Simple Measure

Version 1.1

Excel で自動測定

取扱説明書

Automatic Measurement Systems

URL: http://www.geocities.jp/automeasuresystem/

第 1.00 版

目次

ソフトウェアのご使用条件

1. 使用権
お客様は、許諾プログラムを１ライセンス当たり、１台のパーソナルコンピュータまたは仮想化コンピュータなどの同等の装置に１つだけインストールできます。
リモート接続されるコンピュータやサーバーコンピュータなどが複数のクライアントからネットワーク等を経由してアクセスされる場合は、接続可能なクライアントの数だけ追加ライセンスが必要となります。

2. 逆コンパイル等
お客様は、許諾プログラムを逆コンパイルまたは逆アセンブルなどのリバースエンジニアリングを行うことはできません。

3. 著作権その他の知的財産権
許諾プログラムおよび関連資料等に関する著作権その他の知的財産権は、Automatic Measurement Systems に帰属します。

4. 保証の制限
著作者は、許諾プログラムに関していかなる保証も行いません。許諾プログラムに関し発生する問題はお客様の責任および費用負担をもって処理されるものとします。

5. 責任の制限
著作者は、いかなる場合も、お客様の逸失利益、特別な事情から生じた損害（損害発生につき著作者が予見し、または予見し得た場合を含みます）および第三者からお客様に対してなされた損害賠償請求にもとづく損害について一切責任を負いません。

6. 再配布の制限
ライセンスの再配布は行えません。
パッケージの内容を変更せず、パッケージの状態でのみ再配布が行えます。
パッケージに含まれる個別ファイルの再配布は行えません。ただし、SimpleMeasure.xlsm については、変更したものを個別で再配布が行えます。

1 はじめに

本ソフトウェアを使用する前には、以下の実行環境が必要となります。

●オペレーティングシステム
Windows 32 ビットまたは 64 ビット
　Windows 7 SP1、Windows 8.1、Windows 10

●Microsoft .NET Framework 4.6.2
Microsoft .NET Framework 4.6.2 がセットアップされている必要があります。
次の URL からダウンロードできます。
https://www.microsoft.com/ja-jp/download/details.aspx?id=53344

●アプリケーション
Excel 32 ビット
　Excel 64 ビットでは動作しません。
　Excel 2007、Excel 2016
　Excel 2010、Excel 2013 は動作未確認

●インターフェース（使用する場合のみ）
GPIB インターフェース
　GPIB インターフェース用のドライバーもセットアップする必要があります。
　National Instruments 社の GPIB-USB-HS や KEYSIGHT 社の 82357B などを使用する場合に必要になります。GPIB インターフェースを使用するには次の VISA ドライバーも必要となります。

●VISA ドライバー（使用する場合のみ）
　VISA でサポートされているインターフェースの GPIB、USB（USBTMC）、RS-232C、LAN などを使用する場合、VISA ドライバーをセットアップする必要があります。
　VISA の動作確認は次のバージョンで行いました。
　　National Instruments　　NI-VISA 17.5
　　KESIGHT　　IO ライブラリスイート 10.2017
　次の VISA ドライバーは、VISA-COM のタイプライブラリの確認のみ行いました。
　　菊水電子工業　　KI-VISA 5.5.0.275
　National Instruments 社、菊水電子工業社の VISA ドライバーは、接続されている測定器やインターフェースのメーカーのものを使用します。他社製品のみ接続されている環境では使用できない場合がありますので注意してください。KESIGHT 社の VISA ドライバーは他社製品のみ接続されている環境でも使用できます。詳細は各社 VISA ドライバーの使用許諾を参照してください。

