスクリプトでのみ設定することができます - catmanのユーザインタフェースからは利用できません。

EA_Video.CaptureFrameTo

キャプチャが"still image"(静止画)モードで実行している場合、指定した場所にフレームをJPEGファイルでキャプチャします。

```
  EA_Video.CaptureFrameTo(ByVal CameraIndex As Integer, ByVal ImageFile As String,
ByVal Quality As Long) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
ImageFile	画像を保存するフルパス(例えば"C:\MyFolder\Image.jpg")。
Quality	使用する圧縮係数の値。2から255の間で指定します。2は最高品質で、255は品質は落ちますが圧縮率は最高です。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- *EA_Video.CaptureFrameTo*を呼び出す前に、EA_Video.SetCaptureModeでキャプチャモードを"still image"に設定し、EA_Video.StartCaptureでキャプチャを開始する必要があります。

EA_Video.GetAudioCompressionFormats

利用可能なオーディオ圧縮形式をリストします。

```
  EA_Video.GetAudioCompressionFormats(ByVal CameraIndex As Long, ByRef Formats() As
String) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
Formats	利用可能なオーディオ圧縮形式の文字列配列。

戻り値	説明
0	エラーなし。
-1	エラー。

EA_Video.GetAudioDeviceList

利用可能なオーディオキャプチャデバイス名をリストします。

```
  EA_Video.GetAudioDeviceList(ByVal CameraIndex As Long, ByRef AudioDevices() As
String) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
AudioDevices	利用可能なオーディオキャプチャデバイス名の文字列配列。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- *AudioDevices*配列の最初のデバイス(指標0)は"None"です。カメラオブジェクトが、どのオーディオキャプチャデバイスとも結合されていない時に選択されるデバイスです。

EA_Video.GetSupportedResolutions

現在選択されているビデオキャプチャデバイスでサポートしている、すべての解像度をリストします。

```
  EA_Video.GetSupportedResolutions(ByVal CameraIndex As Long, ByRef Resolutions() As
String) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
Resolutions	現在選択されているビデオキャプチャデバイスでサポートしている、すべての解像度の文字列配列。

戻り値	説明
0	エラーなし。
-1	エラー。

EA_Video.GetVideoCompressionFormats

利用可能な圧縮形式をリストします。

```
  EA_Video.GetVideoCompressionFormats(ByVal CameraIndex As Long, ByRef Formats() As
String) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
Formats	利用可能なビデオ圧縮形式の文字列配列。

戻り値	説明
0	エラーなし。
-1	エラー。

EA_Video.GetVideoDeviceList

利用可能なビデオキャプチャデバイス名をリストします。

```
  EA_Video.GetVideoDeviceList(ByVal CameraIndex As Long, ByRef VideoDevices() As
String) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
VideoDevices	利用可能なビデオキャプチャデバイス名の文字列配列。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- *VideoDevices*配列の最初のデバイス(インデックス0)は"None"です。カメラオブジェクトが、どのビデオキャプチャデバイスとも結合していない時に選択されるデバイスです。

EA_Video.ReadyCapture

カメラがキャプチャを開始できるように準備します。初めてキャプチャを開始するとき、カメラは初期化プロセスを実行する必要があります。カメラやエンコーダの種類により、初期化プロセスに数秒かかる場合があります。この遅延を回避するには時間遅延が重要でないとき(例えばDAQジョブの開始前)に*ReadyCapture*メソッドを使用し、StartCapture(例えばイベントによるトリガ)ですぐに発生するようにします(< 100 msの遅延)。

```
  EA_Video.ReadyCapture(ByVal CameraIndex As Integer, Optional ByVal CaptureMode As
eCaptureMode = -1) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
CaptureMode	任意です。EA_Video.SetCaptureModeを使用せずにキャプチャモードを設定します。キャプチャモードを使用しない場合、カメラの現在のキャプチャモードで準備します。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- *ReadyCapture*の呼び出し後、任意のカメラ設定を変更(例えば解像度、キャプチャモード、圧縮など)すると初期化が必要となり、再度*ReadyCapture*を呼び出す必要があります。そのため*ReadyCapture*を呼び出す前に、すべての構成手順を完了しているようにしてください。

EA_Video.RefreshAudioDeviceList

接続したオーディオ デバイスをスキャンし、利用可能なビデオデバイス(オーディオ)のリストを更新します。スクリプト実行中に抜き差しした場合、EA_Video.GetAudioDeviceListを呼び出す前に、このメソッドを使用します。

```
EA_Video.RefreshAudioDeviceList(ByVal CameraIndex As Long) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。

戻り値	説明
0	エラーなし。
-1	エラー。

EA_Video.RefreshVideoDeviceList

接続したカメラをスキャンし、利用可能なビデオデバイス(カメラ)のリストを更新します。スクリプト実行中に抜き差しした場合、EA_Video.GetVideoDeviceListを呼び出す前に、このメソッドを使用します。

```
EA_Video.RefreshVideoDeviceList(ByVal CameraIndex As Long) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。

戻り値	説明
0	エラーなし。
-1	エラー。

EA_Video.SelectAudioDevice

オーディオデバイオスを選択します。利用可能なデバイスのリストを取得するにはEA_Video.GetAudioDeviceListを使用します。

```
  EA_Video.SelectAudioDevice(ByVal CameraIndex As Integer, ByVal DeviceIndex As Long)
As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
DeviceIndex	使用するオーディオキャプチャデバイスのインデックス。利用可能なビデオキャプチャデバイスのリストを取得するにはEA_Video.GetAudioDeviceListを使用します。

戻り値	説明
0	エラーなし。
-1	エラー。

EA_Video.SelectVideoDevice

ビデオデバイオスを選択します。利用可能なビデオキャプチャデバイスのリストを取得するにはEA_Video.GetVideoDeviceListを使用します。

```
  EA_Video.SelectVideoDevice(ByVal CameraIndex As Integer, ByVal DeviceIndex As Long)
As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
DeviceIndex	使用するビデオキャプチャデバイスのインデックス。利用可能なビデオキャプチャデバイスのリストを取得するにはEA_Video.GetVideoDeviceListを使用します。

戻り値	説明
0	エラーなし。
-1	エラー。

EA_Video.SetAudioCompressionFormat

利用可能なオーディオ圧縮形式をリストします。

```
  EA_Video.SetAudioCompressionFormat(ByVal CameraIndex As Long, ByVal FormatIndex As
Integer) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
FormatIndex	オーディオストリームキャプチャに使用するオーディオ圧縮形式のインデックス。利用可能なオーディオ圧縮形式のリストを取得するにはEA_Video.GetAudioCompressionFormatsを使用します。

戻り値	説明
0	エラーなし。
-1	エラー。

EA_Video.SetCaptureFile

最終的なキャプチャファイルのフルパスおよび名前を設定します。カメラがビデオキャプチャを開始するとcatmanは最初にデータを一時ファイルに取り込みます(EA_Video.SetTempCaptureFileを使用して一時ファイルのフルパスおよび名前を定義することができます)。EA_Video.StopCaptureを使用してキャプチャを停止させると、一時ファイルを確定し、キャプチャファイルに移動します。

```
EA_Video.SetCaptureFile(ByVal CameraIndex As Integer, ByVal VideoCaptureFile As
String, Optional ByVal Overwrite As Integer = 1) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
VideoCaptureFile	キャプチャ停止後、保存するメディアファイルのフルパス。
Overwrite	任意です。Falseに設定してターゲットファイルが既に存在している場合、ファイル名の末尾に番号を追加します(DAQジョブの計測ファイルと同じメカニズムを使用して元のファイルを上書きしないようにします)。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- キャプチャを開始したら、EA_Video.StopCaptureを使用してキャプチャを停止させる必要があります。

EA_Video.SetCaptureMode

キャプチャモードを"video or audio"(動画や音声)、"manual frames"(手動フレーム)および"still image"(静止画)のモード間で変更します。"video or audio"は標準的なビデオファイルのキャプチャ、"manual frames"はフレームを手動でキャプチャ(EA_Video.CaptureFrame)、"still image"モードは単一フレームをJPEGファイルにキャプチャ(EA_Video.CaptureFrameTo)します。

```
  EA_Video.SetCaptureMode(ByVal CameraIndex As Integer, ByVal CaptureMode As
eCaptureMode) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
CaptureMode	0 = Video or audio(動画や音声) 1 = Manual frames(手動フレーム) 2 = Still image(静止画) デフォルトは"video or audio"です。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- *SetCaptureMode*は、EA_Video.StartCaptureを呼び出す前に呼び出されている必要があります。
- "video or audio"モードがデフォルトのモードです。これは選択した形式(AVIがデフォルト)でビデオおよび/またはオーディオストリームをキャプチャします。
- "manual frames"モードでは、手動でビデオファイルのフレームを取得できます。EA_Video.CaptureFrameを使用してフレームをキャプチャします。
- "still image"モードではEA_Video.CaptureFrameToを呼び出すことにより、ビデオストリームのインスタント画像をJPEG ファイルとしてキャプチャすることができます。

EA_Video.SetFramesPerSecond

ビデオキャプチャに使用するフレームレートを設定します。

```
  EA_Video.SetFramesPerSecond(ByVal CameraIndex As Integer, ByVal FramesPerSecond As
Double) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
FramesPerSecond	フレームレート。ビデオキャプチャに使用する1秒あたりのフレーム数。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- フレーム レートが高いほど、同じビデオ長でもより多くの容量を占めます。また、より多くのCPUリソースを消費します。

EA_Video.SetPreTrigger

使用するプリトリガのサイズを設定します。

```
  EA_Video.SetPreTrigger(ByVal CameraIndex As Integer, ByVal PreTrigger As Long) As
Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
PreTrigger	プリトリガサイズ(ms)。プリトリガを無効にする場合、0に設定します。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- この関数は、EA_Video.StartCaptureやEA_Video.ReadyCaptureの前に呼び出す必要があります。
- フレームはRAMにバッファされるため、キャプチャ開始時はRAMに十分な空き容量があることを確認します。
- この関数はAVT社のカメラのみをサポートしています。

EA_Video.SetResolution

ビデオキャプチャデバイスで使用する解像度を設定します。

```
  EA_Video.SetResolution(ByVal CameraIndex As Long, ByVal ResolutionIndex As Integer)
As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
ResolutionIndex	ビデオキャプチャデバイスで使用する解像度のインデックス。利用可能な解像度の一覧を取得するにはEA_Video.GetSupportedResolutionsを使用します。

戻り値	説明
0	エラーなし。
-1	エラー。

EA_Video.SetTempCaptureFile

一時ビデオファイルのフルパスおよび名前を設定します。カメラがビデオキャプチャを開始すると、catmanは最初にこの一時ファイルに書き込みます。キャプチャが終了すると、一時ファイル名は最終的なパスに移動/変更されます(EA_Video.SetCaptureFileを使用して最終的なファイルのフルパスおよび名前を定義することができます)。

```
  EA_Video.SetTempCaptureFile(ByVal CameraIndex As Integer, ByVal TempCaptureFile As
String) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
TempCaptureFile	一時ビデオファイルを保存するフルパス。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- キャプチャを開始したら、EA_Video.StopCaptureを使用してキャプチャを停止させる必要があります。

EA_Video.SetVideoCompressionFormat

ビデオキャプチャで使用する圧縮形式を設定します。

```
  EA_Video.SetVideoCompressionFormat(ByVal CameraIndex As Long, ByVal FormatIndex As
Integer) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
FormatIndex	ビデオキャプチャで使用するインデックスの指標。利用可能な圧縮形式のリストを取得するには EA_Video.GetVideoCompressionFormatsを使用します。

戻り値	説明
0	エラーなし。
-1	エラー。

EA_Video.ShowCameraDialog

カメラメーカから提供する、カメラドライバの組み込みダイアログを表示します。

HBMはこのダイアログの機能についての責任を負いません! 一部のカメラでは構成が実現できない場合があります。

```
EA_Video.ShowCameraDialog(ByVal CameraIndex As Integer) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。

戻り値	説明
0	エラーなし。
-1	エラー。

EA_Video.ShowVideoCompressionDialog

選択したカメラのエンコーダのサプライヤから供給された、エンコーダの組み込みダイアログを表示します(Windows Media Encoder 9はMicrosoftが推奨しています)。

HBMはこのダイアログの機能についての責任を負いません! 一部のエンコーダでは構成が実現できない場合があります。

```
EA_Video.ShowVideoCompressionDialog(ByVal CameraIndex As Integer) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。

戻り値	説明
0	エラーなし。
-1	エラー。

EA_Video.StartCapture

キャプチャを開始します。めてキャプチャを開始するとき、カメラは初期化プロセスを実行する必要があり、これには最大で数秒かかる場合があります。この遅延を回避するためにEA_Video.ReadyCaptureを使用し、カメラを事前に準備しておきます。

```
EA_Video.StartCapture(ByVal CameraIndex As Integer) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- キャプチャを開始したら、EA_Video.StopCaptureを使用してキャプチャを停止させる必要があります。
- *StartCapture*の呼び出しから、効果的なキャプチャ開始までの必要な時間を短縮するため、キャプチャ開始前にEA_Video.ReadyCaptureを呼び出します。

EA_Video.StartPreview

カメラのプレビュー(すなわちライブ画像の表示)を開始します。DAQジョブの実行中にカメラのライブ画像を見ることができます(catmanの可視化システムの"Video display"オブジェクトは、Floatingウィンドウでもライブ画像を表示することができます)。

```
EA_Video.StartPreview(ByVal CameraIndex As Integer) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- プレビューを停止するにはEA_Video.StopPreviewを使用します。
- カメラのライブ画像の表示は高負荷なため、低い性能のPC(CX22 Data Recorder moduleなど)ではプレビューを無効にすることを推奨しています。

EA_Video.StopCapture

キャプチャを停止します。"video or audio"(動画や音声)および"manual frames"(手動フレーム)モードでキャプチャを停止すると、一時キャプチャファイルは最終的なキャプチャファイルの場所に移動します。最終的なファイル名の指定はEA_Video.SetCaptureFileを使用します。

```
EA_Video.StopCapture(ByVal CameraIndex As Integer) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。

戻り値	説明
0	エラーなし。
-1	エラー。

EA_Video.StopPreview

カメラのプレビュー(すなわちライブ画像の表示)を停止します。

```
EA_Video.StopPreview(ByVal CameraIndex As Integer) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- カメラのライブ画像の表示は高負荷なため、低い性能のPC(CX22 Data Recorder moduleなど)ではプレビューを無効にすることを推奨しています。

EA_Video.TriggerEvent

プリトリガによるバッファリングを終了し、キャプチャを開始するトリガイベントを送信します。

```
EA_Video.TriggerEvent(ByVal CameraIndex As Integer, ByVal TEvent As Byte) As Long
```

引数	説明
CameraIndex	使用するカメラオブジェクトのインデックス。これは0から7までです。
TEvent	将来の実装のために予約しています。この引数は0に設定してください。

戻り値	説明
0	エラーなし。
-1	エラー。

注記

- この関数は、EA_Video.SetPreTriggerおよびEA_Video.StartCaptureによるビデオキャプチャを開始した後に呼び出す必要があります。

EA_Web

*EA_Web*オブジェクトは、HTMLを使用して独自のWebページを作成するために使用します。

このオブジェクトは以下が行えます:

- Webページの取得または設定: EA_Web.GetPage、EA_Web.SetPage
- HTML要素のコンテンツを編集: EA_Web.InsertText、EA_Web.ReplaceTag
- HTMLのページファイルを配信: EA_Web.SetResponseFile、EA_Web.SetResponseType
- キーと値のペアでリクエスト項目を取得: EA_Web.GetRequestItem

このオブジェクトでは以下のメソッドが公開されています:

EA_Web.GetPage ・・・

EA_Web.GetPage

HTMLでWebページの内容を取得します。

```
EA_Web.GetPage(strPage As String) As Long
```

引数	説明
strPage	HTML形式のWebページの内容。

戻り値	説明
0	エラーなし。
<0	エラー。Webサーバが実行されていないなど。

EA_Web.GetRequestItem

現在のリクエストの指定したキーの値を取得します。

```
EA_Web.GetRequestItem(ByVal Key As String, Value As Variant) As Long
```

引数	説明
Key	キー名。
Value	指定したキーの値。

戻り値	説明
0	エラーなし。
<0	エラー。Webサーバが実行されていないなど。

EA_Web.InsertText

指定したタグ要素にテキストを追加します。

```
EA_Web.InsertText(ByVal strTag As String, ByVal strIns As String) As Long
```

引数	説明
strTag	要素のタグ名。
strIns	指定したタグ要素に追加するテキスト。

戻り値	説明
0	エラーなし。
<0	エラー。Webサーバが実行されていないなど。

EA_Web.ReplaceTag

HTMLのタグ名を置き換えます。

```
EA_Web.ReplaceTag(ByVal strTag As String, ByVal strRep As String) As Long
```

引数	説明
strTag	置き換えるタグ。
strRep	置き換える文字列。

戻り値	説明
0	エラーなし。
<0	エラー。Webサーバが実行されていないなど。

EA_Web.SetPage

HTMLでWebページの内容を設定します。

```
EA_Web.SetPage(strPage As String) As Long
```

引数	説明
strPage	HTML形式のWebページの内容。

戻り値	説明
0	エラーなし。
<0	エラー。Webサーバが実行されていないなど。

EA_Web.SetResponseFile

HTMLのファイル名を設定します。

```
EA_Web.SetResponseFile(ByVal FileName As String) As Long
```

引数	説明
FileName	HTMLのファイル名。

戻り値	説明
0	エラーなし。
<0	エラー。Webサーバが実行されていないなど。

EA_Web.SetResponseType

リクエストのContent-Typeを設定します。

```
EA_Web.SetResponseType(ByVal ResponseType As Integer) As Long
```

引数	説明
ResponseType	0 = RESPONSETYPE_AUTO Webサーバはファイルの拡張子からContent-Typeを決定します。 1 = RESPONSETYPE_DOWNLOAD ブラウザを開かずにファイルをダウンロードします。

戻り値	説明
0	エラーなし。
<0	エラー。Webサーバが実行されていないなど。

セルのデータソース表記

セルをチャネルから取得した情報(最大値など)にリンクします。これは通常、チャネルをセルにドラッグすることによって行われますが、スクリプトを使用してセルのデータソースを指定することもできます。ここでは、チャネルとさまざまな種類の情報を指定するために使用される表記法について説明します。

EA_Panel.SetCellメソッドを使用して、セルをデータソースにリンクします。

1.DAQモードのセルデータソース

データソース説明文字列の一般的な構造は、"|"で区切られた項目のリストです:

```
"SourceIndex|SourceType|"
```

アイテム	説明
SourceIndex	DAQチャネルリストのチャネルインデックス。
SourceType	CELLSOURCE_VALUE = 995 CELLSOURCE_NAME = 996 CELLSOURCE_UNIT = 997 CELLSOURCE_VALUEANDUNIT = 998 CELLSOURCE_MINVALUE = 2701 CELLSOURCE_MAXVALUE = 2702 CELLSOURCE_MINVALUEANDUNIT = 2703 CELLSOURCE_MAXVALUEANDUNIT = 2704 CELLSOURCE_LIMITVALUE1 = 2717 CELLSOURCE_LIMITVALUE2 = 2718 CELLSOURCE_LIMITVALUE3 = 2719

例

次の例では、複数の隣接するセルをチャネル名、値、およびチャネルの単位"Pressure"にリンクします:

```
Dim pChan As Integer
Dim pDev As Integer
pChan = EA_IO.ChanPos("Pressure")
pDev = EA_IO.DevicePos("HBM_MX840_No1")
pChan = pChan + pDev
EA_Panel.SetCell 1, "DAQ_TABLE", 1, 1, pChan & "|996|", TYPE_DATASOURCE
EA_Panel.SetCell 1, "DAQ_TABLE", 2, 1, pChan & "|995|", TYPE_DATASOURCE
EA_Panel.SetCell 1, "DAQ_TABLE", 3, 1, pChan & "|997|", TYPE_DATASOURCE
```

2. 解析モードのセルデータソース

データソース説明文字列の一般的な構造は、"|"で区切られた項目のリストです:

```
"SourceType|SourceName|Test index|File index|Number format"
```

アイテム	説明
SourceType	CELLSOURCE_TESTPAR = 1 CELLSOURCE_CHANMIN = 2 CELLSOURCE_CHANMAX = 3 CELLSOURCE_CHANMEAN = 4 CELLSOURCE_CHANSTD = 5 CELLSOURCE_CHANNEL = 6 CELLSOURCE_CHANNAME = 7 CELLSOURCE_CHANCOMMENT = 8 CELLSOURCE_CHANUNIT = 9 CELLSOURCE_CHANSENSOR = 10 CELLSOURCE_CHANSENSORTID = 11 CELLSOURCE_CHANAMPTYPE = 12 CELLSOURCE_CHANAPTYPE = 13 CELLSOURCE_TRANSDUCER = 14 CELLSOURCE_EXCITATION = 15 CELLSOURCE_FILTER = 16 CELLSOURCE_GAGEFAC = 17 CELLSOURCE_ZEROVAL = 18 CELLSOURCE_ALLTRACEDATAROW = 19 CELLSOURCE_ALLTRACEDATACOL = 20 CELLSOURCE_SAMPLERATE = 21 CELLSOURCE_SAMPLECOUNT = 22 CELLSOURCE_SCALING = 23 CELLSOURCE_ELECTRICALUNIT = 24 CELLSOURCE_BRIDGEFACTOR = 25 CELLSOURCE_DROPTYPE_MIN = 26 CELLSOURCE_DROPTYPE_MAX = 27 CELLSOURCE_DROPTYPE_MEAN = 28 CELLSOURCE_DROPTYPE_STD = 29 CELLSOURCE_CARRIERFREQ = 30 CELLSOURCE_CHANSENSOR_SN = 31 CELLSOURCE_TESTFILE = 100 CELLSOURCE_DATAFILE = 101 CELLSOURCE_DATAFILECOMMENT = 102 CELLSOURCE_PPMATH = 200
SourceName	情報の生成元となるチャネルまたはテストパラメータの名前(SourceType = CELLSOURCE_TESTPARの場合)。SourceTypeがチャネルまたはテストパラメータ名を必要としない場合、これは空にすることができます:例えば"100||1|1"(ただし省略できません)。
Test index	解析プロジェクトにおけるテストの位置。最初のテストはインデックス1です。
File index	予約済み。常に1です。
Number format	数値精度(例: 0.000)。省略可能です。

例

次の例では、複数の隣接するセルを最小値、最大値、およびチャネルの単位"Pressure"にリンクします:

```
EA_Panel.SetCell 0, "REPORT_TABLE", 1, 1, "2|Pressure|1|1|0.00", TYPE_DATASOURCE
EA_Panel.SetCell 0, "REPORT_TABLE", 2, 1, "3|Pressure|1|1|0.00", TYPE_DATASOURCE
EA_Panel.SetCell 0, "REPORT_TABLE", 3, 1, "9|Pressure|1|1|", TYPE_DATASOURCE
```

EasyScriptのエラーコード

以下のエラー状態がEasyScriptメソッドより返されます。

VBA Script Engineにより報告されるVBAコード自体のエラー("ゼロ除算"など)は、VBAエラーリストを参照してください。

引数	説明
-1	現在のcatmanEasy/APの動作モードで行えない関数(EA.OperationModeを参照)。 考えられる理由: • catmanEasy/APがDAQモードの時にEA_Testメソッドを呼び出した。 • DAQジョブを実行していないにもかかわらずEA_DAQ.CurrentReadingを呼び出した。
-2	デバイスのスキャンで様々な種類のデバイスを検出しました。しかし、これはcatmanEasy/APでサポートしていません。
-3	テストは既に読み込み済みです。
-4	ファイルが見つかりません。
-5	ファイルはこの操作で使用できません。
-6	インデックスが許容範囲外です。
-7	チャネルは存在しません。
-8	テストが読み込まれていません。
-9	ファイルを作成できませんでした。これは演算の結果ファイルの可能性があります(すなわち、ディスクの空き容量が十分でない)。
-10	テストファイルを開けませんでした。
-11	チャネル名は既に存在しています。チャネル名は一意である必要があります。
-12	パネルが見つかりません。
-13	オブジェクトが見つかりません。
-14	DAQジョブがみつかりません。
-15	無効なDAQジョブのプロパティ。
-16	計測データの保存に失敗しました(EA_Job.StoreData)。更なる情報はEA.LastErrorおよびEA.LastErrorTextより取得できます。
-17	DAQジョブは実行していません。
-18	センサが見つかりません。
-19	チャネルを初期化できませんでした。更なる情報はEA.LastErrorおよびEA.LastErrorTextより取得できます。
-20	無効なデバイススキャン。 考えられる理由: • デバイスが見つからない。 • 様々な種類のデバイスを検出を検出しました。しかし、これはcatmanEasy/APでサポートしていません。
-21	オブジェクトまたはチャネルは、この操作で有効でありません。
-22	チャネル名は既に存在しています。
-23	関数の引数は無効です。
-24	関数に引数がありません。
-25	プロット(曲線)が見つかりません。
-26	演算が見つかりません。
-27	無効なパスやパス名です。
-28	プロシージャまたはメソッドがみつかりません。
-29	操作は、このデバイスの種類では有効ではありません。
-30	プロセス変数が見つかりません。
-31	ラベル/ジャンプが見つかりません。
-32	MS Officeの関数の実行エラーです。
-33	数式内の引数が不正です。
-34	数式に括弧がありません。
-35	不正な数式です。
-36	ゼロで除算しました。
-37	数式に引数がありません。
-38	一般的なデータベースアクセスエラー(DAO)です。
-39	データベースが開かれていません。
-40	レコードセットが開かれていません。

-41	チャネルが十分ではありません。catmanのセッションでサポートしている一時保管よりも、多くのチャネルを含むプロジェクトを読み込もうとした場合に発生することがあります。一時保管のサイズを変更します(OPTIONS/DATA STORAGE)。
-42	グラフの最大プロット数を超えています。
-43	不正なオブジェクトです。オブジェクトはこの種類の操作に有効ではありません(例えば等高線グラフではないグラフに対して*EA_Test.PlotMatrix*を呼び出した)。
-44	データがありません。演算は空きの引数のチャネルを参照しました。
-45	長さが一致していません。この操作は同じ長さ(同じサンプル数)でなければならない、複数のチャネルが含まれています。
-46	データが多すぎます(例えば*PlotArrayXY*で最大点数を超えた場合)。
-47	ディレクトリを作成できません。
-48	古いファームウェアです。
-49	NTPの開始に失敗しました。
-50	NTPサーバが見つかりません。
-51	ハードウェアコマンドが失敗しました。
-52	ディスクの空き容量が足りません。
-53	ファイルを開けません(例えば書き込みモードでファイルを開く権限がないなど)。
-54	プロジェクトの再読み込みができません。
-55	Sensor groupが見つかりません。
-56	サポートしてないエクスポート形式です。
-57	不正なチャネル名です。
-58	パネルの読み込みに失敗 - 可視化オブジェクトが最大数を超えました。
-59	Webサーバが見つかりません。
-60	スクリプトコマンドはこの実行ステージで使用できません(例えば*EA_Job.Terminate*は、実行ステージ*AFTERHARDWARETRANSFER*、*AFTERREADBLOCK*および*AFTERDAQSTORE*で許可されません)。
-61	参照したチャネルのデバイスは無効です。
-62	チャネルの内容がその操作に対して無効です(例: CANの生データまたは演算の引数チャネルとしてのクラスカウントデータチャネル)。
-33	ライセンスがありません。
-64	信号が見つかりません。
-65	FTPにアップロードするコンポーネントが見つかりません。
-66	FTPへのアップロード失敗しました。
-67	デバイスはサポートされていません。
-32666	catman Core Servicesのエラーです。更なる情報はEA.LastErrorおよびEA.LastErrorTextより取得できます。
-32767	予期しないエラーです。更なる情報はEA.LastErrorおよびEA.LastErrorTextより取得できます。

VBAリファレンス

VBAのコアとなる言語定義を提供します。以下のリストに、すべてのトピックがグループに分類されています。

- 割り当て
  Erase, Let, LSet, RSet, Set

- 定数
  Empty, False, Nothing, Null, True, Win16, Win32

- 変換
  Array, CBool, CByte, CCur, CDate, CDec, CDbl, CInt, CLng, CSng, CStr, CVar, CVDate, CVErr, Val

- データ型
  Any, Boolean, Byte, Currency, Date, Decimal, Double, Integer, Long, Object, PortInt, Single, String, String*n, Variant, obj type, user enum, user type

- 宣言
  #Reference, #Uses, Class Module, Code Module, Const, Declare, Deftype, Dim, Enum...End Enum, Function...End Function, Option, Private, Property...End Property, Public, ReDim, Static, Sub...End Sub, Type...End Type

- ダイアログ関数
  Dialog Func, DlgControlId, DlgCount, DlgEnable, DlgEnd, DlgFocus, DlgListBoxArray, DlgName, DlgNumber, DlgSetPicture, DlgText, DlgType, DlgValue, DlgVisible

- エラーハンドリング
  Err, Error, On Error, Resume

- ファイル
  ChDir, ChDrive, Close, CurDir, Dir, EOF, FileAttr, FileCopy, FileDateTime, FileLen, FreeFile, Get, GetAttr, Input, Input$, Kill, Line Input, Loc, Lock, LOF, MkDir, Name, Open, Print, Put, Reset, RmDir, Seek, Seek, SetAttr, Unlock, Write

- フロー制御
  Call, CallByName, Do...Loop, End, Exit, For...Next, For Each...Next, GoTo, If...ElseIf...Else...End If, Select Case...End Select, Stop, While...Wend

- 数学
  Abs, Atn, Cos, Exp, Fix, Int, Log, Randomize, Rnd, Round, Sgn, Sin, Sqr, Tan

- その他
  AppActivate, Beep, CallersLine, Choose, Clipboard, Debug.Print, DoEvents, Environ, Eval, IIf, KeyName, QBColor, Rem, RGB, SendKeys, Shell, Wait

- オブジェクト
  CreateObject, GetObject, Me, With...End With

- 演算子
  Operators: +, -, ^, *, /, \, Mod, +, -, &, =, <>, <, >, <=, >=, Like. Not, And, Or, Xor, Eqv, Imp, Is.

- 設定
  DeleteSetting, GetAllSettings, GetSetting, SaveSetting

- 文字列
  Asc, AscB, AscW, Chr, ChrB, ChrW, Format, Hex, InStr, InStrB, InStrRev, Join, LCase, Left, LeftB, Len, LenB, LTrim, Mid, MidB, Oct, Replace, Right, RightB, RTrim, Space, Split, String, Str, StrComp, StrConv, StrReverse, Trim, UCase

- 時間/日付
  Date, DateAdd, DateDiff, DatePart, DateSerial, DateValue, Day, Hour, Minute, Month, MonthName, Now, Second, Time, Timer, TimeSerial, TimeValue, Weekday, WeekdayName, Year

- ユーザダイアログ
  Begin Dialog...End Dialog, CancelButton, CheckBox, ComboBox, DropListBox, GroupBox, ListBox, MultiListBox, OKButton, OptionButton, OptionGroup, Picture, PushButton, Text, TextBox

- ユーザ入力
  Dialog, GetFilePath, InputBox, MsgBox, ShowPopupMenu

- 変数情報
  IsArray, IsDate, IsEmpty, IsError, IsMissing, IsNull, IsNumeric, IsObject, LBound, TypeName, UBound, VarType

VBAリファレンス ・・・

Classモジュール

Class ModuleはActiveX Automationを実装します。

- Publicプロシージャのセットは、他のモジュールからアクセス可能です。
- これらのpublicシンボルは、オブジェクト変数よりアクセスされます。
- Public Consts、Type、配列、固定長の文字列は使用できません。
- Class Moduleはインスタンスが自動的に作成されない点を除いて、オブジェクトモジュールと似ています。
- インスタンスを作成するには:

```
Dim Obj As classname
Set Obj = New classname
```

プロジェクトおよびモジュール、Code Module、Uses Commentも参照してください、

例

```
'A.BAS
'#Uses "File.CLS"
Sub Main
Dim File As New File
File.Attach "C:\AUTOEXEC.BAT"
Debug.Print File.ReadLine
End Sub

'File.CLS
Option Explicit
Dim FN As Integer
Public Sub Attach(FileName As String)
FN = FreeFile
Open FileName For Input As #FN
End Sub
Public Sub Detach()
If FN <> 0 Then Close #FN
FN = 0
End Sub
Public Function ReadLine() As String
Line Input #FN,ReadLine
End Function
Private Sub Class_Initialize()
Debug.Print "Class_Initialize"
End Sub
Private Sub Class_Terminate()
Debug.Print "Class_Terminate"
Detach
End Sub
```

Class_Initialize Sub

Class moduleの初期化サブルーチン。Class module用の新規インスタンスが作成される毎に、Class_Initializeサブクラスが呼び出されます。Class_Initializeが定義されていない場合、特別な初期化は行いません。

```
Private Sub Class_Initialize()
...
End Sub
```

Code Module、Class_Terminateも参照してください。

Class_Terminate Sub

Class moduleの終了サブルーチン。

```
Private Sub Class_Terminate()
...
End Sub
```

Class moduleのインスタンスが破棄される毎に、Class_Terminateサブクラスが呼び出されます。Class_Terminateが定義されていない場合、特別な終了は発生しません。

Code Module、Class_Initializeも参照してください。

Codeモジュール

Code moduleはコードライブラリを実装します。

- Publicプロシージャのセットは、他のモジュールからアクセス可能です。
- これらのpublicシンボルは、直接アクセスされます。

プロジェクトおよびモジュール、Class Module、Uses Commentも参照してください、

例

```
'A.BAS
'#Uses "Module1.BAS"
Sub Main
Debug.Print Value '"Hello"
End Sub

'Module1.BAS
Option Explicit
Private mValue As String
Property Get Value() As String
Value = mValue
End Property
'this sub is called when the module is first loaded
Private Sub Main
mValue = "Hello"
End Sub
```

Main Sub

Form 1: 各マクロはSub Mainを定義する必要があります。マクロとは"プログラム"です。マクロを実行するとSub Mainが起動し、サブルーチンが終了するまで実行を続けます。

Form 2: Code ModuleはPrivate Sub Mainを定義することができます。このSub Mainは、Code Moduleの初期化サブルーチンです。Mainが定義されていない場合、特別な初期化は行われません。

```
Sub Main()
...
End Sub
```

-or-

```
Private Sub Main()
...
End Sub
```

Code Moduleも参照してください。

データ型

Any Data Type ・・・

Any Data Type

任意の変数式(宣言のみ)。

Boolean Data Type

TrueまたはFalse値。

Byte Data Type

8bitの符号なし整数。

Currency Data Type

64bitの固定小数点実数(2の補数のバイナリ値を10,000でスケーリング)。

Date Data Type

64bitの実数。整数部は日付を表し、小数部は時刻を表します。(1899年12月30日= 0)。#date#を式内のリテラル日付値として使用します。

Decimal Data Type

Win32のみ。
96bitでスケーリングした実数。Decimalは有効な変数型ではありませんが、Variant変数にDecimal値を含めることができます(CDecを参照)。
10進数は次の形式です:

```
s * m * 10 ^ -p
```

s - 符号(+1または-1)
m - 仮数、96ビットの符号なし2進値(0から79,228,162,514,264,337,593,543,950,335)
p - スケーリングパワー(0～+28)

Double Data Type

64bitの実数。

Integer Data Type

16bitの整数。

Long Data Type

32bitの整数。

Object Data Type

オブジェクト参照値。(Objectsを参照)。

PortInt Data Type

移植可能な整数。

- Win16: 16bitの整数
- Win32: 32bitの整数

Single Data Type

32bitの実数。

String Data Type

任意の長さの文字列。いくつかの便利な文字列定数はあらかじめ定義されています:

- `vbNullChar` `Chr(0)`と同じです。
- `vbCrLf` `Chr(13)&Chr(10)`と同じです。
- `vbCr` `Chr(13)`と同じです。
- `vbLf` `Chr(10)`と同じです。
- `vbBack` `Chr(8)`と同じです。
- `vbFormFeed` `Chr(12)`と同じです。
- `vbTab` `Chr(9)`と同じです。
- `vbVerticalTab` `Chr(11)`と同じです。

String*n Data Type

固定長(n)文字列。

Variant Data Type

空の数値、通貨、日付、文字列、オブジェクト、エラーコード、NULLまたは配列。

定数およびキーワード

Empty Keyword ・・・

Empty Keyword

任意の値を持たないvariantvar。

False Keyword

値がゼロのときcondexprはFalseです。Falseを返す関数は、ゼロを返します。

Friend Keyword

モジュール内のFriend関数、プロパティ、およびSubは、モジュールにアクセスする他のすべてのモジュールで使用できます。FriendはObject変数からアクセスできません。

Nothing Keyword

任意のオブジェクトを参照しないobjexpr。

Null Keyword

nullのバリアント式。null値は伝播して、式全体がNullとなります。文字列や数値の引数にNull値を使用すると、実行時にエラーとなります。Null値は "#NULL#"として出力されます。

例

```
Sub Main
  X = Null
  Debug.Print X = Null '#NULL#
  Debug.Print IsNull(X) 'True
End Sub
```

Private Keyword

Private Const、Declare、Function、Property、SubおよびTypeは現在のモジュールでのみ使用できます。

Public Keyword

モジュール内のPublic Const、Declare、Function、Property、SubおよびTypeはアクセスする他のすべてのモジュールで使用できます。

True Keyword

値がゼロ以外の場合、条件式はTrueです。Trueを返す関数は-1を返します。

Win16 Keyword

16bitsで動作している場合、True。32bitsで動作している場合、False。

Win32 Keyword

32bitsで動作している場合、True。16bitsで動作している場合、False。

演算子

これらの演算子は、数値n1とn2、または文字列s1とs2で使用できます。式にNullが含まれる場合、式の値はNullになります。
演算子の評価順序は、演算子の優先順位によって制御され
ます。

```
^ Not * / \ Mod + - & < <= > >= = <> Is And Or Xor Eqv Imp
```

演算子	説明
-n1	n1を反転。
n1 ^ n2	n1をn2の累乗にします。
n1 * n2	n1とn2を乗算します。
n1 / n2	n1の整数をn2の整数で除算します。
n1 Mod n2	n1の整数をn2の整数で除算した余り。
n1 + n2	n1とn2を加算します。
s1 + s2	s1とs2を連結します。
n1 - n2	n1とn2を減算します。
s1 & s2	s1とs2を連結します。
n1 < n2	n1がn2より小さい場合、Trueを返します。
n1 <= n2	n1がn2以下の場合、Trueを返します。
n1 > n2	n1がn2より大きい場合、Trueを返します。
n1 >= n2	n1がn2以上の場合、Trueを返します。
n1 = n2	n1とn2が等しい場合、Trueを返します。
n1 <> n2	n1とn2が等しくない場合、Trueを返します。
s1 < s2	s1がs2より小さい場合、Trueを返します。
s1 <= s2	s1がs2以下の場合、Trueを返します。
s1 > s2	s1がs2より大きい場合、Trueを返します。
s1 >= s2	s1がs2以上の場合、Trueを返します。
s1 = s2	s1とs2が等しい場合、Trueを返します。
s1 <> s2	s1とs2が等しくない場合、Trueを返します。
Not n1	Bit単位でn1の整数値の論理否定。Only Not TrueはFalseです。
n1 And n2	整数値n1と整数値n2のbit単位の論理積。
n1 Or n2	整数値n1と整数値n2のbit単位の論理和。
n1 xOr n2	整数値n1と整数値n2のbit単位の排他論理和。
n1 Eqv n2	整数値n1と整数値n2のbit単位の論理等価。
n1 Imp n2	整数値n1と整数値n2のbit単位の論理包含。

例

```
Sub Main
N1 = 10
N2 = 3
S1$ = "asdfg"
S2$ = "hjkl"
Debug.Print -N1 '-10
Debug.Print N1 ^ N2 ' 1000
Debug.Print Not N1 '-11
Debug.Print N1 * N2 ' 30
Debug.Print N1 / N2 ' 3.3333333333333
Debug.Print N1 \ N2 ' 3
Debug.Print N1 Mod N2 ' 1
Debug.Print N1 + N2 ' 13
Debug.Print S1$ + S2$ '"asdfghjkl"
Debug.Print N1 - N2 ' 7
Debug.Print N1 & N2 '"103"
Debug.Print N1 < N2 'False
Debug.Print N1 <= N2 'False
Debug.Print N1 > N2 'True
Debug.Print N1 >= N2 'True
Debug.Print N1 = N2 'False
Debug.Print N1 <> N2 'True
Debug.Print S1$ < S2$ 'True
Debug.Print S1$ <= S2$ 'True
Debug.Print S1$ > S2$ 'False
Debug.Print S1$ >= S2$ 'False
Debug.Print S1$ = S2$ 'False
Debug.Print S1$ <> S2$ 'True
Debug.Print N1 And N2 ' 2
Debug.Print N1 Or N2 ' 11
Debug.Print N1 Xor N2 ' 9
Debug.Print N1 Eqv N2 ' -10
Debug.Print N1 Imp N2 ' -9
End Sub
```

コマンド、定義および制御フロー

As type definition ・・・

As type definition

変数型とパラメータ型だけでなく、関数とプロパティの結果はAs型を使って指定することができます: Boolean、Byte、Currency、Date、Double、Integer、Long、Object、PortInt、Single、String、String*n、UserDialog、Variant、objtype、userenum、usertypeの何れかを指定します。

As [New] type definition

Dim、Private、PublicおよびStaticステートメントは、As typeまたは`As New objtype`を使用して変数型を宣言できます。`As New objtype`を使用して宣言された変数は、変数がNothingの場合、使用前に自動的に作成されます。

Const Definition

exprの値としてnameを定義します。exprは他の定数や組込み関数を参照することができます。定数の型が指定されていない場合は、exprの型が使用されます。Subの外で定義された定数、FunctionやPropertyブロックは、モジュール全体で使用できます。PrivateまたはPublicのいずれも指定されていない場合、Privateが想定されます。

SubのConstステートメント、FunctionやPropertyブロックはPrivateやPublicを使用できません。

```
[ | Private | Public ] Const name[type] [As Type] = expr[, ...]
```

例

```
Sub Main
Const Pi = 4*Atn(1), e = Exp(1)
Debug.Print Pi ' 3.14159265358979
Debug.Print e ' 2.71828182845905
End Sub
```

Declare Definition

DLLで定義されたサブルーチンまたは関数へのインタフェース。呼び出すarglistの値はparamsに割り当てられます。
PrivateもPublicも指定されていない場合、Declareはデフォルトで Publicになります。

警告!
DLLのサブルーチンや関数を宣言するときは特に注意してください。間違ってパラメータや結果を宣言すると、Windowsが停止する可能性があります。新規のDLLを宣言する前に、開いてるドキュメントは全て保存します。
Err.LastDLLErrorは、最後のDLL呼び出しのエラーコードを返します(Windows 32 bit版のみ)。

```
  [ | Private | Public ] Declare Sub name Lib "dll name" [Alias "module name"]
[([param[, ...]])]
```

または

```
  [ | Private | Public ] Declare Function name[type] Lib "dll name" [Alias "module
name"] [([param[, ...]])] [As type[()]]
```

変数	説明
name	定義されているサブルーチンまたは関数の名前。Alias "module name"を省略した場合、このモジュール名になります。
"dll name"	モジュールのコードがあるDLLファイルです。
"module name"	DLLファイル内のモジュールの名。#numberの場合、それはモジュールの序数です。省略された場合、nameはモジュール名です。 DLLは、指定されたモジュール名を検索します。モジュールが存在する場合、それを使用します。すべてのAs Stringパラメータは、DLLを呼び出す前にUnicodeからASCIIに変換し、その後ASCIIからUnicodeに変換します(ASCIIからUnicodeへの変換を防ぐには、"Unicode:module name"を使用します)。 モジュールが存在しない場合、以下の二つでモジュール名が試します。 1. Windows NTのみ: "W" が付加されたモジュール名を試します。すべてのAs Stringパラメータは、DLLを呼び出すためにUnicodeとして渡されます。 2. Windows NTおよびWindows 95: "A" が付加されたモジュール名を試します。すべてのAs Stringパラメータは、DLLを呼び出す前にUnicodeからASCIIに変換し、その後ASCIIからUnicodeに変換します。 モジュール名が見つからない場合、実行時エラーが発生します。
params	DLLサブルーチンや関数で使用する0個以上のparamリスト。

Function、Sub、Callも参照してください。

例

```
Declare Function GetActiveWindow& Lib "user32" ()
Declare Function GetWindowTextLengthA& Lib "user32" _
(ByVal hwnd&)
Declare Sub GetWindowTextA Lib "user32" _
(ByVal hwnd&, ByVal lpsz$, ByVal cbMax&)
Function ActiveWindowTitle$()
ActiveWindow = GetActiveWindow()
TitleLen = GetWindowTextLengthA(ActiveWindow)
Title$ = Space$(TitleLen)
GetWindowTextA ActiveWindow,Title$,TitleLen+1
ActiveWindowTitle$ = Title$
End Function
Sub Main
Debug.Print ActiveWindowTitle$()
End Sub
```

Def Definition

型なし変数は次のように定義します:

- DefBool - Boolean
- DefByte - Byte
- DefCur - Currency
- DefDate - Date
- DefDbl - Double
- DefInt - Integer
- DefLng - Long
- DefObj - Object
- DefSng - Single
- DefStr - String
- DefVar - Variant

```
Def{Bool|Cur|Date|Dbl|Int|Lng|Obj|Sng|Str|Var} _
letterrange[, ...]
```

変数	説明
letterrange	letterまたはletter-letterの場合: AからZの何れかの文字列。letter-letterを使用する場合、アルファベット順で最初のletterは2番目のletterの前でなければなりません。この範囲のletterで始まる変数名は、デフォルトで宣言された型になります。 変数名が任意のletterrangeではないletterで始まる場合、その変数はVariantです。letterrangeを重複させることはできません。

Option Explicitも参照してください。

例

```
DefInt A,C-W,Y' integer
DefBool B ' boolean
DefStr X ' string
' all others are variant
Sub Main
B = 1 ' B is an boolean
Debug.Print B ' True
X = "A" ' X is a string
Debug.Print X '"A"
Z = 1 ' Z is a variant (anything)
Debug.Print Z ' 1
Z = "Z"
Debug.Print Z '"Z"
End Sub
```

Dim Definition

Dimを使用して各次元の最小および最大インデックスを設定してvar配列を作成します。Dimを省略すると、スカラ(単一値)変数が定義されます。動的配列は要素数を指定せず、()と宣言します。ReDimで要素数を宣言し直して使用します。

Begin Dialog、Dialog、Option Base、Private、Public、ReDim、Staticも参照してください。

```
Dim name[type][([dim[, ...]])][As [New] type][, ...]
```

例

```
Sub DoIt(Size)
Dim C0,C1(),C2(2,3)
ReDim C1(Size) ' dynamic array
C0 = 1
C1(0) = 2
C2(0,0) = 3
Debug.Print C0;C1(0);C2(0,0) ' 1 2 3
End Sub
Sub Main
DoIt 1
End Sub
```

Do Statement

Form 1: ループを繰り返します。Exit またはGotoを使用するとループを終了させることができます。

```
Do
statements
Loop
```

Form 2: ループを繰り返す前に終了条件をチェックします。

```
Do {Until|While} condexpr
statements
Loop
```

Form 3: ループの繰り返し後に終了条件をチェックします。ループは少なくとも1回は実行されます。

```
Do
statements
Loop {Until|While} condexpr
```

ループの終了

- Until condexpr: condexprがTrueになるまで Doステートメントを実行します。
- While condexpr: condexprがTrueの間は Doステートメントを実行します。

For、For Each、Exit Do、Whileも参照してください。

例

```
Sub Main
I = 2
Do
I = I*2
Loop Until I > 10
Debug.Print I ' 16
End Sub
```

Enum Definition

新規のuserenumを定義します。各elemは列挙型の要素を定義します。valueを設定している場合、要素の値です。値は任意の整数定数式で構いません。valueを省略した場合、要素の値は前の要素より一つ多くなります。前に要素がない場合、ゼロを使用します。
PrivateまたはPublicのいずれも指定していない場合、Enumはデフォルトで Publicになります。

```
[ | Private | Public ] _
Enum name
elem [ = value]
[...]
End Enum
```

例

```
Enum Days
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
End Enum
Sub Main
Dim D As Days
For D = Monday To Friday
Debug.Print D ' 0 through 4
Next D
End Sub
```

For Statement

FirstからLastの範囲のNumの回数のステートメントを実行します。

```
For Num = First To Last [Step Inc]
  statements
Next [Num]
```

変数	説明
Num	反復変数。
First	Numの初期値。
Last	Numの最終値。Numが範囲内にある間、ループを繰り返します。
Step	ループを繰り返すごとにNumに加算される値を指定。 この数値が0より大きい場合、forループは、NumがLast以下である限り継続します。 この数値が0より小さい場合、forループは、NumがLast以上であれば継続します。 Stepを省略すると、ループを繰り返すごとにNumに1が加算されます。

Do、For Each、Exit For、Whileも参照してください。

例

```
Sub Main
For I = 1 To 2000 Step 100
Debug.Print I;I+I;I*I
Next I
End Sub
```

For Each Statement

Items内の各項目のステートメントを実行します。

```
For Each var In items
statements
Next [var]
```

変数	説明
var	反復変数。
Items	実行する項目のコレクション。

Do、For、Exit For、Whileも参照してください。

例

```
Sub Main
Dim Document As Object
For Each Document In App.Documents
Debug.Print Document.Title
Next Document
End Sub
```

Function Definition

ユーザ定義関数。この関数は呼び出されたときに実行される一連のステートメントを定義します。呼び出すarglistの値はparamsに割り当てられます。name[type]に代入すると、関数の結果の値が設定されます。
Private、PublicまたはFriendが指定されていない場合、関数のデフォルトはPublic です。

Declare、Property、Subも参照してください。

```
[ | Private | Public | Friend ] _
[ Default ] _
Function name[type][([param[, ...]])] [As type[()]]
statements
End Function
```

例

```
Function Power(X,Y)
P = 1
For I = 1 To Y
  P = P*X
Next I
Power = P
End Function

Sub Main
Debug.Print Power(2,8) ' 256
End Sub
```

If Statement

Form 1: 一行のIfステートメント。condexprがTrueの場合、Thenに続く命令を実行します。それ以外は、Elseの以後の命令を実行します。Elseはオプションです。

```
If condexpr Then [instruction] [Else instruction]
```

Form 2: 複数行のifは、複雑なifに有用です。各Ifでcondexprが順にチェックされます。最初のTrueが実行されると、次のステートメントが実行されます。すべてがFalseの場合、Elseステートメントが実行されます。ElseIfおよびElseはオプションです。

```
If condexpr Then
statements
[ElseIf condexpr Then
statements]...
[Else
statements]
End If
```

Form 3: objexprが同じ型または同じobjtypeから参照している場合、Thenに続く命令を実行します。

```
If TypeOf objexpr Is objtype Then ...
```

Select Case、Choose( )、IIf( )も参照してください。

例

```
Sub Main
S = InputBox("Enter hello, goodbye, dinner or sleep:")
S = UCase(S)
If S = "HELLO" Then Debug.Print "come in"
If S = "GOODBYE" Then Debug.Print "see you later"
If S = "DINNER" Then
Debug.Print "Please come in."
Debug.Print "Dinner will be ready soon."
ElseIf S = "SLEEP" Then
Debug.Print "Sorry."
Debug.Print "We are full for the night"
End If
End Sub
```

Is Operator

二つのオブジェクト変数を比較します。二つのexprが同じオブジェクトを参照している場合はTrueを返します。

```
expr Is expr
```

Objectsも参照してください。

例

```
Sub Main
S = InputBox("Enter hello, goodbye, dinner or sleep:")
Dim X As Object
Dim Y As Object
Debug.Print X Is Y ' True
End Sub
```

Like Operator

文字列をパターンと比較します。str1とstr2とのパターン一致する場合はTrueを返します。str2のパターンは、次の表に示す1つ以上の特殊文字列です。

```
str1 Like str2
```

Char(s)	説明
?	任意の1文字に一致。
*	0個以上の文字に一致。
#	任意の一桁(0–9)に一致。
[charlist]	リスト内の任意の1文字に一致。
[!charlist]	リストに含まれていない任意の1文字に一致。

例

```
Sub Main
Debug.Print       "abcdfgcdefg" Like "" ' False
Debug.Print       "abcdfgcdefg" Like "a*g" ' True
Debug.Print       "abcdfgcdefg" Like "a*cde*g" ' True
Debug.Print       "abcdfgcdefg" Like "a*cd*cd*g" ' True
Debug.Print       "abcdfgcdefg" Like "a*cd*cd*g" ' True
Debug.Print       "00aa" Like "####" ' False
Debug.Print       "00aa" Like "????" ' True
Debug.Print       "00aa" Like "##??" ' True
Debug.Print       "00aa" Like "*##*" ' True
Debug.Print       "hk" Like "hk*" ' True
End Sub
```

Option Definition

Form 1: 配列宣言のデフォルトの下限を設定します。Dim、Static、Private、Public、およびReDimに影響します。Typeで宣言しているArray、ParamArrayや配列には影響しません。Option Baseは、0がデフォルトです。

```
Option Base [0|1]
```

Form 2: 文字列のデフォルトの比較モードを設定します。

- Option Compare Binary - バイナリデ表現による順序に基づき文字列を比較(デフォルト)
- OptionCompare Text - システムのロケールに基づき文字列を比較

<、<=、=、>、> =、<>、LikeおよびStrCompを使用した文字列比較は、このモード設定の影響を受けます。

```
Option Compare [Binary | Text]
```

Form 3: 使用前にすべての変数を宣言する必要があります。変数は、Dim、Private、Public、Static、またはSub、Function、Propertyブロックのパラメータとして宣言されます。

```
Option Explicit
```

Form 4: モジュールで定義したpublic部分は、同じプロジェクト内からのみアクセスできます。

```
Option Private Module
```

例

```
Option Base 1
Option Explicit
Sub Main
Dim A
Dim C(2) ' same as Dim C(1 To 2)
Dim D(0 To 2)
A = 1
B = 2 ' B has not been declared
End Sub
```

Private Definition

モジュール内で使用可能ですが、他のモジュールでは使用できない配列(または単純変数)を作成します。dimを使用して各次元の最小値と最大値を設定するvar配列を次元化します。dimを省略するとスカラ(単一値)変数が定義されます。動的配列は、dimを指定せずに()を使用して宣言します。使用するには、ReDimで宣言し直します。Privateステートメントは、Sub、FunctionまたはPropertyブロックの外側に配置する必要があります。

```
Private name[type][([dim[, ...]])] [As [New] type][, ...]
```

dim、Option Base、Public、ReDim、Staticも参照してください。

例

```
Private A0,A1(1),A2(1,1)
Sub Init
A0 = 1
A1(0) = 2
A2(0,0) = 3
End Sub
Sub Main
Init
Debug.Print A0;A1(0);A2(0,0) ' 1 2 3
End Sub
```

Property Definition

ユーザ定義プロパティ。

```
[ | Private | Public | Friend ] _
[ Default ] _
Property Get name[type][([param[, ...]])] [As type[()]]
statements
End Property
```

または

```
[ | Private | Public | Friend ] _
Property [Let|Set] name[([param[, ...]])]
statements
End Property
```

このプロパティは、値が使用または変更されたときに実行される一連のステートメントを定義します。プロパティは変数のように機能しますが、値を取得する場合はProperty Getが呼び出され、変更する場合はProperty Let(またはProperty Set)が呼び出されます。Property GetおよびProperty Letは、同じnameの値を保持するプロパティを定義します。Property GetおよびProperty Letは、同じnameの値を保持するプロパティを定義します。Property GetおよびProperty Setは、同名のオブジェクト参照を保持するプロパティを定義します。呼び出すarglistの値はparamsに割り当てられます(Property GetおよびProperty Setでは、最後のパラメータは代入演算子の右辺の値です)。
Private、PublicまたはFriendが指定されていない場合、Propertyのデフォルトは Publicとなります。

Function、Subも参照してください。

例

```
Dim X_Value
Property Get X()
X = X_Value
End Property
Property Let X(NewValue)
If Not IsNull(NewValue) Then X_Value = NewValue
End Property

Sub Main
X = "Hello"
Debug.Print X
X = Null
Debug.Print X
End Sub
```

Public Definition

モジュール内や他のモジュールで使用できる配列(または単純変数)を作成します。

```
Public name[type][([dim[, ...]])] [As [New] type][, ...]
```

dimを使用して各次元の最小値と最大値を設定するvar配列を次元化します。dimを省略するとスカラ(単一値)変数が定義されます。動的配列は、dimを指定せずに()を使用して宣言します。使用するには、ReDimで宣言し直します。Publicステートメントは、Sub、FunctionまたはPropertyブロックの外側に配置する必要があります。

dim、Option Base、Public、ReDim、Staticも参照してください。

例

```
Public A0,A1(1),A2(1,1)
Sub Init
A0 = 1
A1(0) = 2
A2(0,0) = 3
End Sub
Sub Main
Init
Debug.Print A0;A1(0);A2(0,0) ' 1 2 3
End Sub
```

Reference Comment

Reference commentはタイプライブラリで識別した、現在のモジュールを参照して表示します。

```
'#Reference {uuid}#vermajor.verminor#lcid#[path[#name]]
```

参照コメント行は、モジュール内の最初の行でなければなりません。参照コメントは優先順位が逆(低から高へ)です。エディタでは参照コメントは表示されません。

変数	説明
uuid	タイプライブラリの汎用一意識別子。
vermajor	タイプライブラリのメジャーバージョン番号。
verminor	タイプライブラリのマイナーバージョン番号。
lcid	タイプライブラリのロケール ID。
path	タイプライブラリのパス。
name	タイプライブラリの名前。

例

```
'#Reference {00025E01-0000-0000-C000-000000000046}#4.0#0#C:\PROGRAM FILES\COMMON FILES\MICROSOFT SHARED\DAO\DAO350.DLL#Microsoft DAO 3.5 Object Library
```

Select Case Statement

exprが各caseexprを比較し、適切なケースを選択します。