●ナショナルインスツルメンツ社の DAQmx（使用する場合のみ）

DAQmx 用ドライバーをセットアップする必要があります。（このインターフェースは VISA には対応していません。）

２ セットアップ

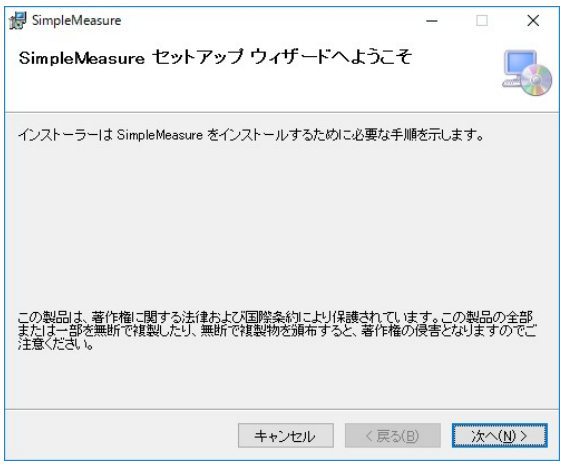
２-１ インストール

GPIB、VISA、DAQmx のセットアップはホームページ、またはそれぞれのドライバーメーカを参照してください。

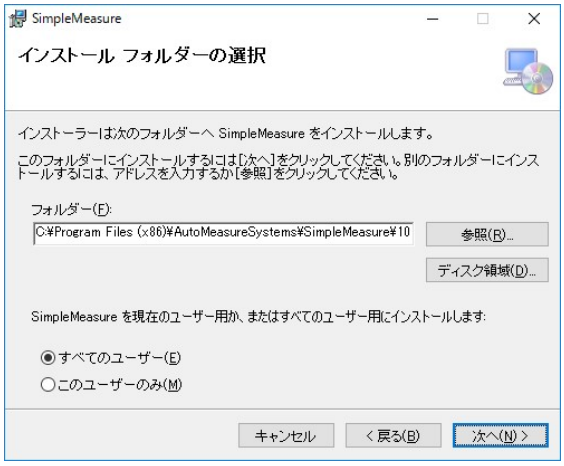
Simple Measure は 1 つのライセンスで 1 台のパソコンまたは 1 つの仮想マシンに 1 つだけセットアップできます。

ZIP ファイルを展開し、Setup.exe を実行してください。
実行すると、以下の画面になります。

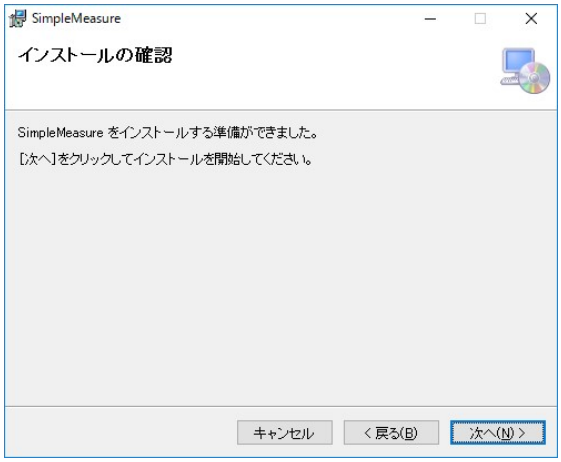
著作権に関する文章が表示されます。同意された場合は「次へ」を押してください。



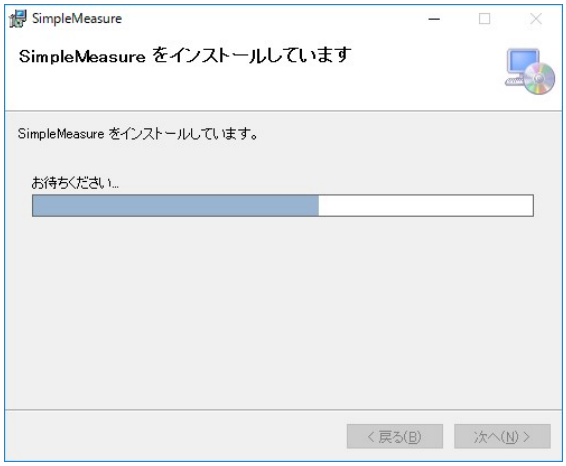
インストールフォルダーを指定してください。通常は変更せずにこのまま「次へ」を押してください。