```
Select Case expr
[Case caseexpr[, ...]
statements]...
[Case Else
statements]
End Select
```

caseexprが一致しない場合は、Case Else部を選択します(Case Elseが省略されている場合は、Select ... End Selectの全体をスキップします)。

変数	説明
expr	等しい場合は実行。
Is < expr	未満の場合は実行。
Is <= expr	以下の場合は実行。
Is > expr	超過の場合は実行。
Is >= expr	以上の場合は実行。
Is <> expr	等しくない場合は実行。
expr1 To expr2	expr1以上かつexpr1以下の場合に実行。

If、Choose()、IIf()も参照してください。

例

```
Sub Main
S = InputBox("Enter hello, goodbye, dinner or sleep:")
Select Case UCase(S)
Case "HELLO"
Debug.Print "come in"
Case "GOODBYE"
Debug.Print "see you later"
Case "DINNER"
Debug.Print "Please come in."
Debug.Print "Dinner will be ready soon."
Case "SLEEP"
Debug.Print "Sorry."
Debug.Print "We are full for the night"
Case Else
Debug.Print "What?"
End Select
End Sub
```

Set Instruction

Form 1:　objvarオブジェクトの参照にobjexprオブジェクトの参照を設定します。

```
Set objvar = objexpr
```

Form 2:　objvarオブジェクトの参照に新規objtypeのインスタンを設定します。

Set instructionは、オブジェクトの参照が割り当てられています。

```
Set objvar = New objtype
```

例

```
Sub Main
Dim App As Object
Set App = CreateObject("WinWrap.CppDemoApplication")
App.Move 20,30 ' move icon to 20,30
Set App = Nothing
App.Quit ' run-time error (no object)
End Sub
```

Static Definition

静的変数は、プロシージャ呼び出し間で値を保持します。

```
Static name[type][([dim[, ...]])][As [New] type][, ...]
```

Dimを使用して各次元の最小および最大指標を設定してvar配列を作成します。Dimを省略すると、スカラ(単一値)変数が定義されます。動的配列は要素数を指定せず、()と宣言します。ReDimで要素数を宣言し直して使用します。

dim、Option Base、Public、Private、ReDimも参照してください。

例

```
Sub A
Static X
Debug.Print X
X = "Hello"
End Sub
Sub Main
A
A ' prints "Hello"
End Sub
```

Sub Definition

ユーザ定義のサブルーチン。

```
[ | Private | Public | Friend ] _
Sub name[([param[, ...]])]
statements
End Sub
```

サブルーチンは、呼び出されるときに実行される一連のステートメントを定義します。呼び出すarglistの値はparamsに割り当てられます。サブルーチンは結果を返しません。
Private、Public、またはFriendが指定されていない場合、Subのデフォルトは Publicに設定されます。

Declare、Function、Propertyも参照してください。

例

```
Sub IdentityArray(A()) ' A() is an array of numbers
For I = LBound(A) To UBound(A)
A(I) = I
Next I
End Sub
Sub CalcArray(A(),B,C) ' A() is an array of numbers
For I = LBound(A) To UBound(A)
A(I) = A(I)*B+C
Next I
End Sub
Sub ShowArray(A()) ' A() is an array of numbers
For I = LBound(A) To UBound(A)
Debug.Print "(";I;")=";A(I)
Next I
End Sub
Sub Main
Dim X(1 To 4)
IdentityArray X() ' X(1)=1, X(2)=2, X(3)=3, X(4)=4
CalcArray X(),2,3 ' X(1)=5, X(2)=7, X(3)=9, X(4)=11
ShowArray X() ' print X(1), X(2), X(3), X(4)
End Sub
```

Type Definition

新規のユーザタイプを定義します。

```
[ | Private | Public ] _
Type name
elem [([dim[, ...]])] As [New] type
[...]
End Type
```

各要素は、データを格納する要素の型を定義します。[New] typeは、保存可能なデータの種類を定義します。ユーザ定義型変数は、各要素に値を持ちます。個々の要素の値にアクセスするにはelemを使用します。
PrivateまたはPublicが指定されていない場合、型のデフォルトはPublicに設定されます。

例

```
Type Employee
FirstName As String
LastName As String
Title As String
Salary As Double
End Type
Sub Main
Dim e As Employee
e.FirstName = "John"
e.LastName = "Doe"
e.Title = "President"
e.Salary = 100000
Debug.Print e.FirstName '"John"
Debug.Print e.LastName '"Doe"
Debug.Print e.Title '"President"
Debug.Print e.Salary ' 100000
End Sub
```

Uses Comment

Uses commentは現在のモジュールが、publicおよびfriendシンボルを使用することを示します。

```
'#Uses "module" [Only:[Win16|Win32]] ...
```

または

```
'$Include: "module"
```

Onlyオプションは、モジュールがWindowsプラットフォームにのみ読み込まれることを示します。

変数	説明
module	このモジュールのpublicシンボルとfriendシンボルにアクセスできます。モジュール名が相対パスの場合、パスはUses commentを含むモジュールに相対的です。たとえば、モジュール"A:\B\C\D.BAS"にUses commentがある場合、次のようになります: '#Uses "E.BAS" "A:\B\C\E.BAS"を使用します。

Class Module、Code Moduleも参照してください。

例

```
'A.BAS
'#Uses "B.BAS"
Sub Main
Debug.Print BFunc$("Hello") '"HELLO"
End Sub
'B.BAS
Public Function BFunc$(S$)
BFunc$ = UCase(S$)
End Function
```

While Statement

condexprがTrueである間、ステートメントを実行します。

```
While condexpr
statements
Wend
```

Do、For、For Each、Exitも参照してください。

例

```
Sub Main
I = 2
While I < 10
I = I*2
Wend
Debug.Print I ' 16
End Sub
```

定義補助

array variable definition ・・・

array variable definition

配列の値を保持する変数です。バリアント型の変数は配列を保持することができます。動的配列は再次元化することができます。

arglist definition

プロシージャのパラメータに割り当てられる0個以上のexprのリスト。

```
[ | expr | param:=expr ][, ...]
```

- 式を省略すると、位置パラメータをスキップできます。省略可能なパラメータのみをスキップできます。
- 位置パラメータの割り当ては exprで行われます。各パラメータは順番に割り当てられます。名前によるパラメータの割り当てが続く場合があります。
- 名前によるパラメータの割り当ては、param:=exprで行われます。以降のパラメータはすべて名前で割り当てる必要があります。

attribute definition

ファイル属性は、以下の値を0個以上加算したものです。

属性	値	説明
bNormal	0	通常ファイル。
bReadOnly	1	読み取り専用ファイル。
bHidden	2	隠しファイル。
bSystem	4	システムファイル。
bVolume	8	ボリュームラベル。
bDirectory	16	MS-DOSディレクトリ。
bArchive	32	前回のバックアップ以降にファイルが変更されています。

big-endian definition

複数バイトのデータ値(文字列ではない)は、最上位バイトが最初に格納されます。たとえば、長整数 &H01020304は、&H01、&H02、&H03、および&H04の4バイトのシーケンスとして格納されます。Putを使用して書き込まれたバイナリまたはランダムファイルはリトルエンディアン形式を使用するため、どのマシンでもGetを使用して読み取ることができます。(Power-PC のようなビッグ エンディアン マシンは、Getによって読み取られるとき、またはPutによって書き込まれるときにバイトを反転します。)

Dir( )、GetAttr( )、SetAttr( )も参照してください。

charlist definition

Like演算子の右側の文字列式の一部として、[ ]で囲まれた1つ以上の文字のグループ。

- このリストには、リスト内の文字を説明する単一の文字および/または文字範囲が含まれます。
- 文字の範囲は、2つの文字の間のハイフン(-)で示されます。通常、最初の文字は2番目の文字以下である必要があります。
- ?、*、#、[などの特殊なパターン文字は、リテラル文字として一致します。
-]文字をcharlistの一部にすることはできませんが、charlist外のパターンの一部にすることはできます。

condexpr definition

数値結果を返す式。結果がゼロの場合、条件はFalseです。結果がゼロ以外の場合、条件はTrueです。

```
0 'false
-1 'true
X > 20 'true if X is greater than 20
S$ = "hello" 'true if S$ equals "hello"
```

dateexpr definition

日付の結果を返す式。#literal-date# を使用して日付値を表現します。

```
#1/1/2000# ' Jan 1, 2000
Now+7 ' seven days from now
DateSerial(Year(Now)+1,Month(Now),Day(Now))
' one year from now
```

dialogfunc definition

UserDialogが表示されている間、ダイアログ関数は実行されます。

dlgvar definition

ダイアログ変数は、ダイアログ内のフィールドの値を保持します。ダイアログ変数は、Dim dlgvar As UserDialogを使用して宣言します。

expr definition

適切な結果を返す式です。

field definition

.fieldを使用して、ダイアログ変数の個々のフィールドにアクセスします。

```
dlg.LastName$
dlg.ZipCode
```

instruction definition

単一のコマンド。

```
Beep
Debug.Print "Hello"
Today = Date
```

単一の命令をコロンで区切ることにより、複数の命令を使用することができます。

```
X = 1:Debug.Print X
If X = 1 Then Debug.Print "X=";X:Stop
Beep ' must resume from Stop to get to here
```

label definition

命令に名前を付ける識別子。識別子は文字で始まります。続く文字は、文字、アンダースコア、または数字です。

little-endian definition

複数バイトのデータ値(文字列ではない)は、最下位バイトが最初に格納されます。たとえば、長整数&H01020304は、&H04、&H03、&H02、および&H01の4バイトのシーケンスとして格納されます。Putを使用して書き込まれたバイナリまたはランダムファイルはリトルエンディアン形式を使用するため、どのマシンでもGetを使用して読み取ることができます。(Power-PCのようなビッグエンディアンマシンは、Getによって読み取られるとき、またはPutによって書き込まれるときにバイトを反転します。)

method definition

オブジェクトはメソッドとプロパティを提供します。

```
App.[Title$]
```

メソッドはサブルーチンとして呼び出すことも(戻り値は無視します)、関数として使用することもできます(戻り値は使用します)。
メソッド名に名前として使用できない文字が含まれている場合は、メソッド名を[]で囲みます。

module definition

#Usesコメントを介して他のモジュールからアクセスできるパブリックシンボルを含むファイル。

- モジュールはオンデマンドで読み込まれます。
- コードモジュールはコードライブラリです。
- クラスモジュールはActiveXオートメーションオブジェクトを実装します。
- モジュールは、独自の#Usesコメントを使用して他のモジュールにアクセスすることもできます。

name definition

識別子は、その変数名またはユーザ定義のプロシージャです。識別子は文字で始まります。以降は文字、アンダースコア、数字でも構いません。

```
Count
DaysTill2000
Get_Data
```

num definition

数値結果を返す式。&Oを使用して8進数を表現します。&Hを使用して16進数を表現します。

numvar definition

1つの数値を保持する変数。数値変数の名前の後には、適切な型文字が続く場合があります。

objexpr definition

オブジェクトまたはモジュールへの参照を返す式。

```
CreateObject("WinWrap.CDemoApplication")
```

objtype definition

アプリケーション、別のアプリケーション、またはオブジェクト モジュールまたはクラスモジュールによって定義された特定のActiveXオートメーションタイプ。

オブジェクトの概要、CreateObject( )、GetObject( )も参照してください。

objvar definition

オブジェクトを参照するobjexprを保持する変数。オブジェクト変数は、Dim、Private、またはPublicステートメントでAs Objectを使用して宣言されます。

オブジェクトの概要も参照してください。

param definition

param は、Declare、Sub、Function、またはProperty呼び出しで関連付けられた式の値を受け取ります(引数リストを参照してください)。

```
  [ [Optional] [ | ByVal | ByRef ] | ParamArray ] param[type][( )] [As type] [ =
defaultvalue ]
```

- オプションのparamは呼び出しから省略できます。defaultvalueを持つ場合もあります。呼び出しで値が指定されていない場合、パラメータはデフォルト値を受け取ります。デフォルト値が省略され、パラメータがバリアントであり、呼び出しで値が指定されていない場合、IsMissingはTrueを返します。

- Optionalパラメータに続くすべてのパラメータもOptionalである必要があります。

- ParamArrayは最終paramで使用できます。Variant型の配列である必要があります。任意のパラメータの後に続けてはいけません。ParamArrayは、呼び出しの終了時にすべての式を配列として受け取ります。LBound(param) > UBound(param) の場合、ParamArrayは式を受け取りません。

- paramがByValではなく、式が単なる変数である場合、paramはその変数(ByRef)への参照になります(paramを変更すると変数が変更されます)。それ以外の場合、パラメータ変数はプロシージャに対してローカルであるため、その値を変更しても呼び出し元には影響しません。

- param( )を使用して配列パラメータを指定します。配列パラメータは参照する必要があり、値で渡すことはできません。パラメータ配列の境界は、LBound( )およびUBound( )を介して取得できます。

precedence definition

式で複数の演算子を使用する場合、各演算子はあらかじめ決められた順序で評価されます。演算子は次の順序で評価されます:

- ^ (指数)
- - (反転)
- * (乗算)、/ (除算)
- \ (integer divide)
- Mod (整数除算)
- + (加算)、- (減算)
- & (文字列連結)
- = (等しい)、<> (等しくない)、< (より小さい)、> (より大きい)、<= (以下)、>= (以上)、Like(文字列の類似性)、Is(オブジェクトの等価性)
- Not (Bit単位で整数値の論理否定)
- And (Bit単位の論理積)
- Or (論理またはBit単位の論理積)
- Xor (論理またはBit単位の排他的論理和)
- Eqv (論理またはBit単位の論理等価)
- Imp (論理またはBit単位の論理包含)

同じ行に示されている演算子は、左から右に評価されます。

procedure definition

サブルーチン、関数またはプロパティ。

statement definition

0個以上の命令。ステートメントは少なくとも1行の長さです。Begin Dialog、Do、For、If(複数行)、Select Case、whileおよびWithステートメントは常に 1 行以上の長さになります。行がスペースとアンダースコア' _'で終わっている場合、単一行のステートメントは次の行に続きます。

```
S$ = "This long string is easier to read, " + _
"if it is broken across two lines."
Debug.Print S$
```

str definition

文字列の結果を返す式。

```
"Hello"
S$
S$ + " Goodbye"
S$ & " Goodbye"
Mid$(S$,2)
```

strarray definition

文字列値の配列を保持する変数。文字列変数の名前の後に$が続く場合があります。

strvar definition

文字列値を保持する変数。文字列変数の名前の後に$が続く場合があります。

type definition

変数とパラメータの型、および関数とプロパティの結果は、名前の最後の文字として型文字を使用して指定できます。

型文字	説明
%	Integer(整数型)
?	PortInt(移植可能整数型)
&	Long(長整数型)
!	Single(単精度浮動小数点数型)
#	Double(倍精度浮動小数点数型)
@	Currency(通貨型)
$	String(文字列型)

userenum definition

ユーザ定義の列挙型はEnumで定義されます。

usertype definition

ユーザー定義タイプはType で定義されます。

usertypevar definition

ユーザ定義型の変数の値はユーザ定義型の値を保持します。ユーザ定義型はType を使用して保持します。

- Dim、Private、PublicまたはStaticとともに宣言します。
- Sub、FunctionやProperty定義の引数として定義します。

var definition

変数は、その型に応じて、文字列、数値、または値の配列のいずれかを保持します。

variantvar definition

バリアント変数の値は任意の型の値を保持(String*nやusertypevarを除く)、または配列を保持することができます。

オブジェクト

オブジェクトの概要 ・・・

オブジェクトの概要

ActiveX Automationは、他のアプリケーションのオブジェクトへのアクセスを提供します。各オブジェクトは、特定のメソッドおよびプロパティのセットをサポートします。パラメータはオプションでも良く、その場合のパラメータはname:= valueを使用して指定します。

- objexpr.method [expr][, ...] [param := expr][,...]

  Call method for objexpr.
- objexpr.method[([expr][, ...] [param := expr][,...])]

  Return the value of method for objexpr.
- objexpr.property[([expr][, ...] [param := expr][,...])]

  Return the value of property for objexpr.
- objexpr[([expr][, ...] [param := expr][,...])]

  Return the default value for the objexpr.
- objexpr.property[([expr][, ...])] = expr

  Assign the value of property for objexpr.
- objexpr[([expr][, ...])] = expr

  Assign the default value for the objexpr.
- Set objexpr.property[([expr][, ...])] = objexpr

  Set the object reference of property for objexpr.

objexpr!nameはobjexpr.defaultproperty("name")の略語です。識別できない任意の文字列を含む場合、objexpr![name]を使用します。

CreateObject Function

Class$型の新規オブジェクトを作成します。Setを使用して、返されたオブジェクトをオブジェクト変数に割り当てます。Setを使用して、返されたオブジェクトをオブジェクト変数に割り当てます。

```
CreateObject(Class$)
```

変数	説明
Class$	この文字列は、アプリケーションが登録したクラス名です。アプリケーションが現在アクティブでない場合、起動されます。

オブジェクトの概要も参照してください。

例

```
Sub Main
Dim App As Object
Set App = CreateObject("WinWrap.CppDemoApplication")
App.Move 20,30 ' move icon to 20,30
Set App = Nothing
App.Quit ' run-time error (no object)
End Sub
```

Debug Object

Form 1: 出力ウィンドウをクリアします。

```
Debug.Clear
```

Form 2: expr(s)を出力ウィンドウに表示します。セミコロン(;)で要素を区切ります。NumはPrint前に自動的に文字列に変換されます(Str $()のように)。命令がセミコロン(;)で終わらない場合、最後に改行が出力されます。

```
Debug.Print [expr[; ...][;]]
```

例

```
Sub Main
X = 4
Debug.Print "X/2=";X/2 ' 2
Debug.Print "Start..."; ' don't print a newline
Debug.Print "Finish" ' print a newline
End Sub
```

Err Object

最後のエラーイベントをクリアするには、Errをゼロに設定します。式のErrは最後のエラーコードを返します。Err.RaiseまたはErrorを使用してエラーイベントをトリガします。

```
Err[.Number]
```

これは最後のエラーイベントのエラーコードです。最後のエラー状態をクリアするには、ゼロに設定します(またはErr.Clearを使用します)。ErrorまたはErr. Raiseを使用してエラーイベントをトリガします。これはデフォルトのプロパティです。

```
Err.Description
```

この文字列は最後のエラーイベントの説明です。

```
Err.Source
```

この文字列は最後のエラーイベントのエラーソースファイル名です。

```
Err.HelpFile
```

この文字列は最後のエラーイベントのヘルプファイル名です。

```
Err.HelpContext
```

この番号は最後のエラーイベントのヘルプコンテキストIDです。

```
Err.Clear
```

最後のエラーイベントをクリアします。

```
Err.Raise [Number:=]errorcode _
[, [Source:=]source] _
[, [Description:=]errordesc] _
[, [HelpFile:=]helpfile] _
[, [HelpContext:=]context]
```

エラーイベントを発生させます。

```
Err.LastDLLError
```

32 bit Windowsの場合、最後のDLL呼び出しのエラーコードを返します(Declareを参照)。16 bit Windowsの場合、常に0を返します。

例

```
Sub Main
On Error GoTo Problem
Err = 1 ' set to error #1 (handler not triggered)
Exit Sub
Problem: ' error handler
Error Err ' halt execution with message
End Sub
```

GetObject Function

Class$からFile$までの既存オブジェクトを取得します。Setを使用してオブジェクトを返し、オブジェクト変数に割り当てます。

```
GetObject([File$][, Class$])
```

変数	説明
File$	オブジェクトが含まれているファイル。省略した場合、Class$の現在のアクティブなオブジェクトを返します。
Class$	この文字列はアプリケーションが登録したクラス名です。アプリケーションがアクティブでない場合、起動されます。省略した場合、ファイルの拡張子に関連付けられたアプリケーションを起動します。

例

```
Sub Main
Dim App As Object
Set App = GetObject(,"WinWrap.CppDemoApplication")
App.Move 20,30 ' move icon to 20,30
Set App = Nothing
App.Quit ' run-time error (no object)
End Sub
```

Me Object

Meは現在のモジュールを参照します。他のオブジェクト変数と同じように使用できますが、参照は変更できません。

```
Me
```

Setも参照してください。

例

```
Sub Main
DoIt
Me.DoIt ' calls the same sub
End Sub
Sub DoIt
MsgBox "Hello"
End Sub
```

With Statement

Withブロック内でメソッドおよびプロパティの参照を省略することができます。With objexprで指定されたオブジェクトにアクセスするには、.methodまたは.propertyを使用します。

```
With objexpr
statements
End With
```

例

```
Sub Main
Dim App As Object
Set App = CreateObject("WinWrap.CppDemoApplication")
With App
.Move 20,30 ' move icon to 20,30
End With
End Sub
```

ダイアログ

Begin Dialog Definition ・・・

Begin Dialog Definition

後でDim As UserDialogステートメントで使用するUserDialog型を定義します。

```
Begin Dialog UserDialog [X, Y,] DX, DY[, Title$] [, .dialogfunc]
User Dialog Item
[User Dialog Item]...
End Dialog
```

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
Title$	ユーザダイアログのタイトル文字列。省略した場合、タイトルを表示しません。
dialogfunc	UserDialogのDialogFuncを実装する関数名。省略した場合、UserDialogにdialogfuncは実装されません。
User Dialog Item	CancelButton、CheckBox、ComboBox、DropListBox、GroupBox、ListBox、MultiListBox、OKButton、OptionButton、OptionGroup、PushButton、Text、TextBoxのいずれかです。

Dim As UserDialogも参照してください。

例

```
Sub Main
Begin Dialog UserDialog 200,120
Text 10,10,180,15,"Please push the OK button"
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
Dialog dlg ' show dialog (wait for ok)
End Sub
```

CancelButton Dialog Item Definition

キャンセルボタンの項目を定義します。Dialog instructionからCancelボタンを押すと、ランタイムエラーが発生します(Dialog()関数を呼び出しすと0を返します)。

```
CancelButton X, Y, DX, DY[, .Field]
```

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
Field	この識別子は、フィールド名です。dialogfuncはこの名前を文字列として受け取ります。省略した場合、フィールド名は"キャンセル"になります。

Begin Dialog、Dim As UserDialogも参照してください。

例

```
Sub Main
Begin Dialog UserDialog 200,120
Text 10,10,180,30,"Please push the Cancel button"
OKButton 40,90,40,20
CancelButton 110,90,60,20
End Dialog
Dim dlg As UserDialog
Dialog dlg ' show dialog (wait for cancel)
Debug.Print "Cancel was not pressed"
End Sub
```

CheckBox Dialog Item Definition

チェックボックスの項目を定義します。

```
CheckBox X, Y, DX, DY,Title$, .Field[, Options]
```

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
Field	チェックボックスの値には、このフィールドからアクセスします。チェックされていない場合は0、チェックされている場合は1、グレーアウトの場合は2です。
Options	この数値は、チェックボックスのタイプを制御します。以下から一つを選択します(省略した場合、ゼロを使用します)。 0: チェック、または未チェックの何れか 1: チェック、未チェックまたはグレーアウトの何れかで、クリックでチェックと未チェックを切り替え 2: チェック、未チェックまたはグレーアウトの何れかで、クリックで三つの状態を循環

Begin Dialog、Dim As UserDialogも参照してください。

例

```
Sub Main
Begin Dialog UserDialog 200,120
Text 10,10,180,15,"Please push the Cancel button"
CheckBox 10,25,180,15,"&Check box",.Check
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
dlg.Check = 1
Dialog dlg ' show dialog (wait for ok)
Debug.Print dlg.Check
End Sub
```

ComboBox Dialog Item Definition

コンボボックスの項目を定義します。コンボボックスは、エディットボックスとリストボックスの機能を組み合わたものです。

```
ComboBox X, Y, DX, DY, StrArray$( ), .Field$[, Options]
```

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
StrArray$( )	選択肢のリストを決定する一次元配列の文字列。配列のすべての要素にnull以外を使用します。
Field$	コンボボックスの値には、このフィールドでアクセスします。これはエディットボックス内のテキストです。
Options	この数値は、チェックボックスのタイプを制御します。以下から一つを選択します(省略した場合、ゼロを使用します)。 0: リストをソートしない 2: リストをソートする

Begin Dialog、Dim As UserDialogも参照してください。

例

```
Sub Main
Dim combos$(3)
combos$(0) = "Combo 0"
combos$(1) = "Combo 1"
combos$(2) = "Combo 2"
combos$(3) = "Combo 3"
Begin Dialog UserDialog 200,120
Text 10,10,180,15,"Please push the OK button"
ComboBox 10,25,180,60,combos$(),.combo$
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
dlg.combo$ = "none"
Dialog dlg ' show dialog (wait for ok)
Debug.Print dlg.combo$
End Sub
```

Dialog Instruction/Function

```
Dialog dialogvar[, default]
```
または
```
Dialog(dialogvar[, default])
```

変数	説明
dlgvar	この変数は、ダイアログ内のフィールド値を保持します。. field を使用して、ダイアログ変数の個々のフィールドにアクセスします。
default	この値は、どのボタンがデフォルトのボタンであるかを示します(Enterキーを押すと、ボタンをクリックすることなく、デフォルトのボタンが押されます)。デフォルトのボタン指定しない場合、-2を使用します。利用可能な値を以下に示します。この値を省略した場合、最初のPushButton、OKButtonまたはCancel Buttonがデフォルトのボタンになります。 -2: デフォルトの指定無し -1: OKボタン 0: キャンセルボタン >0: n番目の押しボタン

Begin Dialog、Dim As UserDialogも参照してください。

例

```
Sub Main
Begin Dialog UserDialog 200,120
Text 10,10,180,15,"Please push the OK button"
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
Dialog dlg ' show dialog (wait for ok)
End Sub
```

DialogFunc Prototype

動的ダイアログ機能を実装します。

```
Function dialogfunc(DlgItem$, Action%, SuppValue%) _
As Boolean
Select Case Action%
Case 1 ' Dialog box initialization
...
Case 2 ' Value changing or button pressed
...
Case 3 ' TextBox or ComboBox text changed
...
Case 4 ' Focus changed
...
Case 5 ' Idle
...
Case 6 ' Function key
...
End Select
End Function
```

変数	説明
DlgItem	ユーザダイアログ項目のフィールド名の文字列です。
Action	値はダイアログ関数に求められているアクションを示します。下表を参照。
SuppValue	Actionに関する追加情報を提供します。

Action	説明
1	ダイアログボックスの初期化。DlgItemは空きの文字列です。SuppValueはダイアログのウィンドウハンドルです。dialogfunc = Trueを設定するとダイアログが終了します
2	CheckBox、DropListBox、ListBox、MultiListBoxまたはOptionGroup: DlgItem値の変更。SuppValueは新規値。 CancelButton、OKButtonまたはPushButton: DlgItemのボタンを押下。SuppValueは無効。ダイアログを閉じるのを防ぐには、dialogfunc = Trueを設定します。
3	ComboBoxまたはTextBox: DlgItemのテキストが変更され、フォーカスを解除します。SuppValueは文字数です。
4	DlgItem項目の変更。SuppValueは、フォーカスを解除している項目(最初の項目は0、2番目は1など)
5	アイドル。DlgItemは空の文字列。SuppValueはゼロ。アイドルアクションの受信を続行するには、 dialogfunc = Trueに設定します。アイドルアクションは可能な限り頻繁に呼び出されます。 `Wait .1` アイドルの呼び出しを0.1/秒に設定。
6	Functionキー(F1-F24)の押下。DlgItemがフォーカスを持っています。SuppValueはFunctionキー番号、Shift/Ctrl/Altキーの状態です。 標準的なFunctionキーは1から24までの範囲。 Shiftキーは&H100です。 Ctrlキーは&H200です。 Altキーは&H400です (Alt + F4はダイアログを閉じますが、Dialog関数に渡されません)。

Begin Dialogも参照してください。

例

```
Sub Main
Begin Dialog UserDialog 200,120,.DialogFunc
Text 10,10,180,15,"Please push the OK button"
TextBox 10,40,180,15,.Text
OKButton 30,90,60,20
PushButton 110,90,60,20,"&Hello"
End Dialog
Dim dlg As UserDialog
Debug.Print Dialog(dlg)
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Debug.Print "Action=";Action%
Debug.Print DlgItem$;"=""";DlgText$(DlgItem$);"""
Debug.Print "SuppValue=";SuppValue%
Select Case Action%
Case 1 ' Dialog box initialization
Beep
Case 2 ' Value changing or button pressed
If DlgItem$ = "Hello" Then
MsgBox "Hello"
DialogFunc% = True 'do not exit the dialog
End If
Case 4 ' Focus changed
Debug.Print "DlgFocus=""";DlgFocus();"""
Case 6 ' Function key
If SuppValue And &H100 Then Debug.Print "Shift-";
If SuppValue And &H200 Then Debug.Print "Ctrl-";
If SuppValue And &H400 Then Debug.Print "Alt-";
Debug.Print "F" & (SuppValue And &HFF)
End Select
```

DropListBox Dialog Item Definition

ドロップダウンリストボックスの項目を定義します。

```
DropListBox X, Y, DX, DY, StrArray$( ), .Field[, Options]
```

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
StrArray$()	選択肢のリストを決定する一次元配列の文字列。配列のすべての要素にnull以外を使用します。
Field$	ドロップダウンリストボックスの値には、このフィールドでアクセスします。これはStrArray$( )変数のインデックスです。
Options	この数値は、ドロップダウンリストボックスのタイプを制御します。以下から一つを選択します(省略した場合、0を使用します)。 0: テキストボックスは編集不可で、リストをソートしない 1: テキストボックスは編集可能で、リストをソートしない 2: テキストボックスは編集不可で、リストをソートする 3: テキストボックスは編集可能で、リストをソートする

Begin Dialog、Dim As UserDialogも参照してください。

例

```
Sub Main
Dim lists$(3)
lists$(0) = "List 0"
lists$(1) = "List 1"
lists$(2) = "List 2"
lists$(3) = "List 3"
Begin Dialog UserDialog 200,120
Text 10,10,180,15,"Please push the OK button"
DropListBox 10,25,180,60,lists$(),.list1
DropListBox 10,50,180,60,lists$(),.list2,1
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
dlg.list1 = 2 ' list1 is a numeric field
dlg.list2 = "xxx" ' list2 is a string field
Dialog dlg ' show dialog (wait for ok)
Debug.Print lists$(dlg.list1)
Debug.Print dlg.list2
End Sub
```

GetFilePath$ Function

ダイアログボックスを開き、ユーザからファイルパスを取得します。戻り値はフルパスおよびファイル名の文字列です。キャンセルボタンが押されると、null文字列を返します。

```
GetFilePath[$]([DefName$], [DefExt$], [DefDir$], _
[Title$], [Option])
```

変数	説明
DefName$	ファイル名の初期値文字列を設定。省略した場合、*.DefExt$を使用します。
DefExt$	拡張子がこの文字列値に一致するファイルを表示(拡張子が複数の場合、";"をセパレータとして使用)。省略した場合、"*"を使用します。
DefDir$	初期ディレクトリ文字列。省略した場合、現在のディレクトリを使用します。
Title$	ダイアログのタイトル文字列。省略した場合"開く"を使用します。
Option	ファイル選択オプションを決定。以下から一つを選択します(省略した場合、0を使用します)。 0: ユーザは、存在するファイルのみを選択可能。 1: ユーザが存在しないファイルを選択した場合、作成するかを確認。 2: ユーザは、存在するかどうかに関わらず、任意のファイルを選択可能。 3: ユーザが存在するファイルを選択した場合、上書きを確認。 +4: 別ディレクトリを選択した場合、アプリケーションの現在のディレクトリを変更。

例

```
Sub Main
Debug.Print GetFilePath$()
End Sub
```

GroupBox Dialog Item Definition

グループボックスの項目を定義します。

```
GroupBox X, Y, DX, DY, Title$[, .Field]
```

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
Title$	グループボックスのタイトル文字列。
Field	この識別子は、フィールド名です。dialogfuncはこの名前を文字列として受け取ります。省略した場合、タイトルの最初の2単語を使用。

Begin Dialog、Dim As UserDialogも参照してください。

例

```
Sub Main
Begin Dialog UserDialog 200,120
Text 10,10,180,15,"Please push the OK button"
GroupBox 10,25,180,60,"Group box"
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
Dialog dlg ' show dialog (wait for ok)
End Sub
```

InputBox$ Function

テキスト行が入力可能な入力ボックスを表示。OKボタンを押すと、入力した文字列を返します。キャンセルボタンを押すと、null文字列を返します。

```
InputBox[$](Prompt$[, Title$][, Default$][, XPos, YPos])
```

変数	説明
Prompt$	文字列値は入力ボックスのプロンプトとして使用。
Title$	文字列値は入力ボックスのタイトルとして使用。省略した場合、入力ボックスにタイトルが付きません。
Default$	文字列値は入力ボックスの初期値として使用。省略した場合、初期値は空白です。
XPos	ダイアログが表示される、画面の左端の位置。省略した場合、ダイアログは中央に配置します。
YPos	ダイアログが表示される、画面の上端の位置。省略した場合、ダイアログは中央に配置します。

例

```
Sub Main
L$ = InputBox$("Enter some text:", _
"Input Box Example","asdf")
Debug.Print L$
End Sub
```

ListBox Dialog Item Definition

リストボックスの項目を定義します。

```
ListBox X, Y, DX, DY, StrArray$( ), .Field[, Options]
```

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
StrArray$( )	選択肢のリストを決定する一次元配列の文字列。配列のすべての要素にnull以外を使用します。
Field$	リストボックスの値には、このフィールドでアクセスします。これはStrArray$( )のインデックスです。
Options	この数値は、リストボックスのタイプを制御します。以下から一つを選択します(省略した場合、0を使用します)。 0: リストはソートしない 1: リストはソートせず、水平スクロールバーが有効 2: リストをソートする 3: リストをソートをソートし、水平スクロールバーが有効

Begin Dialog、Dim As UserDialogおよびMultiListBoxも参照してください。

例

```
Sub Main
Dim lists$(3)
lists$(0) = "List 0"
lists$(1) = "List 1"
lists$(2) = "List 2"
lists$(3) = "List 3"
Begin Dialog UserDialog 200,120
Text 10,10,180,15,"Please push the OK button"
ListBox 10,25,180,60,lists$(),.list
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
dlg.list1 = 2
Dialog dlg ' show dialog (wait for ok)
Debug.Print dlg.list
End Sub
```

MsgBox Instruction/Function

タイトルがTitle$のメッセージボックスを表示します。

```
MsgBox Message$[, Type][, Title$]
```

または

```
MsgBox(Message$[, Type][, Title$])
```

Typeは、メッセージボックスの外観を制御します(各カテゴリから一つの値を選択します)。どのボタンが押されたのかを知る必要がある場合はMsgBox()を使います。結果は、どのボタンが押されたかを示します。

結果	値	ボタン
bOK	1	OKボタン
bCancel	2	キャンセルボタン
bAbort	3	中止ボタン
bRetry	4	再試行ボタン
bIgnore	5	無視ボタン
bYes	6	はいボタン
bNo	7	いいえボタン

変数	説明
Message$	文字列値はメッセージボックスに表示するテキスト。
Type	メッセージボックスの種類。下表から一つを選択します(省略した場合、0を使用します)。
Title$	メッセージボックスのタイトル文字列。

ボタン	値	内容
bOkOnly	0	OKボタンボタンのみ表示。
bOkCancel	1	OKとキャンセルボタンを表示。
bAbortRetryIgnore	2	中止、再試行および無視ボタンを表示。
bYesNoCancel	3	はい、いいえ、およびキャンセルボタンを表示。
bYesNo	4	はいおよび、いいえボタンを表示。
bRetryCancel	5	再試行およびキャンセルボタンを表示。

アイコン	値	内容
	0	アイコン無し。
bCritical	16	警告メッセージアイコンを表示。
vbQuestion	32	問い合わせメッセージアイコンを表示。
vbExclamation	48	注意メッセージアイコンを表示 。
vbInformation	64	情報メッセージアイコンを表示。

デフォルト	値	内容
bDefaultButton1	0	第一ボタンをデフォルトに設定。
bDefaultButton2	256	第二ボタンをデフォルトに設定。
bDefaultButton3	512	第三ボタンをデフォルトに設定。

モーダル	値	内容
bApplicationModal	0	アプリケーションモーダルに設定。
bSystemModal	4096	システムモーダルに設定。
bMsgBoxSetForeground	65536	システムモーダルに設定。最前面のウィンドウとしてメッセージボックス ウィンドウ

例

```
Sub Main
MsgBox "Please press OK button"
If MsgBox("Please press OK button",vbOkCancel) = vbOK Then
Debug.Print "OK was pressed"
Else
Debug.Print "Cancel was pressed"
End If
End Sub
```

MultiListBox Dialog Item Definition

複数項目を選択するリストボックスを定義します。

```
MultiListBox X, Y, DX, DY, StrArray$( ), .Field[, Options]
```

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
StrArray$()	選択肢のリストを決定する一次元配列の文字列。配列のすべての要素にnull以外を使用します。
Field$	リストボックスの値には、このフィールドでアクセスします。これはStrArray$( )のインデックスです。
Options	この数値は、リストボックスのタイプを制御します。以下から一つを選択します(省略した場合、0を使用します)。 0: リストはソートしない 1: リストはソートせず、水平スクロールバーが有効 2: リストをソートする 3: リストをソートをソートし、水平スクロールバーが有効

Begin Dialog、Dim As UserDialogおよびListBoxも参照してください。

例

```
Sub Main
Dim lists$(3)
lists$(0) = "List 0"
lists$(1) = "List 1"
lists$(2) = "List 2"
lists$(3) = "List 3"
Begin Dialog UserDialog 200,120
Text 10,10,180,15,"Please push the OK button"
MultiListBox 10,25,180,60,lists$(),.list
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
dlg.list1 = Array(2)
Dialog dlg ' show dialog (wait for ok)
For i = LBound(dlg.list) To UBound(dlg.list)
Debug.Print dlg.list(i);
Next i
Debug.Print
End Sub
```

OKButton Dialog Item Definition

OKボタンの項目を定義します。

```
OKButton X, Y, DX, DY[, .Field]
```

OKボタンを押した場合、dlgvarフィールド値が更新され、ダイアログを閉じます(Dialog ( ) 関数の呼び出しは-1を返します)。

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
Field	この識別子は、フィールド名です。dialogfuncはこの名前を文字列として受け取ります。省略した場合、フィールド名は"OK"になります。

Begin Dialog、Dim As UserDialogも参照してください。

例

```
Sub Main
Begin Dialog UserDialog 200,120
Text 10,10,180,30,"Please push the OK button"
GroupBox 10,25,180,60,"Group box"
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
Dialog dlg ' show dialog (wait for ok)
End Sub
```

OptionButton Dialog Item Definition

オプションボタンの項目を定義します。

```
OptionButton X, Y, DX, DY, Title$[, .Field]
```

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
Title$	オプションボタンのタイトル文字列。

Begin Dialog、Dim As UserDialogおよびOptionGroupも参照してください。

例

```
Sub Main
Begin Dialog UserDialog 200,120
Text 10,10,180,15,"Please push the OK button"
OptionGroup .options
OptionButton 10,30,180,15,"Option &0"
OptionButton 10,45,180,15,"Option &1"
OptionButton 10,60,180,15,"Option &2"
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
dlg.options = 2
Dialog dlg ' show dialog (wait for ok)
Debug.Print dlg.options
End Sub
```

OptionGroup Dialog Item Definition

オプショングループおよびオプションボタンの項目を定義します。

```
OptionGroup .Field
OptionButton X, Y, DX, DY, Title$[, .Field]
OptionButton X, Y, DX, DY, Title$[, .Field]
...
```

変数	説明
Field	オプショングループの値には、このフィールドでアクセスします。最初のオプションボタンは0、2番目は1、などとなります。
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
Title$	オプションボタンのタイトル文字列。

Begin Dialog、Dim As UserDialogおよびOptionButtonも参照してください。

例

```
Sub Main
Begin Dialog UserDialog 200,120
Text 10,10,180,15,"Please push the OK button"
OptionGroup .options
OptionButton 10,30,180,15,"Option &0"
OptionButton 10,45,180,15,"Option &1"
OptionButton 10,60,180,15,"Option &2"
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
dlg.options = 2
Dialog dlg ' show dialog (wait for ok)
Debug.Print dlg.options
End Sub
```

Picture Dialog Item Definition

画像アイテムを定義します。ビットマップのサイズは、項目の全領域に合わせて自動的に調整されます。

```
Picture X, Y, DX, DY, FileName$, Type[, .Field]
```

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
FileName$	画像コントロールに表示される.BMPファイル。
Type	使用するビットマップのタイプ: 0: FileName$はビットマップファイルの名前。ファイルが存在しない場合"missing picture"を表示します。 3: クリップボードのビットマップを表示。サポートしていません。 +16: "missing picture"を表示せず、ランタイムエラーを発生。
Field	この識別子は、フィールド名です。dialogfuncはこの名前を文字列として受け取ります。省略した場合、タイトルの最初の2単語を使用します。

Begin Dialog、Dim As UserDialogも参照してください。

例

```
Sub Main
Begin Dialog UserDialog 200,120
Picture 10,10,180,75,"SAMPLE.BMP",0
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
Dialog dlg ' show dialog (wait for ok)
End Sub
```

PushButton Dialog Item Definition

押しボタンの項目を定義します。

```
PushButton X, Y, DX, DY, Title$[, .Field]
```

押しボタンを押した場合、dlgvarフィールド値が更新され、ダイアログを閉じます(Dialog ( ) 関数の呼び出しは、ダイアログ内の押しボタンの順序数を返します。最初の押しボタンの戻り値は1です)。最初の押しボタンは-1を返します)。

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
Title$	押しボタンのタイトル文字列。
Field	この識別子は、フィールド名です。dialogfuncはこの名前を文字列として受け取ります。省略した場合、タイトルの最初の2単語を使用します。

Begin Dialog、Dim As UserDialogも参照してください。

例

```
Sub Main
Begin Dialog UserDialog 200,120
Text 10,10,180,30,"Please push the OK button"
OKButton 40,90,40,20
PushButton 110,90,60,20,"&Do It"
End Dialog
Dim dlg As UserDialog
Debug.Print Dialog(dlg)
End Sub
```

ShowPopupMenu Function

ポップアップメニューを表示し、選択した項目の番号を返します。

```
ShowPopupMenu(StrArray$( )[, PopupStyle][, XPos, YPos])
```

項目番号は、選択されたStrArrayのインデックスからLBound(StrArray)を引いた値です。メニュー項目を選択していない場合、-1を返します。

変数	説明
StrArray$( )	選択肢のリストを決定する一次元配列の文字列。配列のすべての要素にnull以外を使用します。
PopupMenuStyle	ポップアップメニューの配置方法を決定。任意のスタイルの組み合わせを使用することができます。下表を参照してください。
XPos	ダイアログが表示される画面の左端の位置。省略した場合、現在のマウス位置を使用します。
YPos	ダイアログが表示される画面の上端の位置。省略した場合、現在のマウス位置を使用します。

PopupMenuStyle	値	内容
bPopupLeftTopAlign	0	メニューをXPosの左端、YPosの上端に整列(デフォルト)。
bPopupUseLeftButton	1	ユーザはマウスの左ボタンでのみメニューを選択可能。
bPopupUseRightButton	2	ユーザはマウスの左または右ボタンでメニューを選択可能。
bPopupRightAlign	4	メニューをXPosの右端に整列。
bPopupCenterAlign	8	メニューをXPosの中心に整列。
bPopupVCenterAlign	16	メニューをYPosの中心に整列。
bPopupBottomAlign	32	メニューをYPosの下端に整列。

例

```
Sub Main
Dim Items(0 To 2) As String
Items(0) = "Item &1"
Items(1) = "Item &2"
Items(2) = "Item &3"
X = ShowPopupMenu(Items) ' show popup menu
Debug.Print X ' item selected
End Sub
```

Text Dialog Item Definition

テキストの項目を定義します。

```
Text X, Y, DX, DY, Title$[, .Field][, Options]
```

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
Title$	テキスト制御器のタイトル文字列。
Field	この識別子は、フィールド名です。dialogfuncはこの名前を文字列として受け取ります。省略した場合、タイトルの最初の2単語を使用します。
Options	この数値は、テキストの配置を制御します。以下から一つを選択します(省略した場合、0を使用します)。 0: 左寄せ 1: 右寄せ 2: 中央寄せ

Begin Dialog、Dim As UserDialogも参照してください。

例

```
Sub Main
Begin Dialog UserDialog 200,120
Text 10,10,180,15,"Please push the OK button"
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
Dialog dlg ' show dialog (wait for ok)
End Sub
```

TextBox Dialog Item Definition

テキストボックスの項目を定義します。

```
TextBox X, Y, DX, DY, .Field$[, Options]
```

変数	説明
X	画面の左端からダイアログボックスの左端までの距離の数値。ダイアログフォントの平均文字幅の1/8で測定されます。省略した場合、ダイアログは中央に配置されます。
Y	画面の上端からダイアログボックスの上端までの距離の数値。ダイアログフォントの平均文字幅の1/12で測定されます。省略した場合、ダイアログは中央に配置されます。
DX	幅の数値。ダイアログフォントの平均文字幅の1/8で測定されます。
DY	高さの数値。ダイアログフォントの文字高さの1/12で測定されます。
Field$	テキストボックスの値には、このフィールドでアクセスします。
Options	この数値はテキストボックスのタイプを制御します。以下から一つを選択します(省略した場合、0を使用します)。 0: 一行のテキストのみ入力可能 1: 複数行のテキストを入力可能 -1: テキストはパスワード表示

Begin Dialog、Dim As UserDialogも参照してください。

例

```
Sub Main
Begin Dialog UserDialog 200,120
Text 10,10,180,15,"Please push the OK button"
TextBox 10,25,180,20,.Text$
OKButton 80,90,40,20
End Dialog
Dim dlg As UserDialog
dlg.Text$ = "none"
Dialog dlg ' show dialog (wait for ok)
Debug.Print dlg.Text$
End Sub
```

UserDialog Data Type

Begin Dialog UserDialogで定義されたユーザタイプ。

関数および命令

Abs Function ・・・

DlgListBoxArray Instruction/Function ・・・

KeyName Function ・・・

Sqr Function ・・・

Abs Function

絶対値を返します。

```
Abs(Num)
```

変数	説明
Num	数値の絶対値を返します。値がNullの場合、Nullを返します。

Sgnも参照してください。

例

```
Sub Main
Debug.Print Abs(9) ' 9
Debug.Print Abs(0) ' 0
Debug.Print Abs(-9) ' 9
End Sub
```

AppActivate Instruction

Form 1:
Title$で指定したアプリケーションのトップレベルウィンドウをアクティブにします。指定したウィンドウが存在しない場合、Title$で始まるタイトルの最初のウィンドウがアクティブになります。一致するウィンドウがない場合、エラーが発生します。

```
AppActivate Title$
```

Form 2
タスクのTaskIDのアプリケーションのトップレベルウィンドウをアクティブにします。タスクのウィンドウが存在しない場合、エラーが発生します。

```
AppActivate TaskID
```

変数	説明
Title$	アクティブ化するアプリケーション ウィンドウのタイトル バーに表示されるタイトルを指定する文字列。TaskIDは数値です。

SendKeys、Shell( )も参照してください。

例

```
Sub Main
' make ProgMan the active application
AppActivate "Program Manager"
End Sub
```

Array Function

式の要素(expr)を含むバリアント値の配列を返します。

```
Array([expr[, ...]])
```

例

```
Sub Main
X = Array(0,1,4,9)
Debug.Print X(2) ' 4
End Sub
```

Asc Function

文字列内の最初の文字のASCII値を返します。

```
Asc(S$)
```

同様のAscB関数は、S$の最初のバイトを返します。またAscW関数は、Unicode値を返します。

変数	説明
S$	任意の文字列を指定します。

Chr$()も参照してください。

例

```
Sub Main
Debug.Print Asc("A") ' 65
End Sub
```

AscB Function

文字列内の最初の文字のバイトコードを返します。

```
AscB(S$)
```

同様のAsc関数は、S$のASCIIコードを返します。またAscW関数は、Unicodeを返します。

変数	説明
S$	任意の文字列を指定します。

Chr$()も参照してください。

例

```
Sub Main
Debug.Print AscB("A") ' 65
End Sub
```

AscW Function

文字列内の最初の文字のUnicodeを返します。

```
AscW(S$)
```

同様のAsc関数は、S$のASCIIコードを返します。またAscB関数は、バイトコードを返します。

変数	説明
S$	任意の文字列を指定します。

Chr$()も参照してください。

例

```
Sub Main
Debug.Print AscW("A") ' 65
End Sub
```

Atn Function

アークタンジェントを返します。これはラジアン値です。円周は2π radです。

```
Atn(Num)
```

変数	説明
Num	指定した角度のアークタンジェントを返します。

Cos、Sin、Tanも参照してください。

例

```
Sub Main
Debug.Print Atn(1)*4 ' 3.1415926535898
End Sub
```

Beep Instruction

ビープ音を鳴らします。

```
Beep
```

例

```
Sub Main
.Beep' beep the bell
End Sub
```

Call Instruction

arglistを評価し、nameサブルーチン(または関数)を呼び出します。Sub(または関数)nameは、Sub、FunctionまたはPropertyのいずれかによって事前に定義されていなければなりません。nameが関数の場合、結果は破棄されます。*Call*を省略し、nameがサブルーチンでの場合、arglistは括弧で囲みません。

```
  Call name[(arglist)]
```
または
```
  name [arglist]
```

Declare、Subも参照してください。

例

```
  Sub Show(Title$,Value)
  Debug.Print Title$;"=";Value
  End Sub

  Sub Main
  Call Show("2000/9",2000/9) ' 222.2222222222
  Show "1<2",1<2 'True
  End Sub
```

CallByName Instruction

Objのメソッド/プロパティであるProcNameを呼び出します。exprをメソッド/プロパティに渡します。

```
CallByName(Obj,ProcName,CallType,[expr[, ...]])
```

変数	説明
Obj	メソッド/プロパティを呼び出すオブジェクトの参照ポインタ。
ProcName	呼び出すメソッド/プロパティの名前の文字列。
CallType	呼び出すメソッド/プロパティの種類。下表を参照してください。
expr	objのメソッド/プロパティに渡す式。

呼び出し型	値	説明
bMethod	1	メソッドの呼び出しまたは評価。
bGet	2	プロパティ値を評価。
bLet	4	プロパティ値を割り当て。
bSet	8	プロパティの参照を設定。

例

```
Sub Main
On Error Resume Next
CallByName Err, "Raise", vbMethod, 1
Debug.Print CallByName(Err, "Number", vbGet) ' 1
End Sub
```

CallersLine Function

呼び出し元の行をテキスト文字列として返します。
テキスト形式は"[modulename|subname#linenum] linetext"です。

CallersLine[(Depth)]

変数	説明
Depth	呼び出し元の行を取得するスタックの深さを示す整数値。*Depth* = -1の場合、現在の行を返します。*Depth* = 0の場合、呼び出したサブルーチンの現在の行などを返します。*Depth*が呼び出しスタックの深さ以上の場合、null文字列を返します。この値を省略した場合、深さは0です。

例

```
Sub Main
A
End Sub
Sub A
Debug.Print CallersLine '"[(untitled 1)|Main# 2] A"
End Sub
```

CBool Function

ブール値に変換します。ゼロはFalseに変換し、他はすべてTrueに変換します。

```
CBool(Num|$)
```

引数	説明
Num|$	数値または文字列をブール値に変換します。

例

```
Sub Main
Debug.Print CBool(-1) 'True
Debug.Print CBool(0) 'False
Debug.Print CBool(1) 'True
End Sub
```

CByte Function

Byte((8ビット符号なし整数)に変換します。

```
CByte(Num|$)
```

引数	説明
Num|$	数値または文字列をバイト値に変換します。

例

```
Sub Main
Debug.Print CByte(1.6) ' 2
End Sub
```

CCur Function

CCurの数値に変換します。

```
CCur(Num|$)
```

引数	説明
Num|$	数値または文字列を通貨に変換します。

例

```
Sub Main
Debug.Print CCur("1E6") ' 1000000
End Sub
```

CDate Function, CVDate Function

日付に変換します。

```
CDate(Num|$)
```

または

```
CVDate(Num|$)
```

引数	説明
Num|$	数値または文字列を日付値に変換します。

例

```
Sub Main
Debug.Print CDate(2) ' 1/1/00
End Sub
```

CDbl Function

Double(倍精度浮動小数点)に変換します。

```
CDbl(Num|$)
```

引数	説明
Num|$	数値または文字列を倍精度浮動小数点に変換します。

例

```
Sub Main
Debug.Print CDbl(1E6) ' 1000000
End Sub
```

CDec Function

Win32のみ。10進数に変換(実数を96bitにスケーリング)します。

```
CDec (Num|$)
```

引数	説明
Num|$	数値または文字列の実数を96 bitスケールに変換します。

例

```
Sub Main
Debug.Print CDec("1E16")+0.1 ' 10000000000000000.1
End Sub
```

ChDir Instruction

現在のディレクトリをDir$に変更します。

```
ChDir Dir$
```

引数	説明
Dir$	ディレクトリのパスおよび名前の文字列。

ChDrive、CurDir$()も参照してください。

例

```
Sub Main
ChDir "C:\"
Debug.Print CurDir$() '"C:\"
End Sub
```

ChDrive Instruction

現在のドライブをDrive$に変更します。

```
ChDrive Drive$
```

引数	説明
Drive$	ドライブ名の文字列。

ChDir、CurDir$()も参照してください。

例

```
Sub Main
ChDrive "B"
Debug.Print CurDir$() '"B:\"
End Sub
```

Choose Function

リストからIndexで指定したexprの値を返します。

```
Choose(Index, expr[, ...])
```

引数	説明
Index	数値は、どのexprを返すかを示します。値が1未満またはexprの数より多い場合、Nullを返します。
expr	すべての式が評価されます。

If、Select Case、IIf()も参照してください。

例

```
Sub Main
Debug.Print Choose(2,"Hi","there") '"there"
End Sub
```

Chr$ Function

指定したASCIIコードに対応する1文字を返します。

```
Chr[$](Num)
```

同様のChrB関数は、1バイトのASCII文字列を返します。またChrW関数は、Unicode文字列を返します。

引数	説明
Num	ASCIIの文字コードを指定。

Asc()も参照してください。

例

```
Sub Main
Debug.Print Chr$(48) '"0"
End Sub
```

ChrB$ Function

指定した1バイトのASCII文字列を返します。

```
ChrB[$](Num)
```

同様のChr関数は、ASCIIコードに対応する1文字を返します。またChrW関数は、Unicode文字列を返します。

引数	説明
Num	ASCIIの文字コードを指定。

Asc()も参照してください。

例

```
Sub Main
Debug.Print ChrB$(48) '"0"
End Sub
```

ChrW$ Function

指定した1文字のUnicode文字列を返します。

```
ChrW[$](Num)
```

同様のChr関数は、ASCIIコードに対応する1文字を返します。またChrW関数は、Unicode文字列を返します。

引数	説明
Num	Unicodeを指定。

Asc()も参照してください。

例

```
Sub Main
Debug.Print ChrW$(48) '"0"
End Sub
```

CInt Function

Integer(16bit整数)に変換します。Num|$が大きすぎる(または小さすぎる)場合、オーバーフローエラーが発生します。

```
CDbl(Num|$)
```

引数	説明
Num|$	数値または文字列を16bit整数に変換します。

例

```
Sub Main
Debug.Print CInt(1.6) ' 2
End Sub
```

Clipboard Instruction/Function

Form 1:
クリップボードをText$に設定します。これは、メニューの"編集"→"コピー"と似ています。

```
Clipboard Text$
```

Form 2:
クリップボード内のテキストを返します。

```
Clipboard[$][( )]
```

引数	説明
Text$	文字列値をクリップボードに格納します。

例

```
Sub Main
Debug.Print Clipboard$()
Clipboard "Hello"
Debug.Print Clipboard$() '"Hello"
End Sub
```

CLng Function

Long(32bit長整数)に変換します。Num | $が大きすぎる(または小さすぎる)場合、オーバーフローエラーが発生します。

```
CLng(Num|$)
```

引数	説明
Num|$	数値または文字列を32bit長整数に変換します。

例

```
Sub Main
Debug.Print CLng(1.6) ' 2
End Sub
```

Close Instruction

StreamNumを閉じます。

```
Close [[#]StreamNum][, ...]
```

引数	説明
StreamNum	1から255は、各モジュールに対してプライベートです。256から511はすべてのモジュールで共有されます。省略した場合、現在のモジュールで開いているすべてのストリームを閉じます。

Open、Resetも参照してください。

例

```
Sub Main
' read the first line of XXX and print it
Open "XXX" For Input As #1
Line Input #1,L$
Debug.Print L$
Close #1
End Sub
```

Cos Function

コサインを返します。

```
Cos(Num)
```

引数	説明
Num	求めるコサインの数値。これはラジアン値です。円周は2π radです。

Atn、Sin、Tanも参照してください。

例

```
Sub Main
Debug.Print Cos(1) ' 0.54030230586814
End Sub
```

CSng Function

Single(単精度浮動小数点)に変換します。Num | $が大きすぎる(または小さすぎる)場合、オーバーフローエラーが発生します。

```
CLng(Num|$)
```

引数	説明
Num|$	数値または文字列を単精度浮動小数点に変換します。

例

```
Sub Main
Debug.Print CSng(Sqr(2)) ' 1.4142135381699
End Sub
```

CStr Function

文字列に変換します。

```
CStr(Num|$)
```

引数	説明
Num|$	数値を文字列に変換します。

例

```
Sub Main
Debug.Print CStr(Sqr(2)) ' "1.4142135623731"
End Sub
```

CurDir$ Function

Drive$の現在のディレクトリを返します。

```
CurDir[$]([Drive$])
```

引数	説明
Drive$	ドライブ名の文字列。省略またはNullの場合、現在のドライブの現在のディレクトリを返します。

ChDir、ChDriveも参照してください。

例

```
Sub Main
Debug.Print CurDir$()
End Sub
```

CVar Function

Variant(バリアント)値に変換します。

```
CVar(Num|$)
```

引数	説明
Num|$	数値または文字列(またはオブジェクトの参照)をバリアントに変換します。

例

```
Sub Main
Debug.Print CVar(Sqr(2)) ' 1.4142135623731
End Sub
```

CVErr Function

エラーコードが含まれるバリアントに変換します。エラーコードは式で使用することはできません。