インストールを開始します。
「次へ」を押してください。

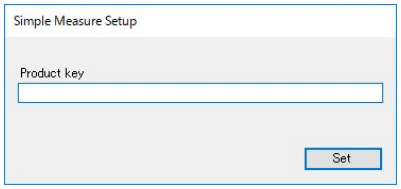


ファイルのコピーが行われます。

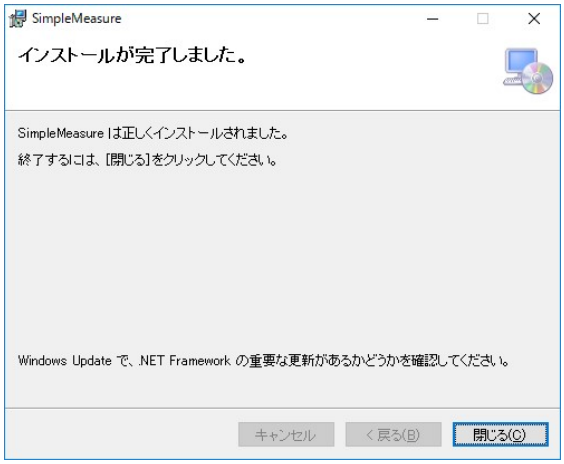


インストール中に、プロダクトキーの入力ダイアログが表示されます。
プロダクトキーを受け取っている場合は、入力してください。
トライアル版またはフリー版の場合は何も入力せずに「Set」を押してください。

過去に Simple Measure をインストールした場合、試用期間は更新されません。最初にインストールされた年月日から計算します。試用期間が過ぎている場合はフリー版としてインストールされます。



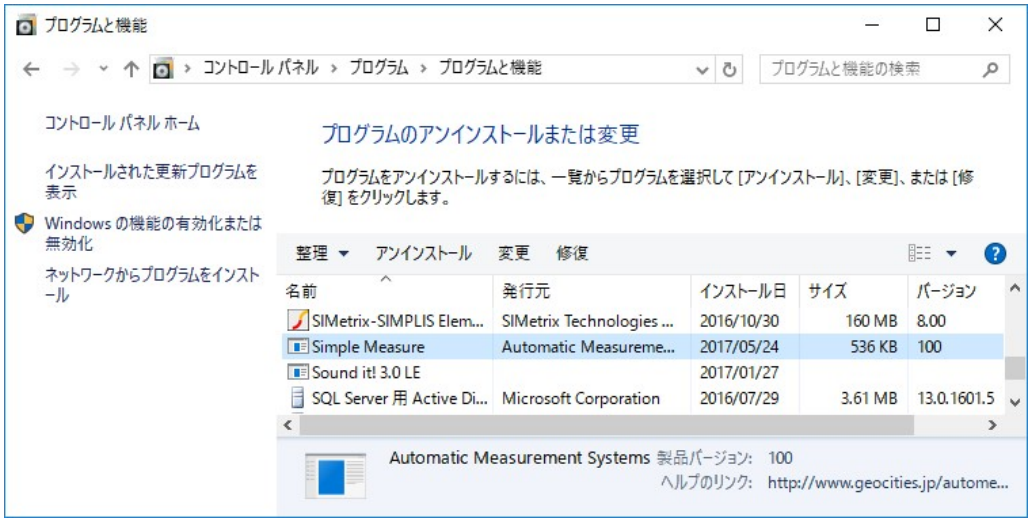
以上でインストールは完了です。



プロダクトキーを設定したときは、「2-4 バージョン情報表示」を参照して登録状態を確認してください。

2-2 アンインストール

[コントロールパネル][プログラムのアンインストール]で
　　名前 SimpleMeasure、発行元 AutoMeasureSystems
を選択し、で右クリックして[アンインストール]を選択してください。
アンインストールしても測定したデータが保存されている Excel ファイルを開くことはできますが、Measure フォームや ScriptTest フォームなどを実行することはできません。
この操作で GPIB、VISA、DAQmx などのドライバーはアンインストールされません。それらのアンインストールはドライバーごとの方法によって行ってください。



2-3 プロダクトキー登録

プロダクトキーを後から設定する場合、次の 2 通りの方法があります。
1．いったんアンインストールしてからインストールを行い、プロダクトキーを入力します。
2．セットアップフォルダー
　・32 ビット OS のデフォルトでは
　　C:¥Program Files¥Automatic Measurement Systems¥Simple Measure¥100
　・64 ビット OS のデフォルトでは
　　C:¥Program Files (x86)¥Automatic Measurement Systems¥Simple Measure¥100
　の「Setup.exe」を管理者権限で実行し、プロダクトキーを設定します。