```
CVErr(Num|$)
```

引数	説明
Num|$	数値または文字列をエラーコードに変換します。

IsErrorも参照してください。

例

```
Sub Main
Debug.Print CVErr(1) ' Error 1
End Sub
```

Date Function

今日の日付をDate型の値として返します。

```
Date[$]
```

Now、Time、Timerも参照してください。

例

```
Sub Main
Debug.Print Date ' example: 1/1/1995
End Sub
```

DateAdd Function

指定した日付に、時間間隔を加算した日付を返します。

```
DateAdd(interval, number, dateexpr)
```

引数	説明
interval	追加する時間間隔の単位の文字列。下表を参照してください。
number	加算する間隔。過去の日時を取得するには負の数を使用します。
dateexpr	時間間隔を追加する基準となる日付。値がNullの場合、Nullを返します。

Interval	説明(戻り値の範囲)
yyyy	年(100-9999)
q	四半期(1-4)
m	月(1-12)
y	年間通算日(1-366)
d	日(1-31)
w	週日(1-7)
ww	週(1-53)
h	時(0-23)
n	分(0-59)
s	秒(0-59)

DateDiff、DatePartも参照してください。

例

```
Sub Main
Debug.Print DateAdd("yyyy",1,#1/1/2000#) ' 1/1/2001
End Sub
```

DateDiff Function

2つの日付の時間間隔を返します。

```
DateDiff(interval, dateexpr1, dateexpr2)
```

引数	説明
interval	減算する時間間隔の単位の文字列。下表を参照してください。
dateexpr1	この日付値からdateexpr2までを計算します。値がNullの場合、Nullを返します。
dateexpr2	dateexpr1からこの日付値までを計算します。値がNullの場合、Nullを返します。

Interval	説明(戻り値の範囲)
yyyy	年(100-9999)
q	四半期(1-4)
m	月(1-12)
y	年間通算日(1-366)
d	日(1-31)
w	週日(1-7)
ww	週(1-53)
h	時(0-23)
n	分(0-59)
s	秒(0-59)

DateAdd、DatePartも参照してください。

例

```
Sub Main
Debug.Print DateDiff("yyyy",#1/1/1990#,#1/1/2000#) ' 10
End Sub
```

DatePart Function

指定した日付の一部分を返します。

```
DatePart(interval, dateexpr)
```

引数	説明
interval	時間間隔の単位の文字列。下表を参照してください。
dateexpr	評価する日付。値がNullの場合、Nullを返します。

Interval	説明(戻り値の範囲)
yyyy	年(100-9999)
q	四半期(1-4)
m	月(1-12)
y	年間通算日(1-366)
d	日(1-31)
w	週日(1-7)
ww	週(1-53)
h	時(0-23)
n	分(0-59)
s	秒(0-59)

DateAdd、DateDiffも参照してください。

例

```
Sub Main
Debug.Print DatePart("yyyy",#1/1/2000#) ' 2000
End Sub
```

DateSerial Function

日付値を返します。

```
DateSerial(Year, Month, Day)
```

引数	説明
Year	年を表す数値(0～9999。0から99はOSにより解釈は依存。)
Month	月を表す数値(1～12)。
Day	日を表す数値(1～31)。

DateValue、TimeSerial、TimeValueも参照してください。

例

```
Sub Main
Debug.Print DateSerial(2000,7,4) ' 7/4/2000
End Sub
```

DateValue Function

文字列としてエンコードした日付の部分を返します。

```
DateValue(Date$)
```

引数	説明
Date$	日付部分に変換する文字列。

DateSerial、TimeSerial、TimeValueも参照してください。

例

```
Sub Main
Debug.Print DateValue("1/1/2000 12:00:01 AM") ' 1/1/2000
End Sub
```

Day Function

日付を表す整数(1～ 31)を返します。

```
Day(dateexpr)
```

引数	説明
dateexpr	日付を表す式。値がNullの場合、Nullを返します。

Date()、Month()、Weekday()、Year()も参照してください。

例

```
Sub Main
Debug.Print Day(#1/1/1900#) ' 1
Debug.Print Day(#1/2/1900#) ' 2
End Sub
```

DeleteSetting Instruction

AppNameプロジェクトのセクションのキー設定を削除します。Win16およびWin32は、*AppName*という名前の.iniファイルに設定を保存します。Win32は設定を登録データベースに保存します。

```
DeleteSetting AppName$, Section$[, Key$]
```

引数	説明
AppName$	セクションまたはキー設定を適用するアプリケーション名、またはプロジェクト名を示す文字列式。
Section$	プロジェクト設定のセクション名を示す文字列。
Key$	削除するプロジェクト設定のキーを示す文字列。省略した場合、セクション全体を削除します。

例

```
Sub Main
SaveSetting "MyApp","Font","Size",10
DeleteSetting "MyApp","Font","Size"
End Sub
```

Dir$ Function

ディレクトリをスキャンして、Pattern$に一致する最初のファイルを検索します。

```
Dir[$]([Pattern$][, AttribMask])
```

引数	説明
Pattern$	検索するファイルパターンのパスおよび名前。省略した場合、前のパターンでスキャンを継続します。現在のディレクトリからの相対パスを使用することができます。
AttribMask	この数値は、検出するファイルを制御します。一致するattributeを持つファイルを検索します。

GetAttr()も参照してください。

例

```
Sub Main
F$ = Dir$("*.*")
While F$ <> ""
Debug.Print F$
F$ = Dir$()
Wend
End Sub
```

DlgControlId Function

フィールドのWindow IDを返します。
この命令/関数はDialogFuncから直接、または間接的に呼び出さなければなりません。

```
DlgControlId(DlgItem|$)
```

引数	説明
DlgItem|$	値が数値の場合、ダイアログの項目番号です。最初の項目が0で、二番目が1…です。値が文字列の場合、ダイアログ項目のフィールド名です。

例

```
Sub Main
Begin Dialog UserDialog 200,120,.DialogFunc
Text 10,10,180,15,"Please push the OK button"
TextBox 10,40,180,15,.Text
OKButton 30,90,60,20
PushButton 110,90,60,20,"&Hello"
End Dialog
Dim dlg As UserDialog
Debug.Print Dialog(dlg)
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Debug.Print "Action=";Action%
Select Case Action%
Case 1 ' Dialog box initialization
Beep
Case 2 ' Value changing or button pressed
If DlgItem$ = "Hello" Then
DialogFunc% = True 'do not exit the dialog
End If
Case 4 ' Focus changed
Debug.Print "DlgFocus=""";DlgFocus();"""
Debug.Print "DlgControlId(";DlgItem$;")=";
Debug.Print DlgControlId(DlgItem$)
End Select
End Function
```

DlgCount Function

ダイアログ内の項目数を返します。
この命令/関数はDialogFuncから直接、または間接的に呼び出さなければなりません。

```
DlgCount()
```

例

```
Sub Main
Begin Dialog UserDialog 200,120,.DialogFunc
Text 10,10,180,15,"Please push the OK button"
TextBox 10,40,180,15,.Text
OKButton 30,90,60,20
End Dialog
Dim dlg As UserDialog
Dialog dlg
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Debug.Print "Action=";Action%
Select Case Action%
Case 1 ' Dialog box initialization
Beep
Debug.Print "DlgCount=";DlgCount() ' 3
End Select
End Function
```

DlgEnable Instruction/Function

DlgItem|$を有効または無効にします。

```
DlgEnable DlgItem|$[, Enable]
```

DlgItem|$が有効の場合、Trueを返します。

```
DlgEnable(DlgItem|$)
```

この命令/関数はDialogFuncから直接、または間接的に呼び出さなければなりません。

引数	説明
DlgItem|$	値が数値の場合、ダイアログの項目番号です。最初の項目が0で、二番目が1...です。値が文字列の場合、ダイアログ項目のフィールド名です。 -1を使用すると、すべてのダイアログ項目を有効/無効にすることができます。
Enable	値がTrueの場合、*DlgItem/$*は有効。そうでない場合は無効。省略した場合は切り替えます。

例

```
Sub Main
Begin Dialog UserDialog 200,120,.DialogFunc
Text 10,10,180,15,"Please push the OK button"
TextBox 10,40,180,15,.Text
OKButton 30,90,60,20
PushButton 110,90,60,20,"&Disable"
End Dialog
Dim dlg As UserDialog
Debug.Print Dialog(dlg)
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Debug.Print "Action=";Action%
Select Case Action%
Case 1 ' Dialog box initialization
Beep
Case 2 ' Value changing or button pressed
Select Case DlgItem$
Case "Disable"
DlgText DlgItem$,"&Enable"
DlgEnable "Text",False
DialogFunc% = True 'do not exit the dialog
Case "Enable"
DlgText DlgItem$,"&Disable"
DlgEnable "Text",True
DialogFunc% = True 'do not exit the dialog
End Select
End Select
End Function
```

DlgEnd Instruction

Dialog関数の戻りコードを設定し、ユーザダイアログを閉じます。
この命令/関数はDialogFuncから直接、または間接的に呼び出さなければなりません。

```
DlgEnd ReturnCode
```

引数	説明
ReturnCode	戻りコード。

例

```
Sub Main
Begin Dialog UserDialog 210,120,.DialogFunc
Text 10,10,190,15,"Please push the Close button"
OKButton 30,90,60,20
CheckBox 120,90,60,20,"&Close",.CheckBox1
End Dialog
Dim dlg As UserDialog
Debug.Print Dialog(dlg)
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Debug.Print "Action=";Action%
Select Case Action%
Case 1 ' Dialog box initialization
Beep
Case 2 ' Value changing or button pressed
Select Case DlgItem$
Case "CheckBox1"
DlgEnd 1000
End Select
End Select
End Function
```

DlgFocus Instruction/Function

DlgItem|$にフォーカスを移動します。

```
DlgFocus DlgItem|$
```

フォーカスされているフィールド名の文字列を返します。

```
DlgFocus[$]()
```

この命令/関数はDialogFuncから直接、または間接的に呼び出さなければなりません。

引数	説明
DlgItem|$	値が数値の場合、ダイアログの項目番号です。最初の項目が0で、二番目が1…です。値が文字列の場合、ダイアログ項目のフィールド名です。

例

```
Sub Main
Begin Dialog UserDialog 200,120,.DialogFunc
Text 10,10,180,15,"Please push the OK button"
TextBox 10,40,180,15,.Text
OKButton 30,90,60,20
PushButton 110,90,60,20,"&Hello"
End Dialog
Dim dlg As UserDialog
Debug.Print Dialog(dlg)
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Debug.Print "Action=";Action%
Select Case Action%
Case 1 ' Dialog box initialization
Beep
Case 2 ' Value changing or button pressed
If DlgItem$ = "Hello" Then
MsgBox "Hello"
DialogFunc% = True 'do not exit the dialog
End If
Case 4 ' Focus changed
Debug.Print "DlgFocus=""";DlgFocus();"""
End Select
End Function
```

DlgListBoxArray Instruction/Function

DlgItem|$のリスト項目を設定します。

```
DlgListBoxArray DlgItem|$, StrArray$( )
```

リスト内のDlgItem|$の項目数を返します。

```
DlgListBoxArray(DlgItem|$[, StrArray$( )])
```

この命令/関数はDialogFuncから直接、または間接的に呼び出さなければなりません。DlgItem|$は、ComboBox、DropListBox、ListBoxまたはMultiListBoxを参照します。

引数	説明
DlgItem|$	値が数値の場合、ダイアログの項目番号です。最初の項目が0で、二番目が1…です。値が文字列の場合、ダイアログ項目のフィールド名です。
StrArray$( )	*DlgItem/$*のリスト項目を設定。選択のリストを定める一次元の文字列配列です。配列のすべての要素はNull以外の文字列を使用します。 *DlgItem/$*のリスト項目を設定します。この一次元の文字列配列により、選択肢のリストが確立されます。配列のnull以外のすべての要素を使用します。

例

```
Dim lists$()
Sub Main
ReDim lists$(0)
Begin Dialog UserDialog 200,119,.DialogFunc
Text 10,7,180,14,"Please push the OK button"
ListBox 10,21,180,63,lists(),.list
OKButton 30,91,40,21
PushButton 110,91,60,21,"&Change"
End Dialog
Dim dlg As UserDialog
dlg.list = 2
Dialog dlg ' show dialog (wait for ok)
Debug.Print dlg.list
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Select Case Action%
Case 2 ' Value changing or button pressed
If DlgItem$ = "Change" Then
Dim N As Integer
N = UBound(lists$)+1
ReDim Preserve lists$(N)
lists$(N) = "List " & N
DlgListBoxArray "list",lists$()
DialogFunc% = True 'do not exit the dialog
End If
End Select
End Function
```

DlgName Function

DlgItem番号のフィールド名を返します。

```
DlgName[$](DlgItem)
```

この命令/関数はDialogFuncから直接、または間接的に呼び出さなければなりません。

引数	説明
DlgItem	この数値はダイアログの項目番号です。最初の項目が0で、二番目が1…です。

例

```
Sub Main
Begin Dialog UserDialog 200,120,.DialogFunc
Text 10,10,180,15,"Please push the OK button"
TextBox 10,40,180,15,.Text
OKButton 30,90,60,20
End Dialog
Dim dlg As UserDialog
Dialog dlg
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Debug.Print "Action=";Action%
Select Case Action%
Case 1 ' Dialog box initialization
For I = 0 To DlgCount()-1
Debug.Print  I;DlgName(I)
Next I
End Select
End Function
```

DlgNumber Function

DlgItem$の番号を返します。最初の項目が0で、二番目が1…です。

```
DlgNumber(DlgItem$)
```

この命令/関数はDialogFuncから直接、または間接的に呼び出さなければなりません。

引数	説明
DlgItem$	ダイアログの項目のフィールド名。

例

```
Sub Main
Begin Dialog UserDialog 200,120,.DialogFunc
Text 10,10,180,15,"Please push the OK button"
TextBox 10,40,180,15,.Text
OKButton 30,90,60,20
End Dialog
Dim dlg As UserDialog
Dialog dlg
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Debug.Print "Action=";Action%
Select Case Action%
Case 1 ' Dialog box initialization
Beep
Case 4 ' Focus changed
Debug.Print DlgItem$;"=";DlgNumber(DlgItem$)
End Select
End Function
```

DlgSetPicture Instruction

DlgItem|$のファイル名を設定します。

```
DlgSetPicture DlgItem|$, FileName, Type
```

この命令/関数はDialogFuncから直接、または間接的に呼び出さなければなりません。

引数	説明
DlgItem|$	値が数値の場合、ダイアログの項目番号です。最初の項目が0で、二番目が1...です。値が文字列の場合、ダイアログ項目のフィールド名です。
FileName	DlgItem|$のファイルを設定する文字列。
Type	使用するビットマップのタイプ: 0: FileName$はビットマップファイルの名前です。ファイルが存在しない場合"missing picture"(画像がありません)を表示します。 3: クリップボードのビットマップを表示。サポートしていません。 +16: "missing picture"を表示せず、ランタイムエラーを発生。

例

```
Sub Main
Begin Dialog UserDialog 200,120,.DialogFunc
Picture 10,10,180,75,"",0,.Picture
OKButton 30,90,60,20
PushButton 110,90,60,20,"&View"
End Dialog
Dim dlg As UserDialog
Debug.Print Dialog(dlg)
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Debug.Print "Action=";Action%
Select Case Action%
Case 1 ' Dialog box initialization
Beep
Case 2 ' Value changing or button pressed
Select Case DlgItem$
Case "View"
FileName = GetFilePath("Bitmap","BMP")
DlgSetPicture "Picture",FileName,0
DialogFunc% = True 'do not exit the dialog
End Select
End Select
End Function
```

DlgText Instruction/Function

DlgItem|$のテキストを設定します。

```
DlgText DlgItem|$, Text
```

DlgItem|$からテキストを返します。

```
DlgText[$](DlgItem|$)
```

この命令/関数はDialogFuncから直接、または間接的に呼び出さなければなりません。

引数	説明
DlgItem|$	値が数値の場合、ダイアログの項目番号です。最初の項目が0で、二番目が1…です。値が文字列の場合、ダイアログ項目のフィールド名です。 -1は、ダイアログのタイトルになります。
Text	DlgItem|$に設定する文字列

例

```
Sub Main
Begin Dialog UserDialog 200,120,.DialogFunc
Text 10,10,180,15,"Please push the OK button"
TextBox 10,40,180,15,.Text
OKButton 30,90,60,20
PushButton 110,90,60,20,"&Now"
End Dialog
Dim dlg As UserDialog
Debug.Print Dialog(dlg)
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Debug.Print "Action=";Action%
Select Case Action%
Case 1 ' Dialog box initialization
Beep
Case 2 ' Value changing or button pressed
Select Case DlgItem$
Case "Now"
DlgText "Text",CStr(Now)
DialogFunc% = True 'do not exit the dialog
End Select
End Select
End Function
```

DlgType Function

DlgItem|$の型を示す文字列を返します。次の何れかになります: "CancelButton"、"CheckBox"、"ComboBox"、"DropListBox"、"GroupBox"、"ListBox"、"MultiListBox"、"OKButton"、"OptionButton"、"OptionGroup"、"PushButton"、"Text"、"TextBox"

```
DlgType[$](DlgItem|$)
```

この命令/関数はDialogFuncから直接、または間接的に呼び出さなければなりません。

引数	説明
DlgItem|$	値が数値の場合、ダイアログの項目番号です。最初の項目が0で、二番目が1...です。値が文字列の場合、ダイアログ項目のフィールド名です。

例

```
Sub Main
Begin Dialog UserDialog 200,120,.DialogFunc
Text 10,10,180,15,"Please push the OK button"
TextBox 10,40,180,15,.Text
OKButton 30,90,60,20
End Dialog
Dim dlg As UserDialog
Dialog dlg
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Debug.Print "Action=";Action%
Select Case Action%
Case 1 ' Dialog box initialization
Beep
For I = 0 To DlgCount()-1
Debug.Print  I;DlgType(I)
Next I
End Select
End Function
```

DlgValue Instruction/Function

DlgItem|$に数値を設定します。

```
DlgValue DlgItem|$, Value
```

DlgItem|$の数値を返します(MultiListBoxユーザダイアログの場合、配列を返します)。

```
DlgValue(DlgItem|$)
```

この命令/関数はDialogFuncから直接、または間接的に呼び出さなければなりません。DlgItem|$は、CheckBox、ComboBox、DropListBox、ListBox、MultiListBox、またはOptionGroupを参照します。

引数	説明
DlgItem|$	値が数値の場合、ダイアログの項目番号です。最初の項目が0で、二番目が1...です。値が文字列の場合、ダイアログ項目のフィールド名です。
Value	*DlgItem/$*のテキストをこの数値に設定します。(MultiListBoxユーザダイアログ項目は配列を使用します)。

例

```
Sub Main
Begin Dialog UserDialog 150,47,.DialogFunc
GroupBox 10,7,130,77,"Direction",.Field1
PushButton 100,28,30,21,"&Up"
PushButton 100,56,30,21,"&Dn"
OptionGroup .Direction
OptionButton 20,21,80,14,"&North",.North
OptionButton 20,35,80,14,"&South",.South
OptionButton 20,49,80,14,"&East",.East
OptionButton 20,63,80,14,"&West",.West
OKButton  10,91,130,21
CancelButton 10,119,130,21
End Dialog
Dim dlg As UserDialog
Dialog dlg
MsgBox "Direction=" & dlg.Direction
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Select Case Action%
Case 1 ' Dialog box initialization
Beep
Case 2 ' Value changing or button pressed
Select Case DlgItem$
Case "Up"
DlgValue "Direction",0
DialogFunc% = True 'do not exit the dialog
Case "Dn"
DlgValue "Direction",1
DialogFunc% = True 'do not exit the dialog
End Select
End Select
End Function
```

DlgVisible Instruction/Function

DlgItem|$を表示または非表示にします。

```
DlgVisible DlgItem|$[, Visible]
```

DlgItem|$が表示されている場合、Trueを返します。

```
DlgVisible(DlgItem|$)
```

この命令/関数はDialogFuncから直接、または間接的に呼び出さなければなりません。

引数	説明
DlgItem|$	値が数値の場合、ダイアログの項目番号です。最初の項目が0で、二番目が1...です。値が文字列の場合、ダイアログ項目のフィールド名です。
Visible	Trueの場合、DlgItem|$を表示。そうでない場合は非表示。省略した場合は、状態を切り替え。

例

```
Sub Main
Begin Dialog UserDialog 200,120,.DialogFunc
Text 10,10,180,15,"Please push the OK button"
TextBox 10,40,180,15,.Text
OKButton 30,90,60,20
PushButton 110,90,60,20,"&Hide"
End Dialog
Dim dlg As UserDialog
Debug.Print Dialog(dlg)
End Sub
Function DialogFunc%(DlgItem$, Action%, SuppValue%)
Debug.Print "Action=";Action%
Select Case Action%
Case 1 ' Dialog box initialization
Beep
Case 2 ' Value changing or button pressed
Select Case DlgItem$
Case "Hide"
DlgText DlgItem$,"&Show"
DlgVisible "Text",False
DialogFunc% = True 'do not exit the dialog
Case "Show"
DlgText DlgItem$,"&Hide"
DlgEnable "Text",True
DialogFunc% = True 'do not exit the dialog
End Select
End Select
End Function
```

DoEvents Instruction

他のアプリケーションがイベントを処理できるよう、実行を渡します。

```
DoEvents
```

例

```
Sub Main
DoEvents ' let other apps work
End Sub
```

End Instruction

スクリプトの実行を実行を直ちに終了させます。

```
End
```

例

```
Sub Main
L$ = UCase$(InputBox$("Enter End:"))
If L$ = "END" Then End
Debug.Print "End was not entered."
End Sub
Sub Main
Debug.Print "Before DoSub"
DoSub
Debug.Print "After DoSub"
End Sub
```

Environ Function

環境変数に関連付けられた文字列を返します。

```
Environ[$](Index)
```
または
```
Environ[$](Name)
```

引数	説明
Index	環境変数の値を返します。このインデックスに環境変数が無い場合、NULL文字列を返します。インデックスは1から始まります。
Name	環境変数の値を返します。環境変数が見つからない場合、NULL文字列を返します。

例

```
Sub Main
Debug.Print Environ("Path")
End Sub
```

EOF Function

StreamNumがファイルの最後に達した場合、Trueを返します。

```
EOF(StreamNum)
```

引数	説明
StreamNum	ストリーム1から255は、各マクロに対してプライベートです。ストリーム256から511はすべてのモジュールで共有されます。

例

```
Sub Main
Open "XXX" For Input As #1
While Not EOF(1)
Line Input #1,L$
Debug.Print L$
Wend
Close #1
End Sub
```

Erase Instruction

arrayvarをリセット、またはユーザ定義型の配列要素をゼロにリセットします(動的配列は次元化していない配列にリセットします)。文字列配列の値はnull文字列に設定します。arrayvarは配列として宣言する必要があります。

- Dim、Private、PublicまたはStaticとともに宣言します。
- Sub、FunctionやProperty定義の引数として宣言します。

```
Erase arrayvar[, ...]
 または
Erase usertypevar.elem[, ...]
```

例

```
Sub Main
Dim X%(2)
X%(1) = 1
Erase X%
Debug.Print X%(1) ' 0
End Sub
```

Error Instruction/Function

エラーを示すErrorCodeを通知します。これにより、実際のエラーと同様にエラー処理がトリガされます。現在のプロシージャのエラーハンドラがアクティブでなければ、アクティブにします。この場合、呼び出し側のプロシージャのエラーハンドラを試行します(完全なエラー情報を得るにはErr.Raiseを使用します)。

```
Error ErrorCode
```

Error()関数は、エラーテキストの文字列を返します。

```
Error[$]([ErrorCode])
```

引数	説明
ErrorCode	エラー番号。

例

```
Sub Main
On Error GoTo Problem
Err.Raise 1 ' simulate error #1
Exit Sub
Problem: ' error handler
Debug.Print "Error$=";Error$
Resume Next
End Sub
```

Eval Function

評価した文字列式の値を返します。

```
Eval(Expr[, Depth])
```

引数	説明
Expr	評価される文字列式。
Depth	どの深度のスタックにローカル変数を配置するかの整数。*Depth* = 0は現在のプロシージャを使用します。省略した場合、深度は0です。

例

```
Sub Main
Dim X As String
X = "Hello"
Debug.Print Eval("X") 'Hello
A
End Sub
Sub A
Dim X As String
X = "Bye"
Debug.Print Eval("X") 'Bye
Debug.Print Eval("X",1) 'Hello
End Sub
```

Exit Instruction

Exitは、残っている処理の一部、またはすべてを実行せずに終了します。

```
Exit {All|Do|For|Function|Property|Sub|While}
```

引数	説明
All	すべての実行を終了(Endと同じ)。
Do	Doループを終了。
For	For EachループのForを終了。
Function	ファンクションブロックを終了。 注意: この命令はErrをクリアし、Error$をNullに設定します。
Property	プロパティブロックを終了。 注意: この命令はErrをクリアし、Error$をNullに設定します。
Sub	Subブロックを終了。 注意: この命令はErrをクリアし、Error$をNullに設定します。
While	Whileループを終了。

例

```
Sub Main
L$ = InputBox$("Enter Do, For, While, Sub or All:")
Debug.Print "Before DoSub"
DoSub UCase$(L$)
Debug.Print "After DoSub"
End Sub
Sub DoSub(L$)
Do
If L$ = "DO" Then Exit Do
I = I+1
Loop While I < 10
If I = 0 Then Debug.Print "Do was entered"
For I = 1 To 10
If L$ = "FOR" Then Exit For
Next I
If I = 1 Then Debug.Print "For was entered"
I = 10
While I > 0
If L$ = "WHILE" Then Exit While
I = I-1
Wend
If I = 10 Then Debug.Print "While was entered"
If L$ = "SUB" Then Exit Sub
Debug.Print "Sub was not entered."
If L$ = "ALL" Then Exit All
Debug.Print "All was not entered."
End Sub
```

Exp Function

指数を返します。

```
Exp(Num)
```

引数	説明
Num	任意の有効な数値、または数式。戻り値はeを底とする数式のべき乗。定数eは約 2.718282です。

Logも参照してください。

例

```
Sub Main
Debug.Print Exp(1) ' 2.718281828459
End Sub
```

FileAttr Function

StreamNumのオープンモードまたはハンドルを返します。

```
FileAttr(StreamNum, ReturnValue)
```

引数	説明
StreamNum	1から255は、各モジュールに対してプライベートです。256から511はすべてのモジュールで共有されます。
ReturnValue	返す情報の種類。 オープンモード: ファイルを開くために使用されたモードを返します。 下表を参照してください。 ハンドル: ファイルハンドルを返します。 Win16のみ。Win32ではエラーが発生。

値	モード
1	入力
2	出力
4	ランダム
8	追加
32	バイナリ

Openも参照してください。

例

```
Sub Main
Open "XXX" For Output As #1
Debug.Print FileAttr(1,1) ' 2
Close #1
End Sub
```

FileCopy Instruction

ファイルをコピーします。

```
FileCopy FromName$, ToName$
```

引数	説明
FromName$	コピー元ファイルのパスと名前を示す文字列。現在のディレクトリを基準とする相対パスも使用できます。
ToName$	コピー先ファイルのパスと名前を示す文字列。現在のディレクトリを基準とする相対パスも使用できます。

例

```
Sub Main
FileCopy "C:\AUTOEXEC.BAT","C:\AUTOEXEC.BAK"
End Sub
```

FileDateTime Function

ファイルName$が最後に更新した日時を返します。ファイルが存在しない場合、実行時にエラーが発生します。

```
FileDateTime(Name$)
```

引数	説明
Name$	ファイルのパスと名前を示す文字列。現在のディレクトリを基準とする相対パスも使用できます。

例

```
Sub Main
F$ = Dir$("*.*")
While F$ <> ""
Debug.Print F$;" ";FileDateTime(F$)
F$ = Dir$()
Wend
End Sub
```

FileLen Function

ファイルName$の長さを返します。ファイルが存在しない場合、実行時にエラーが発生します。

```
FileDateTime(Name$)
```

引数	説明
Name$	ファイルのパスと名前を示す文字列。現在のディレクトリを基準とする相対パスも使用できます。

例

```
Sub Main
F$ = Dir$("*.*")
While F$ <> ""
Debug.Print F$;" ";FileLen(F$)
F$ = Dir$()
Wend
End Sub
```

Fix Function

整数値を返します。

```
Fix(Num)
```

引数	説明
Num	この数値の整数部分を返します。小数部は切り捨てられます。正の値は直近のより小さい値を返します。負の値は直近のより大きい値を返します。値がNullの場合、Nullを返します。

Intも参照してください。

例

```
Sub Main
Debug.Print Fix(9.9) ' 9
Debug.Print Fix(0) ' 0
Debug.Print Fix(-9.9) '-9
End Sub
```

Format$ Function

exprを指定した書式の文字列で返します。

```
Format[$](expr[, form$], [firstday], _
[firstweek])
```

引数	説明
expr	任意の有効な文字列表現。
form$	exprを文字列値にフォーマットします。これを省略した場合は、*expr*を文字列として返します。
firstday	週の最初の曜日として使用する定数。下表を参照してください。省略した場合、*vbSunday*を使用します(Win32のみ)。
firstweek	年の最初の週として使用する定数。下表を参照してください。省略した場合、*vbFirstJan1*を使用します(Win32のみ)。

firstday	値	説明
bUseSystemFirstDay	0	システムで指定してる週の最初の曜日。
bSunday	1	日曜日(デフォルト)。
bMonday	2	月曜日。
bTuesday	3	火曜日。
bWednesday	4	水曜日。
bThursday	5	木曜日。
bFriday	6	金曜日。
bSaturday	7	土曜日。

firstweek	値	説明
bUseSystem	0	システムで指定してる年の最初の週。
bFirstJan1	1	1月1日が含まれる週(デフォルト)。
vbFirstFourDays	2	その年の4日以上ある最初の週。
vbFirstFullWeek	3	その年の1週間全体が含まれる最初の週。

Predefined Date Format、Predefined Number Format、User defined Date Format、User defined Number Format、User defined Text Formatも参照してください。

Format Predefined Date

Format関数は定義済みの日付書式を使用することができます。定義済みの書式は、ユーザ定義の書式や他の定義済みの書式と組み合わせることはできません。

書式	説明
General Date	日付か時刻、または両方を表示。日付表示はシステム設定に依存します。 Format User Defined Dateの"c"と同じです。
Long Date	現在のシステム設定の長い日付書式に従って日付を表示。 Format User Defined Dateの"dddddd"と同じです。
Medium Date	現在はサポートしていません。
Short Date	現在のシステム設定の短い日付書式に従って日付を表示。 Format User Defined Dateの"ddddd"と同じです。
Long Time	現在のシステム設定の長い日付書式に従って時間を表示。 Format User Defined Dateの"ttttt"と同じです。
Medium Time	時刻を12時間形式で表示。 Format User Defined Dateの"hh:mm AMPM"と同じです。
Short Time	時刻を24時間形式で表示。 Format User Defined Dateの"hh:mm"と同じです。

Format Predefined Number

Format関数は定義済みの数値書式を使用することができます。定義済みの書式は、ユーザ定義の書式や他の定義済みの書式と組み合わせることはできません。

書式	説明
General Number	桁区切り記号を付けず、そのまま数値を返します。
Currency	3桁ごとに区切り記号を付けて数値を表示。小数部がある場合、2桁まで表示。 Format User Defined Numberの"$#,##0.00;($#,##0.00)"と同じです。 現在はシステム ロケール設定に依存しません。
Fixed	少なくとも整数部1桁、小数部2桁を表示。 Format User Defined Numberの"0.00"と同じです。
Standard	桁区切り記号を付けて数値を表示。少なくとも整数部1桁、小数部2桁を表示。 Format User Defined Numberの"#,##0.00"と同じです。
Percent	数値を100 倍し、右側にパーセント記号(%)を付けて表示。小数部は常に2桁。 Format User Defined Numberの"0.00%"と同じです。
Scientific	標準的な指数表記を使用。有効桁数は2桁。 Format User Defined Numberの"0.00E+00"と同じです。
Yes/No	数値が0の場合はNoを表示し、それ以外の場合はYesを表示。
True/False	数値が0の場合はFalseを表示し、それ以外の場合はTrueを表示。
On/Off	数値が0の場合は Offを表示し、それ以外の場合はOnを表示。

例

```
Sub Main
Debug.Print Format$(2.145,"Standard") ' 2.15
End Sub
```

Format User Defined Date

Format関数はユーザ定義日付書式を使用することができます。ユーザ定義日付書式を組み合わせて日付書式を作成することができます。ユーザ定義の書式は、定義済みの書式や他のユーザ定義の書式と組み合わせることはできません。

引数	説明
:	ロケールによる時刻の区切り記号。
/	ロケールによる日付の区切り記号。
c	日付、時刻をを*ddddd ttttt*で表示し、*t* = 0の場合は日付のみを表示し、*d* = 0の場合は時間のみを表示。
d	前に0を付けないで日を表示。
dd	前に0を付けて日を表示。
ddd	短縮した曜日名を表示。
dddd	曜日をそのまま表示。
ddddd	短い日付形式で表示。
dddddd	長い日付形式で表示。
w	曜日番号を表示。
ww	週番号を表示。
m	前に0を付けないで月番号を表示。 前に0を付けないで分を表示(hまたはhhの後に続く場合)。
mm	前に0を付けて月番号を表示。 前に0を付けて分を表示(hまたはhhの後に続く場合)。
mmm	月を短縮形で表示。
mmmm	月をそのまま表示。
q	四半期番号を表示。
y	前に0を付けないで年を表示。
yy	年を2桁で表示。
yyyy	年を4桁で表示。
h	前に0を付けないで時間を表示。
hh	前に0を付けて時間を表示。
n	前に0を付けないで分を表示。
nn	前に0を付けて分を表示。
s	前に0を付けないで秒を表示。
ss	前に0を付けて秒を表示。
ttttt	時間書式に従って時間を表示。
AM/PM	12時間制を使用して午前は大文字のAM(0から11)、および午後はPM(12から23)を表示。
am/pm	12時間制を使用して午前は小文字のam(0から11)、および午後はpm(12から23)を表示。
A/P	12時間制を使用して午前は大文字のA(0から11)、および午後はP(12から23)を表示。
a/p	12時間制を使用して午前は小文字のa(0から11)、および午後はp(12から23)を表示。
AMPM	12時間制を使用してロケールのAM/PMを表示。
\c	文字cを表示。
"text"	リテラル文字列を表示。

Format User Defined Number

Format関数はユーザ定義数値書式を使用することができます。ユーザ定義数値書式を組み合わせて数値書式を作成することができます。ユーザ定義の書式は、定義済みの書式や他のユーザ定義の書式と組み合わせることはできません。
ユーザ定義数値書式は、";"で4つまでのセクションを区切ることができます:

```
form;negform;zeroform;nullform
```

- form　正の数に適用する書式設定
- negform　負の数に適用する書式設定
- 0に適用する書式設定
- 文字列に適用する書式設定

引数	説明
#	有効桁でない場合、先頭と末尾の桁の0は表示しません。 例えば、Format(19,"###")の戻り値は"19" 例えば、Format(19,"#")の戻り値も"19"
0	先頭と末尾の桁に0を表示。 例えば、Format(19,"000")の戻り値は"019" 例えば、Format(19,"0")の戻り値は"19"
.	ロケールに従って小数点を挿入。 例えば、Format(19.9,"###.00")の戻り値は"19.90" 例えば、Format(19.9,"###.##")の戻り値は"19.9"
,	3桁ごとに区切り記号を挿入。 "xxx,"または"xxx,."は、数値を1000で除算し、必要に応じて丸めます。 隣接する",,"は、数値を再び1000で除算することを意味します。 例えば、Format(1900000,"0,,")の戻り値は"2" 例えば、Format(1900000,"0,,.0")の戻り値は"1.9"
%	値に100 を乗算し、パーセント記号を挿入。
:	ロケールによる時刻の区切り記号。
/	ロケールによる日付の区切り記号。
E+ e+ E- e-	指数表記を使用。 例えば、Format(1000,"0.00E+00")の戻り値は"1.00E+03" 例えば、Format(.001,"0.00E+00")の戻り値は"1.00E-03"
- + $ ( ) space	リテラル文字を表示。 例えば、Format(10,"$#")の戻り値は"$10"
\c	文字cを表示。 例えば、Format(19,"\####\#")の戻り値は"#19#"
"text"	リテラル文文字列を表示。 例えば、Format(19,"""##""###""##""")の戻り値は"##19##"

例

```
Sub Main
Debug.Print Format$(2.145,"#.00") ' 2.15
End Sub
```

Format User Defined Text

Format関数はユーザ定義文字列書式を使用することができます。ユーザ定義文字列書式を組み合わせて文字列書式を作成することができます。ユーザ定義文字列書式は、定義済みの書式や他のユーザ定義文字列書式と組み合わせることはできません。
ユーザ定義文字列書式は";"で2つまでのセクションを区切ることができます。

引数	説明
@	文字列のプレースフォルダ。文字またはスペースを表示。
&	文字列のプレースフォルダ。文字を表示または何も表示しない。
<	すべての文字を小文字で表示。
>	すべての文字を大文字で表示。
!	プレースホルダーを左から右に指定(デフォルトは右から左)。
\c	文字cを表示。
"text"	リテラル文文字列を表示。

例

```
Sub Main
Debug.Print Format("123","ab@c") '" ab1c23"
Debug.Print Format("123","!ab@c") '" ab3c"
End Sub
```

FreeFile Function

次の未使用共有ストリーム番号(256以上)を返します。ストリーム1から255は、各マクロに対してプライベートです。256から511はすべてのマクロで共有されます。

```
FreeFile[()]
```

例

```
Sub Main
Debug.Print FreeFile ' 256
FN = FreeFile
Open "XXX" For Output As #FN
Debug.Print FreeFile ' 257
Close #FN
Debug.Print FreeFile ' 256
End Sub
```

Get Instruction

StreamNumから変数の値を取得します。

```
Get StreamNum, [RecordNum], var
```

引数	説明
StreamNum	ストリーム1から255は、各マクロに対してプライベートです。256から511はすべてのマクロで共有されます。
RecordNum	Randomモードのファイルの場合、レコード番号で、最初のレコードは1です。それ以外の場合は、バイト位置で、最初のバイトは1です。省略した場合、現在の位置(またはレコード番号)を使用します。バリアント変数の場合、その型を説明する2バイトが読み取り、それに応じて変数値が読み取られます。usertype変数の場合、各フィールドは順番に読み込まれます。配列変数の場合、各要素は順番に読み込まれます。動的配列変数の場合、配列値を読み取る前に各次元の次元数と範囲を読み込みます。すべてのバイナリデータ値は、little-endian形式でファイルから読み込まれます。 Binaryモードのファイルから文字列(または動的配列)を読み取る場合、長さ(または配列次元)情報は読み込まれません。現在の文字列長さによって、読み込まれる文字列のデータ量を決定します。現在の配列の次元によって、配列要素をどのように読み込むかを決定します。
var	変数値*var*はファイルから読み取られます。固定長の変数(Longなど)の場合、変数を復元するために必要なバイト数を読み取ります。

Open、Putも参照してください。

例

```
Sub Main
Dim V As Variant
Open "SAVE_V.DAT" For Binary Access Read As #1
Get #1, , V
Close #1
End Sub
```

GetAllSettings Function

AppNameプロジェクトのセクションの設定をすべて取得します。設定はVariantで返されます。セクションにキーがない場合、空が返されます。それ以外の場合、Variantには2次元配列が含まれます: (I, 0)がキー、(I,1)が設定です。Win16およびWin32sは、AppNameの.iniファイル名に設定を保存します。Win32は設定を登録データベースに保存します。

```
GetAllSettings(AppName$, Section$, Key$)
```

引数	説明
AppName$	セクションおよびキーを持ったプロジェクト名前の文字列値。
Section$	プロジェクト設定のセクション名の文字列値。

例

```
Sub Main
SaveSetting "MyApp","Font","Size",10
SaveSetting "MyApp","Font","Name","Courier"
Settings = GetAllSettings("MyApp","Font")
For I = LBound(Settings) To UBound(Settings)
Debug.Print Settings(I,0); "="; Settings(I,1)
Next I
DeleteSetting "MyApp","Font"
End Sub
```

GetAttr Function

Name$ファイルの属性を返します。ファイルが存在しない場合、実行時エラーが発生します。

```
GetAttr(Name$)
```

引数	説明
Name$	ファイルのパスおよび名前の文字列。現在のディレクトリからの相対パスが使用できます。

例

```
Sub Main
F$ = Dir$("*.*")
While F$ <> ""
Debug.Print F$;" ";GetAttr(F$)
F$ = Dir$()
Wend
End Sub
```

GetSetting Function

AppNameプロジェクトのセクションのキー設定値を取得します。Win16およびWin32sは、AppNameの.iniファイル名に設定を保存します。Win32は設定を登録データベースに保存します。

```
GetSetting[$](AppName$, Section$, Key$[, Default$])
```

引数	説明
AppName$	セクションおよびキーを持ったプロジェクト名の文字列値。
Section$	プロジェクト設定のセクション名の文字列値。
Key$	プロジェクト設定のセクションのキー名の文字列値。
Default$	キーに設定が保存されていない場合に返す文字列。省略した場合、null文字列を使用します。

例

```
Sub Main
SaveSetting "MyApp","Font","Size",10
Debug.Print GetSetting("MyApp","Font","Size") ' 10
End Sub
```

Goto Instruction

labelに移動し、そこから実行を継続します。現在のユーザ定義プロシージャ内のlabelにのみアクセスできます。

```
GoTo label
```

例

```
Sub Main
X = 2
Loop:
X = X*X
If X < 100 Then GoTo Loop
Debug.Print X ' 256
End Sub
```

Hex$ Function

16進文字列を返します。

```
Hex[$](Num)
```

引数	説明
Num	16進数でエンコードした文字列を返す数値。

例

```
Sub Main
Debug.Print Hex$(15) 'F
End Sub
```

Hour Function

1日の時刻を表す数値(0から23)を返します。

```
Hour(dateexpr)
```

引数	説明
dateexpr	日付の時刻を返します。値がNullの場合、Nullを返します。

Minute()、Second()、Time()も参照してください。

例

```
Sub Main
Debug.Print Hour(#12:00:01 AM#) ' 0
End Sub
```

IIf Function

condexprで指定したパラメータ値を返します。TruePartおよびFalsePartの両方を評価します。

```
IIf(condexpr, TruePart, FalsePart)
```

引数	説明
condexpr	値がTrueの場合はTruePartを返します。それ以外はFalsePartを返します。
TruePart	condexprがTrueの場合に返す値。
FalsePart	condexprがFalseの場合に返す値。

If、Select Case、Choose( )も参照してください。

例

```
Sub Main
Debug.Print  IIf(1 > 0,"True","False") ' "True"
End Sub
```

Input Instruction

StreamNumから入力を受け取り、varに割り当てます。入力値はコンマで区切られます。先頭および末尾のスペースは無視されます。最初の文字(先頭のスペースから続く)が引用符(")の場合、文字列は終了引用符で終了します。#NULL#、#FALSE#、#TRUE#、#date#および#ERROR number#の特殊な値は、適切な値およびデータ型に変換されます。

```
Input [#]StreamNum, var[, ...]
```

Line Input、Print、Writeも参照してください。

例

```
Sub Main
Open "XXX" For Input As #1
Input #1,A,B,C$
Debug.Print A;B;C$
Close #1
End Sub
```

Input$ Function

StreamNumからN個の文字を返します。

```
Input[$](N, StreamNum)
```

引数	説明
N	読み込んだ値を格納する変数。ファイルの終了前に、その数より少ない文字が残っている場合、ランタイムエラーが発生します。
StreamNum	ストリーム1から255は、各マクロに対してプライベートです。256から511はすべてのマクロで共有されます。

例

```
Sub Main
Open "XXX" For Input As #1
L = LOF(1)
T$ = Input$(L,1)
Close #1
Debug.Print T$;
End Sub
```

InStr Function

S2$が最初にS1$と一致するインデックスを返します。一致するものが見つからない場合は0を返します。

同様の関数InStrBは、代わりにバイトインデックスを返します。

```
InStr([Index, ]S1$, S2$)
```

引数	説明
Index	S1$内のこのインデックスからS2$の検索を開始します。省略した場合はS1$の先頭から検索します。
S1$	この文字列値でS2$を検索します。この値がNullの場合は、Nullを返します。
S2$	S1$でこの文字列値を検索します。この値がNullの場合は、Nullを返します。

InStrRev( )、Left$( )、Len( )、Mid$( )、Replace$( )、Right$( )も参照してください。

例

```
Sub Main
Debug.Print InStr("Hello","l") ' 3
End Sub
```

InStrRev Function

S2$が最後にS1$と一致するインデックスを返します。一致するものが見つからない場合は0を返します。

```
InStrRev(S1$, S2$[, Index])
```

引数	説明
S1$	この文字列値でS2$を検索します。この値がNullの場合は、Nullを返します。
S2$	S1$でこの文字列値を検索します。この値がNullの場合は、Nullを返します。
Index	S1$内のこのインデックスで終了するS2$の検索を開始します。省略した場合はS1$の先頭から検索します。

Left$()、Len()、Mid$()、Replace$()、Right$()も参照してください。

Int Function

整数値を返します。

```
Int(Num)
```

引数	説明
Num	この数値以下の最大の整数を返します。この値がNullの場合は、Nullを返します。

例

```
Sub Main
Debug.Print Int(9.9) ' 9
Debug.Print Int(0) ' 0
Debug.Print Int(-9.9) '-10
End Sub
```

IsArray Function

varが値の配列の場合はTrueを返します。

```
IsArray(var)
```

引数	説明
var	配列変数またはバリアントvarには複数の値を含めることができます。

例

```
Sub Main
Dim X As Variant, Y(2) As Integer
Debug.Print IsArray(X) 'False
X = Array(1,4,9)
Debug.Print IsArray(X) 'True
X = Y
Debug.Print IsArray(X) 'True
End Sub
```

IsDate Function

exprが有効な日付の場合はTrueを返します。

```
IsDate(expr)
```

引数	説明
expr	有効な日付かどうかをテストするバリアント式。

TypeName、VarTypeも参照してください。

例

```
Sub Main
Dim X As Variant
X = 1
Debug.Print IsDate(X) 'False
X = Now
Debug.Print IsDate(X) 'True
End Sub
```

IsEmpty Function

variantvarが空の場合はTrueを返します。

```
IsEmpty(variantvar)
```

引数	説明
variantvar	値が割り当てられていない場合、バリアント var は空になります。

TypeName、VarTypeも参照してください。

例

```
Sub Main
Dim X As Variant
Debug.Print IsEmpty(X) 'True
X = 0
Debug.Print IsEmpty(X) 'False
X = Empty
Debug.Print IsEmpty(X) 'True
End Sub
```

IsError Function

exprがエラーコードの場合はTrueを返します。

```
IsError(expr)
```

引数	説明
expr	エラーコード値をテストするバリアント式。

TypeName、VarTypeも参照してください。

例

```
Sub Main
Dim X As Variant
Debug.Print IsError(X) 'False
X = CVErr(1)
Debug.Print IsError(X) 'True
End Sub
```

IsMissing Function

オプションのパラメータvariantvarにデフォルト値がなく、値を取得できなかった場合は、Trueを返します。Sub、Function、またはProperty呼び出しでは、オプションの引数を省略できます。

```
IsMissing(variantvar)
```

引数	説明
variantvar	このバリアント引数の引数式が Sub、Function、またはProperty呼び出しで指定されていない場合は、Trueを返します。

TypeName、VarTypeも参照してください。

例

```
Sub Main
Opt 'IsMissing(A)=True
Opt "Hi" 'IsMissing(A)=False
Many 'No args
Many 1,"Hello" 'A(0)=1 A(1)=Hello
OptBye '"Bye"
OptBye "No" '"No"
End Sub
Sub Opt(Optional A)
Debug.Print "IsMissing(A)=";IsMissing(A)
End Sub
Sub Many(ParamArray A())
If LBound(A) > UBound(A) Then
Debug.Print "No args"
Else
For I = LBound(A) To UBound(A)
Debug.Print "A(" & I & ")=" & A(I) & " ";
Next I
Debug.Print
End If
End Sub
Sub OptBye(Optional A As String = "Bye")
Debug.Print A
End Sub
```

IsNull Function

exprがNullの場合はTrueを返します。

```
IsNull(expr)
```

引数	説明
expr	Nullをテストするバリアント式。

TypeName、VarTypeも参照してください。

例

```
Sub Main
Dim X As Variant
Debug.Print IsNull(X) 'False
X = 1
Debug.Print IsNull(X) 'False
X = "1"
Debug.Print IsNull(X) 'False
X = Null
Debug.Print IsNull(X) 'True
X = X*2
Debug.Print IsNull(X) 'True
End Sub
```

IsNumeric Function

exprが数値の場合はTrueを返します。

```
IsNumeric(expr)
```

引数	説明
expr	バリアント式は、数値または数値を表す文字列値の場合は数値です。

TypeName、VarTypeも参照してください。

例

```
Sub Main
Dim X As Variant
Debug.Print IsNumeric(X) 'True
X = 1
Debug.Print IsNumeric(X) 'True
X = "A"
Debug.Print IsNumeric(X) 'False
End Sub
```

IsObject Function

var にオブジェクト参照が含まれる場合はTrueを返します。

```
IsObject(var)
```

引数	説明
var	objexpr参照の場合、varにはオブジェクト参照が含まれます。

TypeName、VarTypeも参照してください。

例

```
Sub Main
Dim X As Variant
X = 1
Debug.Print IsObject(X) 'False
X = "1"
Debug.Print IsObject(X) 'False
set X = Nothing
Debug.Print IsObject(X) 'True
End Sub
```

Join Function

配列内のすべての値を各値の間に*Sep*で区切り、連結した文字列を返します。

```
Join(StrArray, [Sep])
```

引数	説明
StrArray	この配列からの値を連結します。
Sep	この文字列値を使用して値を区切ります。(デフォルト: " ")

Split( )も参照してください。

例

```
Sub Main
Debug.Print Join(Array(1,2,3)) '"1 2 3"
End Sub
```

KeyName Function

キー番号のキー名を返します。これはSendKeysで使用される名前です。

```
KeyName(Key)
```

引数	説明
Key	キー番号。

SendKeysも参照してください。

例

```
Sub Main
Debug.Print KeyName(&H270) '"^{F1}"
End Sub
```

Kill Instruction

Name$で指定されたファイルを削除します。

```
Kill Name$
```

引数	説明
Name$	この文字列値はファイルのパスと名前です。現在のディレクトリからの相対パスを使用できます。

例

```
Sub Main
Kill "XXX"
End Sub
```

LBound Function

最小のインデックスを返します。

```
LBound(arrayvar[, dimension])
```

引数	説明
arrayvar	この配列変数の最小インデックスを返します。
dimension	*arrayvar*のこの次元の最小インデックスを返します。これを省略した場合は、最初の次元の最小インデックスを返します。

UBound()も参照してください。

例

```
Sub Main
Dim A(-1 To 3,2 To 6)
Debug.Print LBound(A) '-1
Debug.Print LBound(A,1) '-1
Debug.Print LBound(A,2) '2
End Sub
```

LCase$ Function

すべての大文字を小文字したS$の文字列を返します。

```
LCase[$](S$)
```

引数	説明
S$	すべての文字を小文字に変換した後、この文字列値を返します。この値がNullの場合は、Nullを返します。

StrComp( )、StrConv$( )、UCase$( )、も参照してください。

例

```
Sub Main
Debug.Print LCase$("Hello") '"hello"
End Sub
```

Left$ Function

S$からLen文字のみを含む文字列を返します。

同様の関数LeftBは、最初のLenバイトを返します。

```
Left[$](S$, Len)
```

引数	説明
S$	この文字列値の左側の部分を返します。この値がNullの場合は、Nullを返します。
Len	この数の文字を返します。S$がそれより短い場合は、S$を返します。

InStr()、InStrRev()、Len()、Mid$()、Replace$()、Right$()も参照してください。

例

```
Sub Main
Debug.Print Left$("Hello",2) '"he"
End Sub
```

Len Function

S$の文字数を返します。

同様の関数LenBは、文字列内のバイト数を返します。usertypevarの場合、LenBは変数のデータが占有するメモリのバイト数を返します。

```
Len(S$)
```

引数	説明
S$	この文字列値の文字数を返します。この値がNullの場合は、Nullを返します。
usertypevar	

InStr( )、InStrRev( )、Left$( )、Mid$( )、Replace$( )、Right$( )も参照してください。

例

```
Sub Main
Debug.Print Len("Hello") '5
End Sub
```

Let Instruction

varにexprの値を割り当てます。Letキーワードは任意です。

```
[Let] var = expr
```

例

```
Sub Main
  Let X = 1
  X = X*2
  Debug.Print X ' 2
End Sub
```

Line Input Instruction

StreamNumから入力行を取得し、それをS$に割り当てます。

```
Line Input [#]StreamNum, S$
```

Input、Print、Writeも参照してください。

例

```
Sub Main
Open "XXX" For Input As #1
Line Input #1,S$
Debug.Print S$
Close #1
End Sub
```

Loc Function

StreamNumファイルの位置を返します。

```
Loc(StreamNum)
```

Randomモードファイルの場合、これは現在のレコード番号から1を引いた値になります。Binaryモードファイルの場合、現在のバイト位置から1を引いた値になります。それ以外の場合、現在のバイト位置から1を引いた値を128で割った値になります。ファイル内の最初の位置は0です。

引数	説明
StreamNum	ストリーム1から255は各マクロに対してプライベートです。ストリーム256から511はすべてのマクロで共有されます。

例

```
Sub Main
Open "XXX" For Input As #1
L = Loc(1)
Close #1
Debug.Print L ' 0
End Sub
```

Lock Instruction

Form 1: StreamNumをすべてロックします。

```
Lock StreamNum
```

Form 2: StreamNumのレコード(またはバイト)をロックします。

```
StreamNum, RecordNum
```

Form 3: StreamNumのレコード(またはバイト)の範囲をロックします。startが省略された場合は、最初のレコード(またはバイト)からロックが開始されます。

```
Lock StreamNum, [start] To end
```

注記

- Lock命令ごとに必ずUnlock 行ってください。
- シーケンシャル ファイル(入力、出力、追加)の場合、ロックは常にファイル全体に影響します。

引数	説明
StreamNum	ストリーム1から255は各マクロに対してプライベートです。ストリーム256から511はすべてのマクロで共有されます。
RecordNum	Randomモードファイルの場合、これはレコード番号です。最初のレコードは1です。それ以外の場合はバイト位置です。最初のバイトは1です。
start	範囲内の最初のレコード(またはバイト)。
end	範囲内の最後のレコード(またはバイト)。

Open、Unlockも参照してください。

例

```
Sub Main
Dim V As Variant
Open "SAVE_V.DAT" For Binary As #1
Lock #1
Get #1, 1, V
V = "Hello"
Put #1, 1, V
Unlock #1
Close #1
End Sub
```

LOF Function

StreamNumファイルの長さ(バイト単位)を返します。

```
LOF(StreamNum)
```

引数	説明
StreamNum	ストリーム1から255は各マクロに対してプライベートです。ストリーム256から511はすべてのマクロで共有されます。

例

```
Sub Main
Open "XXX" For Input As #1
L = LOF(1)
Close #1
Debug.Print L
End Sub
```

Log Function

自然対数を返します。

```
Log(Num)
```

引数	説明
Num	この数値の自然対数を返します。値eは約2.718282です。

Expも参照してください。

例

```
Sub Main
Debug.Print Log(1) ' 0
End Sub
```

LSet Instruction

Form 1:　strvarにstrの値を割り当てます。末尾の文字を削除しstrを短縮(または空白で延長)します。以前のstrvarの長さが維持されます。

```
LSet strvar = str
```

Form 2:　usertypevar1にusertypevar2の値を割り当てます。usertypevar2がusertypevar1より長い場合、usertypevar1が処理できる量だけをコピーします。

```
LSet usertypevar1 = usertypevar2
```

RSetも参照してください。

例

```
Sub Main
  S$ = "123"
  LSet S$ = "A"
  Debug.Print ".";S$;"." '".A ."
End Sub
```

LTrim$ Function

S$の先頭のスペースを削除した文字列を返します。

```
LTrim[$](S$)
```

引数	説明
S$	この文字列を先頭のスペースを除いてコピーします。この値がNullの場合は、Nullを返します。

RTrim$()、Trim$()も参照してください。

例

```
Sub Main
Debug.Print ".";LTrim$(" x ");"." '".x ."
End Sub
```

Mid$ Function/Assignment

Function: Len文字のIndexで始まるS$の部分文字列を返します。

```
Mid[$](S$, Index[, Len])
```

Instruction: Len文字のIndexから始まるstrvarの部分文字列にS$を割り当てます。

```
Mid[$](strvar, Index[, Len]) = S$
```

同様の関数MidBは、Indexバイトから始まるLenバイトを返します。

引数	説明(Mid Function)
S$	この文字列値から文字をコピーします。この値がNullの場合は、Nullを返します。
Index	このインデックス値から文字のコピーを開始します。文字列の長さより大きい値の場合、NULL文字列を返します。
Len	この数の文字をコピーします。Indexから始まる文字数がS$の長さより大きくなる場合、S$の末尾までをコピーします。

引数	説明(Mid Assignment)
strvar	この文字列の一部を変更します。
Index	このインデックス値からstrvarを変更します。文字列の長より小さい場合、文字列は変更されません。
Len	次の中で最小文字の数をコピーします: Lenの値、S$の長さ、およびstrvarの残りの長さ この値を省略した場合、コピーされる文字の数は、S$の長さとstrvarの残りの長さの最小値になります。
S$	この文字列値から文字をコピーします。

InStr()、Left$()、Len()、Replace$()、Right$()も参照してください。

例

```
Sub Main
S$ = "Hello There"
Mid$(S$,7) = "??????????"
Debug.Print S$ '"Hello ?????"
Debug.Print Mid$("Hello",2,1) '"e"
End Sub
```

Minute Function

時間の分を返します(0～59)。