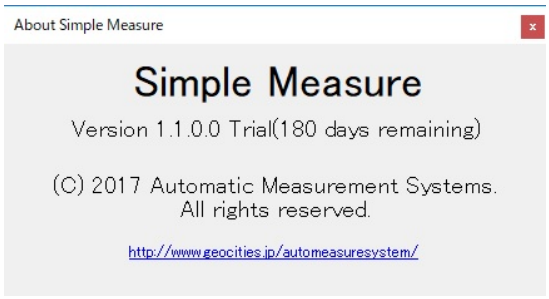
SimpleMeasure.xlsm を開き About を表示したとき、バージョン表示の右側の Trial または Free の表示がなければ正常に登録が完了しています。「2-4 バージョン情報表示」を参照してください。

2-4 バージョン情報表示

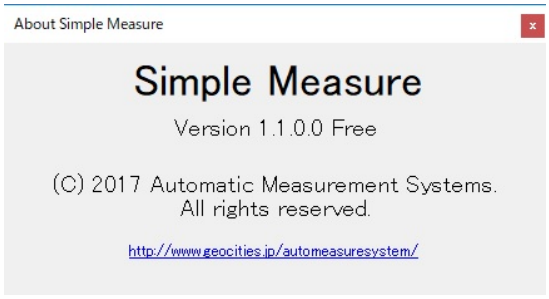
SimpleMeasure.xlsm を Excel で「マクロを有効」にして開き、Menu シートの About を実行すると、次のように表示されます。（バージョンの数値は異なる場合があります。）

正規のプロダクトキーを設定したときの表示。
正規版で実行しています。

プロダクトキーを設定しないでセットアップを行ったときの表示。
トライアル版で実行しています。
残りの日数はインストールした日時から計算されて表示されます。



プロダクトキーを設定しないでセットアップし、試用期間が過ぎたときの表示。
フリー版で実行しています。



３ 基本操作

試験プログラムを作成するための準備をします。
新規に作成する場合は以下のようになります。
Excel はマクロが有効になるように設定してください。

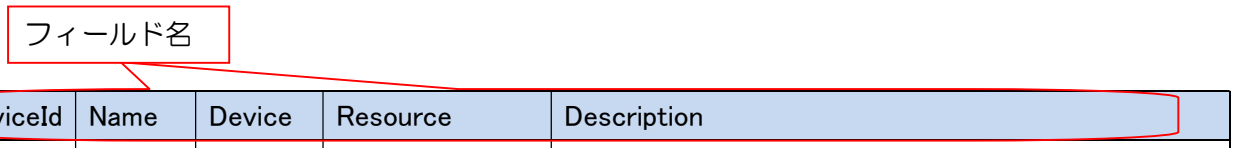
・作業用ファイルの作成
　展開した ZIP ファイル内の SimpleMeasure.xlsm をコピーして適当な名前にリネームして作成してください。ファイルを開くとき、マクロを有効にします。
　この Excel ファイルに測定条件を設定し、試験結果を保存します。通常はこのファイル 1 つですべての操作が行えます。
・測定器の登録
　Device シートに測定器を登録します。
・試験項目の作成
　Measure シートに試験項目を作成します。
・スペック判定値の設定
　OutOfSpec、LowerSpec、UpperSpec シートにスペック判定範囲の設定を行います。
　スペック判定が必要ない場合は設定の必要はありません。

３-１ 測定器の登録（Device シート）

Device シートに測定器を登録します。
DAQmx は測定器として登録できません。「４ スクリプト」を使用して操作を行います。

「図 ３-１ Device シート」は設定例です。
システムデバイスが DeviceId の 1 から 7 までで、これらは変更不要です。8 と 9 が登録した測定器です。

図 3-1 Device シート



DeviceId	Name	Device	Resource	Description
1	Dummy	Dummy		システムデバイス。何もしない。
2	End	End		システムデバイス。測定の最後。 この行以降は実行しない。
3	Input	Input	Simple Measure	システムデバイス。インプットボックスを表示する。
4	Msg	Message	Simple Measure	システムデバイス。メッセージボックスを表示する。
5	Picture	Picture	Simple Measure	システムデバイス。ピクチャーボックスを表示する。
6	Var	Variable		システムデバイス。変数オブジェクト。
7	Wait	Wait		システムデバイス。実行を設定時間停止する。 単位は[msec]
8	Osc	VISA	GPIB0::22::INSTR	発振器
9	Oscillo	VISA	GPIB0::2::INSTR	オシロスコープ

各フィールドの説明

DeviceId

1 から始まる連番を設定します。システムでは使用していませんが、データーベースに登録するときのインデックスとしての条件を満たす設定にしておくことをお勧めします。

Name

後述する、測定項目を設定する Measure シートの Device フィールドで使われます。重複しない任意の名前を設定します。一文字目が ”;” の場合、そのデバイスは無効となります。

Device

システムに登録されているデバイス名を設定します。詳細は「3-2 Device の詳細」を参照してください。

Resource

各デバイスがオブジェクトを作成するときに設定するオプションです。Device ごとに設定できる内容が異なります。詳細は「3-2 Device の詳細」を参照してください。

Description

システムでは使用していません。各デバイスの役割や機能などの説明を記述します。

３-２ Device の詳細

システムに登録されているデバイスです。大文字小文字の区別はありません。Device はメッセージボックスの表示や、VISA などのインターフェースへの入出力を行うオブジェクトです。
Device への設定は Device シートの Resource で行います。Device への操作は Measure シートの Method で行います。
次の説明で Resource は Device シートの Resource フィールドで設定します。Parameter と Parameter2 は、Measure シートのフィールドで、Method の第 1、2 引数となります。各機能は次のようになります。

３-２-１ 各 Device 共通事項

Method

Script　CurrentDevice に呼び出したオブジェクトが設定され、Parameter に設定されているコードを実行します。スクリプトへの引数は Parameter2 に設定します。詳細は「４ スクリプト」を参照してください。

Read、ReadString

読み取り操作を行います。このメソッドが有効な場合、Read はカンマまたはセミコロン区切りで分割、ReadString はそのままの文字列を Data シートに返します。最後に読み取った値を記録する変数 LastaData には、ReadString が返す値と同じものが記録されます。

Parameter に値が設定されている場合、オブジェクトに書き込みを行った後、読込を行います。

３-２-２ Device

各デバイスの機能は以下のようになります。

Dummy

機能

何もしないオブジェクトです。

Resource

指定できません。

Method

Read　Parameter が設定されていない場合は、最後に測定した値（LastData）をカンマまたはセミコロン区切りで分割して返します。

Parameter が設定されている場合は、その値をカンマまたはセミコロン区切りで分割して返し、LastData にその値を設定します。

ReadString　Parameter が設定されていない場合は、最後に測定した値（LastData）を返します。

Parameter が設定されている場合は、その値を返し、LastData にその値を設定します。

Timeout　使用できません。

Write　最後に読み取った値を記録する変数 LastData に Parameter の値が設定されます。

End

機能

測定項目の終わりを示すオブジェクトです。

Resource

指定できません

Method

Read	使用できません。
ReadString	使用できません。
Timeout	使用できません。
Write	現在の項目で実行を停止します。Measure フォームの Measure ボタンが有効となり、Measure シートのカーソル移動が可能となります。

Input

機能

インプットボックスを表示し、入力された値を取得します。

「キャンセル」を押すと測定を中断します。エンターキーで OK が押され、エスケープキーでキャンセルが押されます。

Resource

インプットボックスのタイトルに表示するメッセージを設定します。

Method

Read	InputBox を表示し、Parameter に設定されている値をメッセージに表示、入力された値をカンマまたはセミコロン区切りで分割して返します。
ReadString	InputBox を表示し、Parameter に設定されている値をメッセージに表示、入力された値を返します。
Timeout	使用できません。
Write	使用できません。

Message

機能

メッセージボックスを表示するオブジェクトです。

「OK」を押すと実行を再開、「キャンセル」を押すと測定を中断します。表示されたときの状態ではエンターキーまたはスペースキーで OK が押され、エスケープキーでキャンセルが押されます。

Resource

メッセージボックスのタイトルに表示されるメッセージを設定します。

Method

Read	使用できません。
ReadString	使用できません。
Timeout	使用できません。
Write	Parameter に設定されている値をメッセージボックスに表示し、実行を一時停止します。

Picture

機能

画像ファイルを表示するオブジェクトです。
指定できる画像ファイルは次のようになります。

BMP
Windows で使用される標準形式です。

GIF (Graphics Interchange Format)
GIF は、Web ページ上に表示されるイメージの一般的な形式です。
複数の GIF イメージのシーケンスを 1 つのファイルに格納して、GIF のアニメーションを作成できます。 GIF は、ピクセルあたり最大 8 ビットしか格納できないため、色の数は 256 色に制限されます。