```
Minute(dateexpr)
```

引数	説明
dateexpr	この日付値の分を返します。この値がNullの場合は、Nullが返されます。

Hour()、Second()、Time()も参照してください。

例

```
Sub Main
Debug.Print Minute(#12:00:01 AM#) ' 0
End Sub
```

MkDir Instruction

ディレクトリName$を作成します。

```
MkDir Name$
```

引数	説明
Name$	この文字列値は、ディレクトリのパスと名前です。現在のディレクトリからの相対パスを使用できます。

RmDirも参照してください。

例

```
Sub Main
MkDir "C:\WWTEMP"
End Sub
```

Month Function

年の月(1～12)を返します。

```
Month(dateexpr)
```

引数	説明
dateexpr	この日付値の月を返します。この値がNullの場合は、Nullが返されます。

Date( )、Day( )、MonthName( )、Weekday( )、Year( )も参照してください。

例

```
Sub Main
Debug.Print Month(#1/1/1900#) ' 1
Debug.Print Month(#2/1/1900#) ' 2
End Sub
```

MonthName Function

月の名前を返します。

```
MonthName(NumZ{month}[, CondZ{abbrev}])
```

引数	説明
month	ローカライズされた月名を返します(1-12)。
abbrev	この条件値がTrueの場合、月名の短縮形を返します。

Month( )も参照してください。

例

```
Sub Main
Debug.Print MonthName(1) 'January
Debug.Print MonthName(Month(Now))
End Sub
```

Name Instruction

ファイル名をOldName$からNewName$に変更します。

```
Name OldName$ As NewName$
```

引数	説明
OldName$	変更前のファイルパスおよび名前。現在のディレクトリからの相対パスを使用できます。
NewName$	新しいファイル名(およびパス)。現在のディレクトリからの相対パスを使用できます。

例

```
Sub Main
Name "AUTOEXEC.BAK" As "AUTOEXEC.SAV"
End Sub
```

Now Function

現在の日付と時刻を日付値として返します。

```
Now
```

Date、Time、Timerも参照してください。

例

```
Sub Main
Debug.Print Now ' example: 1/1/1995 10:05:32 AM
End Sub
```

Oct$ Function

8進数の文字列を返します。

```
Oct[$](Num)
```

引数	説明
Num	この数値を8進数でエンコードした文字列を返します。

Hex$()、Str$()、Val()も参照してください。

例

```
Sub Main
Debug.Print Oct$(15) '17
End Sub
```

On Error Instruction

Form 1: エラーハンドラを無効にします(デフォルト)。

```
On Error GoTo 0
```

Form 2: エラー条件をエラーハンドラに送信します。

```
Error GoTo label
```

Form 3: エラー条件が発生した場合、次のステートメントで実行を継続します。

```
On Error Resume Next
```

On Errorは、エラーハンドラを設定または無効にします。各ユーザ定義プロシージャには独自のエラーハンドラがあります。デフォルトでは、エラーが発生するとスクリプトの実行が終了します。Errオブジェクトのプロパティは、エラーが発生するたびに設定されます。エラーが発生し、エラーハンドラが実行されると、Errオブジェクトがクリアされない限り、それ以上のエラーが発生するとスクリプトの実行が終了します。

この命令はErrをクリアし、Error$をnullに設定します。

例

```
Sub Main
On Error Resume Next
Err.Raise 1
Debug.Print "RESUMING, Err=";Err
On Error GoTo X
Err.Raise 1
Exit Sub
X: Debug.Print "Err=";Err
Err.Clear
Debug.Print "Err=";Err
Resume Next
End Sub
```

Open Instruction

ファイルName$をStreamNumのモードで開きます。

```
  Open Name$ For mode [Access access] [lock] As _
[#]StreamNum [Len = RecordLen]
```

引数	説明
Name$	この文字列値はファイルパスと名前です。現在のディレクトリからの相対パスを使用できます。モードは、入力、出力、追加、バイナリ、またはランダムです。access は、読み取り、書き込み、または読み取り/書き込みのいずれかです。ロックは共有、ロック読み取り、ロック書き込み、またはロック読み取り/書き込みのいずれかです。
StreamNum	ストリーム1～255は各マクロにプライベートです。ストリーム256～511はすべてのマクロで共有されます。
RecordLen	この数値は、ランダムモードファイルのレコード長です。他のファイルモードではこの値は無視されます。

Close、FileAttr、FreeFile、Resetも参照してください。

例

```
Sub Main
Open "XXX" For Output As #1
Print #1,"1,2,""Hello"""
Close #1
End Sub
```

Print Instruction

exprをStreamNumに出力します。式を区切るにはセミコロン(;)を使用します。数値は、出力前に自動的に文字列に変換されます(Str$( )と同様)。命令が ; で終わっていない場合はその後、最後に改行が出力されます。

```
Print #StreamNum, [expr[; ...][;]]
```

Input、Line Input、Writeも参照してください。

例

```
Sub Main
A = 1
B = 2
C$ = "Hello"
Open "XXX" For Output As #1
Print #1,A;",";B;",""";C$;"""
Close #1
End Sub
```

Put Instruction

変数の値をStreamNumに書き込みます。

```
Put StreamNum, [RecordNum], var
```

引数	説明
StreamNum	ストリーム1～255は各マクロにプライベートです。ストリーム256～511はすべてのマクロで共有されます。
RecordNum	ランダムモード ファイルの場合、これはレコード番号で、最初のレコードは1です。それ以外の場合はバイト位置で、最初のバイトは1です。これを省略すると、現在の位置(またはレコード番号)を使用します。
var	この変数値はファイルに書き込まれます。固定長変数(Longなど)の場合、変数を格納するために必要なバイト数が書き込まれます。バリアント型変数の場合、その型を説明する2バイトが書き込まれ、それに応じて変数値が書き込まれます。usertype変数の場合、各フィールドは順番に書き込まれます。配列変数の場合、各要素は順番に書き込まれます。動的配列変数の場合、次元数と各次元の範囲は配列値を書き込む前に書き込まれます。すべてのバイナリデータ値は、リトルエンディアン形式でファイルに書き込まれます。 ☞ 文字列(または動的配列)をバイナリモードファイルに書き込む場合、文字列の長さ(または配列の次元)情報は書き込まれません。文字列データまたは配列要素のみが書き込まれます。

ℹ Get、Openも参照してください。

例

```
Sub Main
Dim V As Variant
Open "SAVE_V.DAT" For Binary Access Write As #1
Put #1, , V
Close #1
End Sub
```

QBColor Function

Quick Basicで定義された適切な色コードを返します。

```
QBColor(num)
```

num	色
0	黒色
1	青色
2	緑色
3	青緑色(シアン)
4	赤色
5	赤紫色(マゼンタ)
6	黄色
7	白色
8	灰色
9	水色
10	薄緑色
11	薄水色
12	栗毛色
13	桃色
14	淡黄色
15	明白色

RGB()も参照してください。

例

```
Sub Main
Debug.Print Hex(QBColor(1)) '"800000"
Debug.Print Hex(QBColor(7)) '"C0C0C0"
Debug.Print Hex(QBColor(8)) '"808080"
Debug.Print Hex(QBColor(9)) '"FF0000"
Debug.Print Hex(QBColor(10)) '"FF00"
Debug.Print Hex(QBColor(12)) '"FF"
Debug.Print Hex(QBColor(15)) '"FFFFFF"
End Sub
```

Randomize Instruction

乱数発生器をランダム化します。

```
Randomize [Seed]
```

引数	説明
Seed	この数値は、乱数発生器の初期シードを設定します。この値を省略した場合、現在の時刻がシードとして使用されます。

Rnd()も参照してください。

```
Sub Main
Randomize
Debug.Print Rnd ' 0.?????????????
End Sub
```

ReDim Instruction

動的arrayvarまたはユーザ定義型の配列要素を再次元化します。配列値を保持するには、Preserveを使用します。それ以外の場合、配列値はすべてリセットされます。prepareを使用すると、配列の最後のインデックスのみが変更される可能性がありますが、インデックスの数は変更されない可能性があります。(1次元配列を2次元配列として再次元化することはできません。)

```
ReDim [Preserve] name[type][([dim[, ...]])] [As type][, ...]
```

または

```
ReDim [Preserve] usertypevar.elem[type][([dim[, ...]])] [As type][, ...]
```

Dim、Option Base、Private、Public、Staticも参照してください。

例

```
Sub Main
Dim X()
ReDim X(3)
Debug.Print UBound(X) ' 3
ReDim X(200)
Debug.Print UBound(X) ' 200
End Sub
```

Rem Instruction

下記、どちらの形式もコメントです。Rem形式は命令です。'形式は任意の行の末尾に使用します。'またはRemから行末までのすべてのテキストがコメントの一部です。そのテキストは実行されません。

```
Rem ...
```

または

```
'...
```

例

```
Sub Main
Debug.Print "Hello" ' prints to the output window
' the script terminates at Main's End Sub
End Sub
```

Replace$ Function

S$のPat$をRep$に置き換えます。

```
Replace[$](S$, Pat$, Rep$, [Index], [Count])
```

引数	説明
S$	この文字列値を検索します。置換の対象となる文字列を指定します。
Pat$	この文字列値は、検索パターンです。
Rep$	置換する文字列を指定します。
Index	この数値はS$の開始インデックスです。Replace(S,Pat,Rep,N)は、Replace(Mid(S,N),Pat,Rep)と同等です。省略した場合は1を使用します。
Count	この数値は、実行する置換の最大数です。これを省略した場合は、-1を使用します(すべての出現箇所を置換することを意味します)。

InStr()、InStrRev()、Left$()、Len()、Mid$()、Right$()も参照してください。

例

```
Sub Main
Debug.Print Replace$("abcabc","b","B") '"aBcaBc"
Debug.Print Replace$("abcabc","b","B",,1) '"aBcabc"
Debug.Print Replace$("abcabc","b","B",3) '"caBc"
Debug.Print Replace$("abcabc","b","B",9) '""
End Sub
```

Reset Instruction

現在のマクロ/モジュールで開いているストリームをすべて閉じます。

```
Reset
```

Close、Openも参照してください。

例

```
Sub Main
' read the first line of XXX and print it
Open "XXX" For Input As #1
Line Input #1,L$
Debug.Print L$
Reset
End Sub
```

Resume Instruction

Form 1: labelで実行を再開します。

```
Resume label
```

Form 2: 次のステートメントから実行を再開します。

```
Resume Next
```

エラーが発生すると、エラーハンドラはResumeを使用して実行を続行できます。エラーハンドラは最後にResumeまたはExitを使用する必要があります。

この命令はErrをクリアし、Error$をnullに設定します。

例

```
Sub Main
On Error GoTo X
Err.Raise 1
Debug.Print "RESUMING"
Exit Sub
X: Debug.Print "Err=";Err
Resume Next
End Sub
```

RGB Function

色のRGB値を返します。いくつかの色定数が事前定義されています:
vbBlack - RGB(0,0,0)と同じ
vbRed - RGB(255,0,0)と同じ
vbGreen - RGB(0,255,0)と同じ
vbYellow - RGB(255,255,0)と同じ
vbBlue - RGB(0,0,255)と同じ
vbMagenta - RGB(255,0,255)と同じ
vbCyan - RGB(0,255,255)と同じ
vbWhite - RGB(255,255,255)と同じ

```
RGB(red, green, blue)
```

QBColor()も参照してください。

例

```
Sub Main
Debug.Print Hex(RGB(255,0,0)) '"FF0000"
End Sub
```

Right$ Function

S$の最後からLenで指定した文字を返します。

```
Right[$](S$, Len)
```

同様のRightB関数は、最後のLenのバイト数を返します。

引数	説明
S$	この文字列値の右側の部分を返します。この値がNullの場合は、Nullを返します。
Len	この数の文字を返します。S$がLenより短い場合、S$を返します。

InStr()、InStrRev()、Left$()、Len()、Mid$()、Replace$()も参照してください。

例

```
Sub Main
Debug.Print Right$("Hello",3) '"llo"
End Sub
```

RmDir Instruction

ディレクトリName$を削除します。

```
RmDir Name$
```

引数	説明
Name$	この文字列値は、ディレクトリのパスと名前です。現在のディレクトリからの相対パスを使用できます。

MkDirも参照してください。

例

```
Sub Main
RmDir "C:\WWTEMP"
End Sub
```

Rnd Function

0以上1未満の乱数を返します。

```
Rnd([Num])
```

引数	説明
Num	下記の表を参照してください。

Num	説明
<0	Num をシードとして使用し、毎回同じ数値を返します。
>0	シーケンス内の次の乱数を返します。
0	最後に生成された数値を返します。
省略	シーケンス内の次の乱数を返します。

Randomizeも参照してください。

例

```
Sub Main
Debug.Print Rnd() ' 0.?????????????
End Sub
```

Round Function

指定した小数点以下の桁数に四捨五入した数値を返します。

```
Round([Num][, Places])
```

引数	説明
Num	この数値を四捨五入します。この値がNullの場合、Nullを返します。
Places	指定した小数点以下の桁数に丸めます。これを省略した場合は、最も近い整数値に丸められます。

```
Sub Main
Debug.Print Round(.5) ' 0
Debug.Print Round(.500001) ' 1
Debug.Print Round(1.499999) ' 1
Debug.Print Round(1.5) ' 2
Debug.Print Round(11.11) ' 11
Debug.Print Round(11.11,1) ' 11.1
End Sub
```

RSet Instruction

strvarにstrの値を割り当てます。末尾の文字を除去しstrを短縮(または先頭空白でで延長)します。以前のstrvarの長さが維持されます。

LSetも参照してください。

例

```
Sub Main
  S$ = "123"
  RSet S$ = "A"
  Debug.Print "."+S$+"." '". A."
End Sub
```

RTrim$ Function

S$の末尾のスペースを削除した文字列を返します。

```
RTrim[$](S$)
```

引数	説明
S$	この文字列を末尾のスペースを除いてコピーします。この値がNullの場合、Nullを返します。

LTrim$()、Trim$()も参照してください。

例

```
Sub Main
Debug.Print ".";RTrim$(" x ");"." '". x."
End Sub
```

SaveSetting Instruction

プロジェクトAppNameのSectionにあるKeyのSettingを保存します。Win16およびWin32sは、AppNameという名前の.iniファイルに設定を保存します。Win32は設定を登録データベースに保存します。

```
SaveSetting AppName$, Section$, Key$, Setting
```

引数	説明
AppName$	この文字列値は、このSectionとKeyを持つプロジェクト名です。
Section$	この文字列値は、プロジェクト設定のセクション名です。
Key$	この文字列値は、プロジェクト設定セクションのキー名です。
Setting	キーをこの値に設定します(値は文字列として保存します)。

例

```
Sub Main
SaveSetting "MyApp","Font","Size",10
End Sub
```

Second Function

秒を返します(0～59)。

```
Second(dateexpr)
```

引数	説明
dateexpr	この日付値の秒を返します。この値がNullの場合、Nullを返します。

Hour()、Minute()、Time()も参照してください。

例

```
Sub Main
Debug.Print Second(#12:00:01 AM#) ' 1
End Sub
```

Seek Function

StreamNumの現在位置を返します。ランダム モードファイルの場合、これはレコード番号で、最初のレコードは1です。それ以外の場合はバイト位置で、最初のバイトは1です。

```
Seek(StreamNum)
```

引数	説明
StreamNum	ストリーム1～255は各マクロにプライベートです。ストリーム256～511はすべてのマクロで共有されます。

Seekも参照してください。

例

```
Sub Main
Open "XXX" For Input As #1
Debug.Print Seek(1) ' 1
Line Input #1,L$
Debug.Print Seek(1)
Close #1
End Sub
```

Seek Instruction

入力CountのStreamNumを配置します。

```
Seek [#]StreamNum, Count
```

引数	説明
StreamNum	ストリーム1～255は各マクロにプライベートです。ストリーム256～511はすべてのマクロで共有されます。
Count	ランダムモード ファイルの場合、これはレコード番号で、最初のレコードは1です。それ以外の場合、バイト位置で、最初のバイトは1です。

Seekも参照してください。

例

```
Sub Main
Open "XXX" For Input As #1
Line Input #1,L$
Seek #1,1 ' rewind to start of file
Input #1,A
Close #1
Debug.Print A
End Sub
```

SendKeys Instruction

Keys$をWindowsに送信します。

```
SendKeys Keys$[, Wait]
```

引数	説明
SendKeys	この文字列値のキーをWindowsに送信します(下表参照)。
Wait	これがゼロでない場合、キーは次の命令を実行する前に送信します。これが省略されるかゼロの場合、次の命令中にキーを送信します。

Key	説明
+	Shift修飾キー。
^	Ctrl修飾キー。
%	Alt修飾キー。
~	Enterキーを送信。
k	kキーを送信(kは任意小文字)。
K	Shift kキーを送信(Kは任意の大文字)。
{special n}	特殊キー(nはオプションの繰り返し回数)。
{mouse x,y}	マウスキー(x、yはオプションの画面位置)。
{k}	kキーを送信(kは任意小文字)。
{K}	Shift kキーを送信します(Kは任意の大文字)。
{Cancel}	Breakキーを送信。
{Esc}	Escapeキーを送信。
{Escape	Escapeキーを送信。
{Enter}	Enterキーを送信。
{Menu}	Menuキーを送信(Alt)。
{Help}	Helpキーを送信(?)。
{Prtsc}	Helpキーを送信。
{Print}	送信。
{Execute}	送信?。
{Tab}	送信。
{Pause}	Pauseキーを送信。
{BS}	Back Spaceキーを送信。
{BkSp}	Back Spaceキーを送信。
{BackSpace}	Back Spaceキーを送信。
{Del}	Deleteキーを送信。
{Delete}	Deleteキーを送信。
{Ins}	Insertキーを送信。
{Insert}	Insertキーを送信。
{Left}	Left Arrowキーを送信。
{Right}	Right Arrowキーを送信。
{Up}	Up Arrowキーを送信。
{Down}	Down Arrowキーを送信。
{PgUp}	Page Upキーを送信。
{PgDn}	Page Downキーを送信。
{Home}	Homeキーを送信。
{End}	Endキーを送信。
{Select}	送信?。
{Clear}	Num Pad 5キーを送信。
{Pad0..9}	Num Pad 0から9キーを送信。
{Pad*}	Num Pad *キーを送信。
{Pad+}	Num Pad +キーを送信。
{PadEnter}	Num Pad Enterキーを送信。
{Pad-}	Pad -キーを送信。
{Pad/}	Pad /キーを送信。
{F1..24}	F1からF24キーを送信。

Mouse

マウス動作とクリック:

- {Move x,y} - マウスを(x,y)に移動。
- {ClickLeft x,y} - マウスを(x,y)に移動し、左ボタンをクリック(これは{DownLeft x,y}{UpLeft}と同様)。
- {DoubleClickLeft x,y} - マウスを(x,y)に移動し、左ボタンをクリック(これは、{ClickLeft x,y}{ClickLeft}とは異なります)。
- {DownLeft x,y} - マウスを(x,y)に移動し、左ボタンを押下。
- {UpLeft x,y} - マウスを(x,y)に移動し、左ボタンを開放。
- {...Middle x,y} - マウスの中ボタンと同様の名前のキー。
- {...Right x,y} - マウスの右ボタンの同様の名前のキー。

x、y値は画面のピクセル位置であり、(0,0)は左上隅になります。すべての場合において、x、yはオプションです。省略した場合、以前のマウス位置を使用します。

AppActivate、KeyName、Shell( )も参照してください。

例

```
Sub Main
SendKeys "%S" ' send Alt-S (Search)
SendKeys "GoTo~~" ' send G o T o {Enter} {Enter}
End Sub
```

SetAttr Instruction

ファイルName$の属性を設定します。ファイルが存在しない場合、ランタイムエラーが発生します。

```
SetAttr Name$, Attrib
```

引数	説明
Name$	この文字列値はファイルのパスと名前です。現在のディレクトリからの相対パスを使用できます。
Attrib	ファイルの属性をこの数値に設定します。

例

```
Sub Main
Attrib = GetAttr("XXX")
SetAttr "XXX",1 ' readonly
Debug.Print GetAttr("XXX") ' 1
SetAttr "XXX",Attrib
End Sub
```

Sgn Function

数値の符号を返します。

```
Sgn(Num)
```

引数	説明
Num	この数値の符号を返します。負の場合は-1を返します。ゼロの場合は0を返します。正の場合は1を返します。

Absも参照してください。

例

```
Sub Main
Debug.Print Sgn(9) ' 1
Debug.Print Sgn(0) ' 0
Debug.Print Sgn(-9) '-1
End Sub
```

Shell Function

プログラムName$を実行します。これは、Windowsキー+Rキーから、"ファイル名を指定して実行"と同様です。この命令では、.COM、.EXE、.BAT、および.PIFファイルを実行できます。成功した場合は、タスクIDを返します。

```
Shell(Name$[, WindowType])
```

引数	説明
Name$	この文字列値は、実行するプログラムのパスと名前です。コマンドラインの引数はプログラム名の後に続けて指定します(スペースを含む長いファイル名は二重引用符で囲む必要があります)。
WindowType	アプリケーションのメインウィンドウの表示方法を制御します。以下の表を参照してください。

WindowType	値	効果
bHide	0	ウィンドウを隠す。
bNormalFocus	1, 5, 9	通常のウィンドウ。
bMinimizedFocus	2	ウィンドウを最小化(デフォルト)。
bMaximizedFocus	3	ウィンドウを最大化。
bNormalNoFocus	4, 8	通常の非アクティブ化したウィンドウ。
bMinimizedNoFocus	6,7	最小化した非アクティブ化されたウィンドウ。

AppActivate、SendKeysも参照してください。

例

```
Sub Main
X = Shell("Calc") ' run the calc program
AppActivate X
SendKeys "% R" ' restore calc's main window
SendKeys "30*2{+}10=",1 '70
End Sub
```

Sin Function

サインを返します。

```
Sin(Num)
```

引数	説明
Num	求めるサインの数値。これはラジアン値です。円周は2π radです。

Atn、Cos、Tanも参照してください。

例

```
Sub Main
Debug.Print Sin(1) ' 0.8414709848079
End Sub
```

Space$ Function

Lenで指定したスペース長の文字列を返します。

```
Space[$](Len)
```

引数	説明
Len	このスペース数の長さの文字列を作成します。

String$()も参照してください。

例

```
Sub Main
Debug.Print ".";Space$(3);"." '". ."
End Sub
```

Split Function

Sepで区切った文字列配列から部分文字列を抽出します。

```
Split(Str, [Sep], [Max])
```

引数	説明
Str	この文字列値から部分文字列を抽出します。
Sep	部分文字列を区切るために、この文字列値を検索します(デフォルト: " ")。
Max	最大でこの数の部分文字列を抽出します(デフォルトは-1。見つかった数だけ作成することを意味します)。

Join()も参照してください。

例

```
Sub Main
Debug.Print Split("1 2 3")(1) '"2"
End Sub
```

Sqr Function

平方根を返します。

```
Sqr(Num)
```

引数	説明
Num	この数値の平方根を返します。

例

```
Sub Main
Debug.Print Sqr(9) ' 3
End Sub
```

Stop Instruction

実行を一時停止します。実行が再開された場合、次の命令から開始します。スクリプトの実行を完全に終了するには、End を使用します。

```
Stop
```

引数	説明
Len	このスペース数の長さの文字列を作成します。

String$()も参照してください。

例

```
Sub Main
For I = 1 To 10
Debug.Print I
If I = 3 Then Stop
Next I
End Sub
```

Str$ Function

Numを文字列で返します。

```
Str[$](Num)
```

引数	説明
Num	この数値の文字列表現を返します。正の値は空白で始まります。負の値はダッシュ"-"で始まります。

CStr()、Hex$()、Oct$()、Val()も参照してください。

例

```
Sub Main
Debug.Print Str$(9*9) ' 81
End Sub
```

StrComp$ Function

2つの文字列を比較します。

```
StrComp(Str1,Str2,Comp)
```

引数	説明
Str1	この文字列をStr2と比較します。この値がNullの場合、Nullを返します。
Str2	この文字列をStr1と比較します。この値がNullの場合、Nullを返します。
Comp	この数値は比較の種類を示します。以下の比較表を参照してください。

結果	説明
-1	Str1はStr2より小さい。
0	Str1はStr2に等しい。
1	Str1はStr2より大きい。
Null	Str1またはStr2がNullです。

Comp	値	効果
bUseCompareOption	-1	Option Compareステートメントの値を使用して比較を実行します。
bBinaryCompare	0	文字列のバイナリデータを比較します。
bTextCompare	1	照合規則を使用して文字列のテキストを比較します。
bDatabaseCompare	2	Microsoft Accessのみ(サポートしていません)。

LCase$( )、Option Compare、StrConv$( )、UCase$( )も参照してください。

例

```
Sub Main
Debug.Print StrComp("F","e") ' -1
Debug.Print StrComp("F","e",1) ' 1
Debug.Print StrComp("F","f",1) ' 0
End Sub
```

StrConv$ Function

文字列を変換します。

```
StrConv[$](Str,Conv)
```

引数	説明
Str	この文字列値を変換します。この値がNullの場合、Nullを返します。
Conv	この数値は変換の種類を示します。以下の変換表を参照してください。

Conv	値	効果
bUpperCase	1	大文字に変換します。
bLowerCase	2	小文字に変換します。
bProperCase	3	最初の文字を大文字に変換し、残りの文字を小文字に変換(サポートしていません)。
bWide	4	全角文字に変換します(東アジアロケールのWin32でのみサポートしています)。
bNarrow	8	半角文字に変換します(東アジアロケールのWin32でのみサポートしています)。
bKatakana	16	カタカナに変換します(日本語ロケールのWin32でのみサポートしています)。
bHiragana	32	ひらがなに変換します(日本語ロケールのWin32でのみサポートしています)。
bUnicode	64	Unicodeに変換します(Win32でのみサポートしています)。
bFromUnicode	128	Unicodeから変換します(Win32でのみサポートしています)。

LCase$()、StrComp()、UCase$()も参照してください。

例

```
Sub Main
Dim B(1 To 3) As Byte
B(1) = 65
B(2) = 66
B(3) = 67
Debug.Print StrConv$(B,vbUnicode) '"ABC"
End Sub
```

String$ Function

CharまたはChar$の最初の文字をLenで指定した文字数で埋める。

```
String[$](Len, Char|$)
```

引数	説明
Len	この文字数の文字列を作成します。
Char|$	この値を文字列に入力します。これが数値の場合、相当するASCII文字を使用します。これが文字列の場合、その文字列の最初の文字を使用します。この値がNullの場合、Nullを返します。

Space$()も参照してください。

例

```
Sub Main
Debug.Print String$(4,65) '"AAAA"
Debug.Print String$(,"ABC") '"AAAA"
End Sub
```

StrReverse$ Function

文字を逆順にした文字列を返します。

```
StrReverse[$](S)
```

引数	説明
S	この文字列を逆順の文字で返します。

例

```
Sub Main
Debug.Print StrReverse$("ABC") 'CBA
End Sub
```

Tan Function

タンジェントを返します。

```
Tan(Num)
```

引数	説明
Num	この数値のタンジェントを返します。

Atn、Cos、Sinも参照してください。

例

```
Sub Main
Debug.Print Tan(1) '1.5574077246549
End Sub
```

Time Function

現在の時刻を日付値として返します。

```
Time[$]
```

Date、Now、Timerも参照してください。

例

```
Sub Main
Debug.Print Time 'example: 09:45:00 am
End Sub
```

Timer Function

午前0時から過ぎた秒数を返します(これは実数であり、約1/18 秒の精度です)。

```
Timer
```

Date、Now、Timeも参照してください。

例

```
Sub Main
Debug.Print Timer 'example: 45188.13
End Sub
```

TimeSerial Function

日付値を返します。

```
TimeSerial(Hour, Minute, Second)
```

引数	説明
Hour	この数値は時間(0～23)です。
Minute	この数値は分(0～59)です。
Second	この数値は秒(0～59)です。

DateSerial、DateValue、TimeValueも参照してください。

例

```
Sub Main
Debug.Print TimeSerial(13,30,0) '1:30:00 PM
End Sub
```

TimeValue Function

日付の時刻部分を文字列値にエンコードして返します。

```
TimeValue(Date$)
```

引数	説明
Date$	この文字列値を、対応する日付の時刻部分に変換します。

DateSerial、DateValue、TimeSerialも参照してください。

例

```
Sub Main
Debug.Print TimeValue("1/1/2000 12:00:01 AM") '12:00:01 AM
End Sub
```

Trim$ Function

S$の先頭と末尾のスペースを削除した文字列を返します。

```
Trim[$](S$)
```

引数	説明
S$	この文字列を先頭または末尾のスペースを削除します。この値がNullの場合、Nullを返します。

例

```
Sub Main
Debug.Print ".";Trim$(" x ");"." '".x."
End Sub
```

TypeName Function

varに格納されている値の型を示す文字列を返します。