JPEG (Joint Photographic Experts Group)
非可逆圧縮の画像フォーマットです。

EXIF (Exchangeable Image File)
EXIF は、デジタル カメラでキャプチャした写真のために使用されるファイル形式です。

PNG (Portable Network Graphics)
圧縮による画質の劣化のない可逆圧縮の画像ファイルフォーマットです。

TIFF (Tag Image File Format)
TIFF は、さまざまなプラットフォームおよびイメージ処理アプリケーションで広くサポートされる、柔軟性と拡張性の高い形式です。

ICO
Windows におけるアイコンで使用する画像ファイルフォーマットです。

「OK」を押すと実行を再開、「キャンセル」を押すと測定を中断します。表示されたときの状態ではエンターキーまたはスペースキーで OK が押され、エスケープキーでキャンセルが押されます。

Resource

ピクチャーボックスのタイトルに表示されるメッセージを設定します。

Method

Read　使用できません。

ReadString　使用できません。

Timeout　使用できません。

Write　Parameter はカンマ区切りで設定します。

画像ファイル名，表示するテキスト [，幅，高さ]

幅と高さは省略可能です。

OK または Cancel ボタンを押すまで実行を一時停止します。

Variable

機能

測定値や定数を保存する変数オブジェクトです。

Resource

指定できません

Method

Read　Parameter に設定された変数名から値をカンマまたはセミコロン区切りで読み取ります。

ReadString　Parameter に設定された変数名から値を読み取ります。

Timeout　使用できません。

Write　Parameter に設定された変数名に、最後に測定した値（LastData）を記録します。

Wait

機能

一定時間実行を中断するオブジェクトです。

キャンセルボタンが押された（または Status.Cancel が true になった）ときは、Wait 処理を中止し、測定を中断します。

Resource

指定できません

Method

Read　使用できません。

ReadString　使用できません。

Timeout　使用できません。

Write　Parameter に設定された時間、実行を一時中断します。設定値は 0 以上の整数で単位は[msec]です。

VISA

機能

GPIB や USBTMC、RS-232C、LAN などのインターフェースに接続された測定器と通信するオブジェクトです。

Resource

VISA リソースを設定します。指定するインターフェースによって異なります。詳細は VISA のマニュアルを参照してください。設定する値は NI MAX などのツールを利用しても確認できます。

GPIB

GPIB[ボード番号]::1 次アドレス[::GPIB 2 次アドレス][::INSTR]

例

GPIB ボード番号：0　　省略

1 次アドレス：2

2 次アドレス：なし　　省略

Resource に設定する値：　**GPIB::2::INSTR**

USBTMC

USB[ボード]::manufacturer ID::model code::serial number[::USB インターフェース番号][::INSTR]

例

USB ボード：0　　省略

manufacturer ID：0x1234（16 進数）

model code：125

serial number：A22-5

USB インターフェース番号：1　　省略

Resource に設定する値：　**USB::0x1234::125::A22-5::INSTR**

Method

Read	測定器からカンマまたはセミコロン区切りで読み取ります。
ReadString	測定器から読み取ります。
Timeout	VISA に Timeout 値を設定します。単位は[msec]です。
Write	Parameter に設定された文字列を VISA デバイスに書き込みます。

３-３ 試験項目の設定（Measure シート）

試験やシーケンス処理を行う条件を設定します。接続の変更などのメッセージ表示や、測定器への設定、測定値の読み取りなどの操作を行います。
「図 ３-2 Measure シート」は設定例です。

各フィールドの説明

MeasureId

1 から始まる重複しない連番を設定します。測定を実行すると Data シートにコピーされます。
システムでは使用していませんが、データーベースで処理を行うときに、この値を使用して Data シートと内部結合させます。インデックスとしての条件を満たす値に設定しておくことをお勧めします。

Label

システムでは使用していません。測定データの項目名として使用します。データの識別ができるように設定します。値は重複しても構いません。

Method

Read、ReadString、Script、Timeout、Write があります。Device フィールドで指定したオブジェクトを操作します。機能については「３-２ Device の詳細」を参照してください。最初の一文字目が ”;” の場合、その項目は無効となります。