```
TypeName[$](var)
```

引数	説明
var	この変数に格納されている値の型を示す文字列を返します。

結果	説明
Empty	Variant変数は空。値が割り当てられたことはありません。
Null	Variant変数はnull。
Integer	変数にはInteger値が含まれています。
Long	変数にはLong値が含まれています。
Single	変数にはSingle値が含まれています。
Double	変数にはDouble値が含まれています。
Currency	変数にはCurrency値が含まれています。
Date	変数にはDate値が含まれています。
String	変数にはString値が含まれています。
Object	変数にはNothing以外のObjectが含まれています(オブジェクトの種類に固有の型を返す場合があります)。
Nothing	変数にはNothingのオブジェクト参照が含まれています。
Error	変数にはエラーコード値が含まれています。
Boolean	変数にはBoolean値が含まれています。
Variant	変数にはVariant値が含まれています(Variantの配列にのみ使用されます)。
Unknown	変数にはActiveXオートメーション以外のオブジェクト参照が含まれています。
Byte	変数にはByteのオブジェクト参照が含まれています。
( )	変数には配列値が含まれます。要素のTypeNameの後に()が続きます。

VarTypeも参照してください。

例

```
Sub Main
Dim X As Variant
Debug.Print TypeName(X) '"Empty"
X = 1
Debug.Print TypeName(X) '"Integer"
X = 100000
Debug.Print TypeName(X) '"Long"
X = 1.1
Debug.Print TypeName(X) '"Double"
X = "A"
Debug.Print TypeName(X) '"String"
Set X = CreateObject("Word.Basic")
Debug.Print TypeName(X) '"Object"
X = Array(0,1,2)
Debug.Print TypeName(X) '"Variant()"
End Sub
```

UBound Function

最大のインデックスを返します。

```
UBound(arrayvar[, dimension])
```

引数	説明
arrayvar	この配列変数の最大インデックスを返します。
dimension	*arrayvar*のこの次元の最大インデックスを返します。これを省略した場合は、最初の次元の最大インデックスを返します。

LBound()も参照してください。

例

```
Sub Main
Dim A(3,6)
Debug.Print UBound(A) '3
Debug.Print UBound(A,1) '3
Debug.Print UBound(A,2) '6
End Sub
```

UCase$ Function

すべての小文字を大文字に変換したS$の文字列を返します。

```
UCase[$](S$)
```

引数	説明
S$	すべての文字を大文字に変換した後、この文字列値を返します。この値がNullの場合は、Nullを返します。

StrComp( )、StrConv$( )、LCase$( )、も参照してください。

例

```
Sub Main
Debug.Print UCase$("Hello") '"HELLO"
End Sub
```

Unlock Instruction

Form 1: StreamNumのすべてのロックを解除します。

```
Unlock StreamNum
```

Form 2: StreamNumのレコード(またはバイト)のロックを解除します。

```
Unlock StreamNum, RecordNum
```

Form 3: StreamNumのレコード(またはバイト)の範囲のロックを解除します。startが省略された場合は、最初のレコード(またはバイト)からロックが解除されます。

```
Unlock StreamNum, [start] To end
```

☞ シーケンシャルファイル(入力、出力、追加)の場合、ロック解除は常にファイル全体に影響します。

引数	説明
StreamNum	ストリーム1から255は各マクロに対してプライベートです。ストリーム256から511はすべてのマクロで共有されます。
RecordNum	Randomモードファイルの場合、これはレコード番号です。最初のレコードは1です。それ以外の場合はバイト位置です。最初のバイトは1です。
start	範囲内の最初のレコード(またはバイト)。
end	範囲内の最後のレコード(またはバイト)。

ℹ Open、Lockも参照してください。

例

```
Sub Main
Dim V As Variant
Open "SAVE_V.DAT" For Binary As #1
Lock #1
Get #1, 1, V
V = "Hello"
Put #1, 1, V
Unlock #1
Close #1
End Sub
```

Val Function

S$の値を返します。

```
Val(S$)
```

引数	説明
S$	文字列に数値を返します。&Oで始まる文字列は8進数です。&Hで始まる文字列は16進数です。それ以外は10進数です。

例

```
Sub Main
Debug.Print Val("-1000") ' -1000
End Sub
```

VarType Function

varに格納されている値の型を示す数値を返します。

```
VarType(var)
```

引数	説明
var	この変数に格納されている値のタイプを示す数値を返します。

結果	値	説明
bEmpty	0	Variant変数は空。値が割り当てられたことはありません。
bNull	1	Variant変数はnull。
bInteger	2	変数にはInteger値が含まれています。
bLong	3	変数にはLong値が含まれています。
bSingle	4	変数にはSingle値が含まれています。
bDouble	5	変数にはDouble値が含まれています。
bCurrency	6	変数にはCurrency値が含まれています。
bDate	7	変数にはDate値が含まれています。
bString	8	変数にはString値が含まれています。
Object	9	変数にはNothing以外のObjectが含まれています(オブジェクトの種類に固有の型を返す場合があります)。
bNothing	10	変数にはNothingのオブジェクト参照が含まれています。
bError	11	変数にはエラーコード値が含まれています。
bBoolean	12	変数にはBoolean値が含まれています。
bVariant	13	変数にはVariant値が含まれています(Variantの配列にのみ使用されます)。
bUnknown	14	変数にはActiveXオートメーション以外のオブジェクト参照が含まれています。
bByte	17	変数にはByteのオブジェクト参照が含まれています。
bUserDefinedType	36	変数にはユーザ定義タイプの値が含まれています。
+vbArray	8192	変数には配列値が含まれます。VarType( )と255を使用して、配列に格納されている要素の型を取得します。

TypeNameも参照してください。

例

```
Sub Main
Dim X As Variant
Debug.Print VarType(X) '0
X = 1
Debug.Print VarType(X) '2
X = 100000
Debug.Print VarType(X) '3
X = 1.1
Debug.Print VarType(X) '5
X = "A"
Debug.Print VarType(X) '8
Set X = CreateObject("Word.Basic")
Debug.Print VarType(X) '9
X = Array(0,1,2)
Debug.Print VarType(X) '8204 (8192+12)
End Sub
```

Wait Instruction

Delay秒間待ちます。

```
Wait Delay
```

例

```
Sub Main
Wait 5 ' wait for 5 seconds
End Sub
```

Weekday Function

曜日を返します。

```
Weekday(dateexpr)
```

- vbSunday (1) - 日曜日
- vbMonday (2) - 月曜日
- vbTuesday (3) - 火曜日
- vbWednesday (4) - 水曜日
- vbThursday (5) - 木曜日
- vbFriday (6) - 金曜日
- vbSaturday (7) - 土曜日

引数	説明
dateexpr	この日付値の曜日を返します。この値がNullの場合は、Nullを返します。

Date( )、Day( )、Month( )、WeekdayName( )、Year( )も参照してください。

例

```
Sub Main
Debug.Print Weekday(#1/1/1900#) ' 2
Debug.Print Weekday(#1/1/2000#) ' 7
End Sub
```

WeekdayName Function

曜日名を返します。

```
WeekdayName(NumZ{day}[, CondZ{abbrev}])
```

引数	説明
day	この曜日のローカライズされた名前を返します(1から7)。
abbrev	この条件値がTrueの場合、曜日名の短縮形を返します。

Weekday( )も参照してください。

例

```
Sub Main
Debug.Print WeekdayName(1) 'Sunday
Debug.Print WeekdayName(Weekday(Now))
End Sub
```

Write Instruction

exprをStreamNumに書き込みます。

```
Write #StreamNum, expr[, ...]
```

文字列値は引用符で囲まれます。Null値は#NULL#として記述されます。ブール値は#FALSE#または#TRUE#として記述されます。日付値は#date#として記述されます。エラーコードは#ERROR番号#と記述されます。。

Input、Line Input、Printも参照してください。

例

```
Sub Main
A = 1
B = 2
C$ = "Hello"
Open "XXX" For Output As #1
Write #1,A,B,C$
Close #1
End Sub
```

Year Function

年を返します。

```
Year(dateexpr)
```

引数	説明
dateexpr	この日付値の年を返します。この値がNullの場合は、Nullを返します。

Date()、Day()、Month()、WeekdayName()、Weekday()も参照してください。

例

```
Sub Main
Debug.Print Year(#1/1/1900#) ' 1900
Debug.Print Year(#1/1/2000#) ' 2000
End Sub
```

VBAエラーリスト

次の表に、すべてのエラー コードと関連するエラーテキストを示します。

エラー番号	説明
10000	実行が中断されました。
10001	メモリ不足です。
10008	'#Uses "module"コメントが無効です。
10009	'#Uses "module"依存関係が無効です。
10010	マクロがすでに実行中です。
10011	マクロ/モジュールにメモリを割り当てることができません。
10012	マクロ/モジュールに構文エラーがあります。
10013	マクロ/モジュールが存在しません。
10014	別のマクロが一時停止されているため、現時点では続行できません。
10017	現在有効なマクロはありません。
10018	Sub/Functionが存在しません。
10019	パラメータの数が正しくありません。
10021	大きな配列を割り当てられません。
10022	配列に次元が設定されていません。
10023	配列のインデックスが範囲外です。
10024	配列の下限が上限よりも大きくなっています。
10025	配列のインデックスの数が異なります。
10030	ユーザダイアログが定義されていません。
10031	ユーザがキャンセルしました。
10032	ユーザダイアログ項目IDが範囲外です。
10033	現在、UserDialogは表示されていません。
10034	現在、UserDialogにアクセスできません。
10035	Withが間違っています。Withブロックの外側または内側にGoToを使用しないでください。
10040	モジュールを読み込めませんでした。
10041	モジュール内に関数が見つかりません。
10048	読み取り権限でファイルを開けません。
10049	書き込み権限でファイルを開けません。
10050	レコード長を超えています。
10051	ファイルを開けませんでした。
10052	ファイルを開いていません。
10053	ファイルの終わりを超えて読み取ろうとしています。
10054	ストリーム番号は1から511の範囲が必要です。
10055	入力がvar型と一致しません。
10056	1から32767の範囲の長さが必要です。
10057	ストリーム番号はすでに開いています。
10058	この操作に対して間違ったモードでファイルを開きました。
10059	ファイル操作中にエラーが発生しました。
10060	式に無効な浮動小数点演算が含まれています。
10061	ゼロで除算。
10062	オーバーフロー。
10063	式が最小表現をアンダーフローしました。
10064	式表現の精度の損失。
10069	文字列値は有効な数値ではありません。
10071	ResumeはOn Errorハンドラでのみ使用できます。
10075	ここではNull値を使用できません。

エラー番号	説明
10080	型が一致しません。
10081	パラメータ#1の型が一致しません。
10082	パラメータ#2の型が一致しません。
10083	パラメータ#3の型が一致しません。
10084	パラメータ#4の型が一致しません。
10085	パラメータ#5の型が一致しません。
10086	パラメータ#6の型が一致しません。
10087	パラメータ#7の型が一致しません。
10088	パラメータ#8の型が一致しません。
10089	パラメータ#9の型が一致しません。
10090	OLEオートメーションエラー。
10091	OLEオートメーション: そのようなプロパティやメソッドはありません。
10092	OLEオートメーション: サーバはオブジェクトを作成できません。
10093	OLEオートメーション: サーバーはファイルを読み込めません。
10094	OLEオートメーション: オブジェクト変数は'Nothing'です。
10095	OLEオートメーション: サーバが見つかりませんでした。
10096	OLEオートメーション: 現在アクティブなオブジェクトはありません。
10097	OLEオートメーション: パラメータ数が間違っています。
10098	OLEオートメーション: インデックスが正しくありません。
10099	OLEオートメーション: そのような名前のパラメータはありません。
10100	ディレクトリが見つかりません。
10101	ファイルを削除できません。
10102	ディレクトリを作成できません。
10103	ファイル名の変更ができません。
10104	ディレクトリを削除できません。
10105	ドライブが見つかりません。
10106	ソースファイルを開けません。
10107	変更先ファイルを作成できません。
10108	ソースファイルを完全に読み込むことができませんでした。
10109	変更先ファイルを完全に書き込むことができませんでした。
10110	右中括弧"}"がありません。
10111	無効なキー名です。
10112	右括弧")"がありません。
10113	右大括弧"]"がありません。
10114	カンマ","がありません。
10115	セミコロン";"がありません。
10116	SendKeysはWindowsジャーナル再生フックをインストールできませんでした。
10119	文字列が長すぎます(キーが多すぎます)。
10120	ウィンドウが見つかりませんでした。
10130	DDEは使用できません。
10131	同時に行われるDDE会話が多すぎます。
10132	無効なチャネル番号です。
10133	DDE操作が時間内に完了しませんでした。
10134	DDEサーバが停止しました。
10135	DDEの操作に失敗しました。
10140	クリップボードにアクセスできません。
10150	シェルが失敗しました。
10151	シェルが失敗しました。
10160	ウィンドウスタイルは1から9の範囲内でなければなりません。
10200	Basicは回復不能なエラー状態のため停止しています。
10201	Basicはビジー状態のため、要求されたサービスを提供できません。
10202	Basicの呼び出しに失敗しました。
10203	ハンドラプロパティ: プロトタイプの指定が不正です。
10204	ハンドラはすでに使用されています。

ヒントとコツ

その方法は?

catmanEasy/AP標準ユーザインタフェース要素(リボンタブ、メニューなど)を非表示にするには?

catmanメインメニューの"Customize user interface"ダイアログを開きます。"System"設定では、個々のリボンタブまたはリボン全体を非表示にすることができます。システムメニューやステータスパネルを非表示にすることもできます。次に、設定をファイルに保存します。スクリプトの先頭で、EA_Ribbon.LoadExtensionsメソッドを使用してこのファイルをロードします。

ユーザインタフェースの作成を参照してください。

ユーザがマウスをクリックしてスクリプトのプロシージャを実行するには?

バージョン2.5以降、catmanEasy/APメインのリボンバーを独自のコマンドボタンで拡張できます。

catmanメインメニューの"Customize user interface"ダイアログを開きます。グループ内に独自のリボンタブ、グループ、ボタン(いわゆるコントロール)を追加できるようになりました。次に、スクリプトプロシージャをコントロールに割り当てます。次に、設定をファイルに保存します。スクリプトの先頭で、EA_Ribbon.LoadExtensionsメソッドを使用してこのファイルをロードします。

コントロールの作成後でもスクリプトプロシージャを割り当てることができます。コントロールの作成前にスクリプトプロシージャが存在することは必須ではありません。リボンタブをEasyScriptに変更し、コントロール上でマウスの右ボタンをクリックします。コンテキストメニューから"Event procedure OnClick"を選択します。catmanEasyは、プロシージャがまだ存在しない場合は作成するか、既存のプロシージャを表示します。

もちろん、コマンドボタンをパネルに直接配置することも引き続きサポートされています。ただし、ツール チップやショートカットキーなどの高度な機能を備えているのはリボンバー コントロールのみであることに注意してください。

ユーザインタフェースの作成を参照してください。

ユーザ入力の取得方法は?

バージョン2.5以降、catmanEasy/APのメインのリボンバーを独自のテキストボックス、チェックボックス、またはリストボックスで拡張できます。catmanメインメニューの"Customize user interface"ダイアログを開きます。独自のリボンタブ、グループ、ボタンおよびコントロールをグループ内に追加できるようになりました。次に、コントロールごとに"control type"を設定します。

最後に設定をファイルに保存します。 スクリプトの先頭で、EA_Ribbon.LoadExtensionsメソッドを使用してこのファイルをロードします。

スクリプトの実行時に、EA_Ribbonオブジェクトのメソッドを使用してユーザ入力を取得します。

ユーザインタフェースの作成を参照してください。

テキストボックス、チェックボックス、またはコンボボックスをパネルに直接配置し、*EA_Panel*オブジェクトのメソッドを使用してアクセスすることは、もちろん引き続きサポートされます。

サブメニュー(ドロップダウンメニュー)を作成するには?

リボングループ内の任意の場所に新しいコントロールを作成します。次に、このコントロールの"Allow subcontrols"(サブコントロールを許可)オプションを有効にします。その後、"New subcontrol"をクリックして最初のサブメニューコントロールを挿入します。これにより、最初のサブメニューコントロールが作成されます。次に、このコントロールを選択し、"New control below"をクリックしてさらにサブメニュー コントロールを作成します。次に、このコントロールを選択し、"New control"を使用して同じレベルにサブコントロールを追加します。ドロップダウンコントロールには最大2つのサブレベルを含めることができます。

サブメニューコントロールの型は、テキストボックスまたはリストボックスであってはなりません。

単一のスクリプトプロシージャを実行するには?

クイックアクセスツールバーを使用します(**Cockpit** -> **EasyScript** -> **Quick Access toolbar**)。このツールバーには、現在ロードされているプロジェクトで使用可能なすべてのプロシージャがリストされます。

1. **EasyScript** -> **Load script project**を使用してスクリプトプロジェクトをロードします。または、 。
2. EasyScript -> **Start script project**でスクリプトプロジェクトを開始します。または、 。
3. 必要な手順を選択し、 をクリックします。

このボタンは、スクリプトが開始されている場合にのみアクティブになります。

ボタンにスクリプトのプロシージャを割り当てるには?

EasyScriptエディタでスクリプトプロジェクトを読み込むか、catmanEasy/APのメインウィンドウで**EasyScript -> Load script project**からスクリプトのプロジェクトをを読み込みます。次に、ボタンの設定ダイアログを開き、**EasyScript procedure to execute**(実行するEasyScriptプロシージャ)のリストから必要なプロシージャを選択します。
プロシージャがまだ存在しない場合、ボタンのコンテキスト メニューから**Event procedure OnClick**...を選択します。catmanEasy/APは、EasyScriptエディタの現在のコードモジュールにそのようなプロシージャを作成します。

Note: バージョン2.5以降では、独自のコマンドボタン(コントロール)を使用してリボンバーを拡張できます。これは、パネル上のコマンドボタンを使用するよりも強力かつ柔軟です。

ユーザインタフェースの作成を参照してください。

ボタンに割り当てられているスクリプトのプロシージャを実行するには?

個々のプロシージャは、スクリプトが開始されている場合にのみ開始できます。**EasyScript** -> **Start script project**またはQuick Accessツールバーを使用します。

スクリプトを開始した直後から、対話型パネルオブジェクト(ボタン、テキストボックス、リストボックス、およびチェックボックス)を編集したり移動したりすることはできなくなります。

Note: バージョン2.5以降では、独自のコマンドボタン(コントロール)を使用してリボンバーを拡張できます。これは、パネル上のコマンドボタンを使用するよりも強力かつ柔軟です(参照: ユーザインタフェースの作成)。

特定の時点、例えばDAQ ジョブの開始後などにスクリプトのプロシージャを自動的に実行するには?

Auto-script executionダイアログで、プロシージャを特定の時点(実行ステージとも呼ばれます)に割り当てます(**EasyScript** → **Configure Auto-script execution**)。

この手順は、スクリプトが開始されている場合にのみ実行されます (**EasyScript** -> **Start script project**)。

catmanEasy/APの起動時にスクリプトのプロジェクトを自動的に実行するには?

次のコマンドライン オプションを使用します:

```
/script: Project file
```

例えば:

```
c:\programs\hbm\catmanEasy/AP\catmanEasy/AP.exe /script: c:\programs\hbm\
catmanEasy/AP\projects\myscript.esp
```

スクリプトでcatmanEasy/APを起動を参照してください。

	catmanEasy/APプロジェクトの読み込み時にスクリプトのプロジェクトを自動的に実行するには? **Run script automatically**(スクリプトを自動的に実行)オプションを有効にします(**メインメニュー** -> **Options**、**EasyScript**タブ、**General**セクション)。
	DAQジョブを実行せずに、チャネルの現在の計測値を読み出すには? EA_IO.Measureを使用します。
	テストからチャンネルをグラフィカルに表示するには? EA_Test.Loadからテストを読み込みます。次に、EA_Test.PlotYTやEA_Test.PlotYXを使用して、テストからのチャネルをグラフに表示します。
	ユーザ入力をクエリするには? 2つの方法があります: 1. バージョン2.5以降、catmanEasy/APのメインのリボンバーを独自のテキストボックス、チェックボックス、またはリストボックスで拡張し、EA_Ribbon.GetValueを介してユーザ入力をクエリできます。 ユーザインタフェースの作成を参照してください。 2. テキストボックスやチェックボックスなどの対話型入力オブジェクトをパネルに配置し、EA_Panel.GetValueを介して内容をクエリします。
	メッセージの表示やユーザに確認を求めるには? 簡単なメッセージと確認には、VBAのMsgBox関数を使用します。メッセージボックスの外観をcatmanEasy/APのメッセージボックスと同じにする場合は、EA.MessageBoxを使用します。
	Microsoft ExcelまたはWordから関数を呼び出すには? CreateObject(遅延バインディング)を使用してこれらのアプリケーションのオブジェクトを生成するか、このオブジェクトへの参照を定義します(スクリプト エディタで**EasyScript** -> **Project** -> **Object references**)。 他のプログラムをオブジェクトとして作成して使用およびMicrosoft Officeの使用を参照してください。
	DLL(ダイナミックリンクライブラリ)から関数を呼び出すには?たとえばWindowsAPIなど Declare命令を使用して目的の関数を宣言します。 DLL関数の呼び出しを参照してください。
	DAQジョブを実行せず、チャネルの一時計測ストレージにデータを書き込むには? EA_DAQ.SetTempStoreを使用します。
	テストを保存せず、チャネルの一時計測ストレージからデータを読み出すには? Use EA_DAQ.GetTempStoreを使用します。
	catmanEasy/APがDAQモードのときにテストを読み込むには? テストは解析モードでのみ読み込めます。EA.OperationModeでcatmanEasy/APを解析モードに変更します。解析の終了後にDAQモードに戻す場合にも、この方法を使用します。
	自動的に解析モードに変更して、最後のDAQジョブのデータを読み出すには? 最後に測定したデータを保存します。次に、EA.OperationModeで操作モードを変更し、パラメータとして空の文字列指定してEA_Test.Loadを使用します: `EA_Test.Load ""`
	テキスト(ASCII)データをファイルに書き込むには? 1. VBA関数FreeFileを使用してファイルハンドル(*stream*)を取得します。 2. VBA関数Openを使用してファイルを開きます。 3. VBA関数Printを使用してデータを書き込みます。 4. VBA関数Closeを使用してファイルを閉じます。
	ファイルからテキスト(ASCII)データを読み出すには? 1. VBA関数FreeFileを使用してファイルハンドル(*stream*)を取得します。 2. VBA関数Openを使用してファイルを開きます。 3. VBA関数のInputまたはLine Inputを使用してデータを読み取ります。 4. VBA関数Closeを使用してファイルを閉じます。

	バイナリデータをファイルに書き込むには? 1. VBA関数FreeFileを使用してファイルハンドル(*stream*)を取得します。 2. VBA関数Openを使用してファイルを開きます。 3. VBA関数Putを使用してデータを書き込みます。 4. VBA関数Closeを使用してファイルを閉じます。
	ファイルからバイナリデータを読み出すには? 1. VBA関数FreeFileを使用してファイルハンドル(*stream*)を取得します。 2. VBA関数Openを使用してファイルを開きます。 3. VBA関数のGetを使用してデータを読み取ります。 4. VBA関数Closeを使用してファイルを閉じます。
	ユーザにファイルを選択させるには? VBA関数GetFilePathを使用します。これにより、ユーザはフォルダを参照して既存のファイルを選択したり、新しいファイルの名前とパスを指定したりできます。
	新規フォルダ(ディレクトリ)を作成するには? 最も簡単な解決策は、VBA関数MkDirを使用することです。 FileSystemオブジェクトは、より高度な機能を提供します。
	フォルダ(ディレクトリ)を削除するには? 最も簡単な解決策は、VBA関数RmDirを使用することです。 FileSystemオブジェクトは、より高度な機能を提供します。
	ファイル属性にアクセスするには? FileSystemオブジェクトを作成し、そこから Fileのサブオブジェクトを作成します。

	Windowsレジストリにアクセスするには?
	VBA関数GetSetting(読み取り)およびSaveSetting(書き込み)を使用します。
	
	例:
	Directoriesタブの**Options**で指定したDAQおよびテストデータのフォルダを D:\MyTestsに変更する場合は、次のようになります。
	`SaveSetting "catmanEasy/AP", "DEFAULTS", "DAQDATADIR", "D:\MyTests"`
	
	セクション(DEFAULTS)またはキー(DAQDATADIR)が存在しない場合は作成します。同様の方法で値を読み取ります:
	`DAQDirectory = GetSetting("catmanEasy/AP", "DEFAULTS", "DAQDATADIR")`
	
	キーが存在しない場合に備えて、デフォルト値(D:\MyNewDir)を指定します:
	`DAQDirectory = GetSetting("catmanEasy/AP", "DEFAULTS", "DAQDATADIR" "D:\MyNewDir")`

	多くの入力フィールドを作成できますか? 多数のテキストボックスを使用してこれらを作成するよりも簡単な方法はありますか? 解析モードではテーブルを使用し、セルタイプを**Text (editable)**(編集可能)に設定できます(**Advanced**セクション)。 EA_Panel.GetCellを使用して、そのようなセルの内容をクエリします。
	フォーマットされた数値を解析モードで表のセルに書き込むには? Format$(number, format string)を使用して数値をテキストに変換し、書式文字列で小数点以下の桁数を指定します。次に、テーブルのセルタイプとして**Text (static)**(編集不可) を使用します(**Advanced**セクション)。

"Allow catman base services"オプションの意味は何ですか?

catmanEasy/APは、HBKのデータ収集パッケージcatman ProfessionalのActiveXインタフェースを介して動作するため、catman DLLが提供する関数も使用できます。このオプションをアクティブにすると、オブジェクトcatmanが作成され、スクリプトで使用できるようになります。このオブジェクトを使用して、catmanのActiveXインタフェースにアクセスできます。

通常、このインタフェースにアクセスする必要はありません。ただし、catScriptコマンドの実行など、catmanコアサービスが役立つことがまれにある場合があります。いずれの場合でも、この機能を使用するには、catmanとその動作原理についての十分な知識が必要です。不適切に使用すると、プログラムがクラッシュする可能性があります。

catmanのActiveXインタフェースのドキュメントは、catman Professional 4.5以降のインストールフォルダのActiveXサブディレクトリにあります。

トラブルシューティング

EasyScriptメニューが使用できません。
EasyScript を使用できるようにするには、HBKからEasyScriptのオプション(モジュール) を購入する必要があります。HBKから取得したライセンスを使用して、スクリプトモジュールを有効にします: **File**メニュー -> **Options** -> **Program functions**タブ。

スクリプトプロジェクトを開始すると、"Invalid instruction..."(無効な命令です)というエラー メッセージが表示されます。
別のモジュールのプロシージャを呼び出そうとしていますが、そのプロシージャは呼び出し側モジュールで認識されません。
'#usesディレクティブを使用して、プロシージャを含むモジュールを宣言します。

ボタン経由でプロシージャを呼び出すとき、またはスクリプトの自動実行中に"Invalid instruction..."(無効な命令です)というエラー メッセージが表示されます。
呼び出されたプロシージャが不明です。これにはいくつかの理由があります:
- 現在読み込まれているスクリプトプロジェクトにプロシージャが存在しません。
- プロシージャは`Private`として宣言しています。

ボタンにプロシージャを割り当てましたが、ボタンをクリックしても何も起こりません。
ボタンからプロシージャを実行できるようにするには、スクリプトを開始する必要があります。EasyScriptプロジェクトの実行を参照してください。

クイックアクセスツールバーからプロシージャを実行したいのですが、対応するボタンがグレー表示になっています。
プロシージャを実行するには、スクリプトを開始する必要があります。EasyScriptプロジェクトの実行を参照してください。

Mainプロシージャのコードは終了しましたが、スクリプトが終了しません。
*Main*プロシージャのコードが完全に処理された場合でも、catmanEasy/APは実行中のスクリプトを自動的に終了しません。代わりに、catmanEasy/APはボタンによって発行されるプロシージャ呼び出しまたは自動スクリプトの実行を待ちます。
EasyScriptメニューまたはクイックアクセス ツールバーを使用するか、EA.Terminateを使用してスクリプトを終了する必要があります。

エディタでモジュールのコードを編集したいのですが、必ず"Would you like to put the script into the edit status"(スクリプトを編集状態にしますか?)というメッセージが表示されます。
エディタ(コンテキストメニュー)でモジュールを閉じ、プロジェクト ツリー(エディタウィンドウの右側にある現在のEasyScriptプロジェクトのツリービュー)をクリックしてモジュールを再度開きます。

ボタン、リストボックス、またはチェックボックスタイプのパネルオブジェクトを移動、サイズ変更、構成することができません。
スクリプトを実行している場合、これらのオブジェクトを構成または移動することはできません。メニューの**EasyScript** -> **Stop script execution**、またはクイックアクセスツールバー( )を使用して、スクリプトの実行を停止します。

ユーザが"Cancel"でユーザダイアログを中断すると、エラーメッセージが表示されます。

*Cancel*ボタンを押すと、常に実行時エラーが発生します。メッセージを抑制するには、On Errorを使用します:

```
Sub MyDialog
  Dim Canceled
  On Error GoTo CheckError
  ...
  Dialog dlg   'Show dialog
  If Canceled Then
  ...
  End If
Exit Sub

CheckError:
If Err.Number = 10031 Then
  Canceled = True
  Resume Next
Else
  ...
End If
End Sub
```

Advancing Automation
ADVAN AUTO MATION
日本語訳:　アドバン オートメーション株式会社
info-advan@adv-auto.co.jp　　www.adv-auto.co.jp