Device

使用するオブジェクトを設定します。Device シートの Name フィールドで設定した名前を設定します。大文字小文字の区別はありません。

Parameter

Method の第 1 引数です。

Parameter2

Method の第 2 引数です。

ReportId

システムでは使用していません。試験成績書などを作成するとき、データの書き込み先の位置情報などを設定しておきます。

Description

システムでは使用していません。測定項目の説明を記述します。

図 3-2 Measure シート

フィールド名

MeasureId	Label	Method	Device	Parameter	Parameter2	ReportId	Description
1	●初期化	Write	Osc	*Rst			
2	初期化	Write	Oscillo	*Rst			
3	初期化	Write	Oscillo	Autoset Exec;			
4	初期化	Write	Oscillo	:MEASUrement:MEAS1:SOURCE CH1;:Measurement:Meas1:State ON;:MEASUrement:MEAS1:TYPe CRms;			
5	初期化	Write	Oscillo	:MEASUrement:MEAS2:SOURCE CH1;:Measurement:Meas2:State ON;:MEASUrement:MEAS2:TYPe Ampl			
6	出力電圧	Write	Msg	OSC:　CH1 出力 -> オシロ: CH1			
7	●出力電圧 Sin 20Vpp	Write	Osc	*Rst; Amv 20			
8	出力電圧 Squ 20Vpp	Write	Wait	1000			
9	出力電圧 Sin 20Vpp	Write	Oscillo	Autoset Exec; :ACQuire:NUMAVg 8; :ACQuire:Mode Average			
10	出力電圧 Sin 20Vpp	Write	Wait	3000			
11	出力電圧 Sin 20Vpp	Read	Oscillo	:Measure:Meas1:Value?			

Label のつけ方と初期化設定

測定器は最初にすべて初期化します。あとは、必要に応じて各測定器を設定し測定を行います。

測定の実行中、接続ミスなどのエラーによって測定をやりなおしたい場合があります。
そのような時、キャンセルボタンを押して測定を一時中断し、やり直す項目に移動して測定を再開することができます。
測定項目を選択して実行できるように構成する場合、次の点に注意する必要があります。

単純に戻った位置から始めると、測定条件が変わってしまう場合があります。
例えば、デジタルマルチメータで直流電圧を測定し、そのあと交流電圧の測定が終わった時点でエラーに気が付いたとします。このとき、直流電圧の測定に戻ると、戻った時点では交流測定の状態になっています。そのような状態にならないように、測定項目の区切りで初期化を実行して、常に同一条件で測定が行えるように構成することが重要です。
戻れる位置が明確になるように Label に段落記号のような、しるしを付けておくのも一つの方法です。

下記の MeasureId が 1 と 7 に戻れる位置を示す、しるしをつけてあります。
「*Rst」は測定器の初期化を行うコマンドです。7 行目の Osc は「*Rst;Amv 20」となっており、初期化後、振幅を 20Vpp に設定しています。

MeasureId	Label	Method	Device	Parameter
1	●初期化	Write	Osc	*Rst
2	初期化	Write	Oscillo	*Rst
…				
7	●出力電圧 Sin 20Vpp	Write	Osc	*Rst; Amv 20

測定器の設定

Method の Write は測定器などにコマンドを送ります。コマンドが長くなる場合は分割します。測定機のバッファーには制限があります。サイズを超えないように注意してください。また、コマンドの組み合わせによっては同時に設定できない場合もあります。設定が反映されない、エラーとなる場合はコマンドをひとつずつ送れば、どこで問題が発生しているか確認できます。
バイナリーデータを送る場合は「４ スクリプト」を使用します。

測定値の取得

測定値の取得は Write で問い合わせコマンドを送り、そのあと Read または ReadString で取得する方法と、Read または ReadString の Parameter に問い合わせコマンドを設定する方法があります。バイナリーデータの取得・設定は Excel のセルに直接書き込めないので「４ スクリプト」を使用し、データを変換しま

す。
Device が DMM で、測定値の取得コマンドが“MEAS:VOLT:AC?”の場合は次の 2 つの方法は同じ結果を返します。ただし、Data シートに記録される行は Read または ReadString の行によって変わります。

問い合わせコマンドを送った後、すぐに読み取ると正しい値が読み取れない場合がります。そのような時は、Write で問い合わせコマンドを送り、Wait で待ち時間を挿入、Read で読み取るようにタイミングを調整してください。カウンターのインターバル設定が長い場合や、古い測定器などは読み取るタイミングの調整が必要になる場合があります。

Write と Read を使用した測定値の取得

MeasureId	Label	Method	Device	Parameter
10	問い合わせ	Write	DMM	MEAS:VOLT:AC?
11	データ取得	Read	DMM	

Read のみ使用した測定値の取得

MeasureId	Label	Method	Device	Parameter
10	測定	Read	DMM	MEAS:VOLT:AC?

Read と ReadString の違い

Read はカンマまたはセミコロン区切りの結果を分割して取得します。
取得した測定値が次のような場合、Data シートの Data1、Data2・・・に分割して記録されます。
取得データ：1.000E3,20.00,45.00

Read の場合

Data1	Data2	Data3
1.000E3	20.00	45.00

ダブルクォーテーションまたはシングルクォーテーションでくくられた文字列の場合は分割されません。
Data シートにはその中身のみ記録されます。
取得データ：”xx Electronic,Model 1234A,S/N:666”

Read でダブルクォーテーションの文字列の場合

Data1	Data2	Data3
xx Electronic,Model 1234A,S/N:666		

ReadString は取得した文字をそのまま Data シートに記録します。
取得データ：1.000E3,20.00,45.00

ReadString の場合

Data1	Data2	Data3
1.000E3,20.00,45.00		

測定器コマンドの調べ方

最近の測定器は SCPI 準拠となり、構文の階層が深く、目的の操作を行うコマンドを見つけるのに苦労することがあります。基本的な操作は、測定器の取扱説明書などに書かれているサンプルプログラムを参考にするのが近道です。それをひな型にして設定を追加していきます。

3-4 データ（Data シート）

Measure シートで、Read または ReadString で Device から取得した値は Data シートに記録されます。Write や Wait など、戻り値のないメソッドの行は MeasureId のみ記録され、Data フィールドは空白となります。Measure シートと Data シートの関係は「図 3-3 Measure シートと Data シート」のようになります。

図 3-3 Measure シートと Data シート

Measure シート（フィールドの一部）

MeasureId	Label	Method	Device
1	初期化	Write	Osc
2	初期化	Write	Oscillo
3	初期化	Write	Oscillo
4	初期化	Write	Oscillo
5	初期化	Write	Oscillo
6	出力電圧	Write	Msg
7	出力電圧 Sin 20Vpp	Write	Osc
8	出力電圧 Squ 20Vpp	Write	Wait
9	出力電圧 Sin 20Vpp	Write	Oscillo
10	出力電圧 Sin 20Vpp	Write	Wait
11	出力電圧 Sin 20Vpp	Read	Oscillo
12	出力電圧 Sin 10Vpp	Write	Osc
13	出力電圧 Sin 10Vpp	Write	Wait
14	出力電圧 Sin 10Vpp	Write	Oscillo
15	出力電圧 Sin 10Vpp	Write	Wait
16	出力電圧 Sin 10Vpp	Read	Oscillo

Data シート（フィールドの一部）

MeasureId	OutOfSpec	Data1
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11	FALSE	7.14E+00
12		
13		
14		
15		
16	FALSE	3.58E+00

各フィールドの説明

MeasureId

Measure シートと Data シートの MeasureId は同じ行であることを示します。データーベースで処理を行う場合は、MeasureId で内部結合します。

OutOfSpec

規格値に対して測定値の合否判定を行います。規格外の場合 True になります。このフラグは LowerSpec、UpperSpec に設定してある値の範囲内か否かで判定されます。スペックの判定方法は「３-５ スペック判定」を参照して下し。

このフィールドは任意で使用してください。必要のない場合は空白にしても動作します。

Data1、Data2・・・

測定器などから取得したデータが記録されます。Data1、Data2・・・と分割されたデータが記録される条件は、カンマまたはセミコロン区切りのデータを Read で読み取るか、スクリプトを使用した場合です。詳しくは「３-３ 試験項目の設定（Measure シート）」の「Read と ReadString の違い」と「４ スクリプト」を参照してください。

3-5 スペック判定（xxxSpec シート）

スペック判定値の設定は、Measure シートの作成が完了してから行ってください。Measure シートに変更を加えた場合、スペック判定を行っている 3 つのシートも変更が必要となる場合があります。

スペック判定値の設定

複数のデータの判定ができるように、スペック判定は 3 つのシートを使用して行います。

OutOfSpec　判定
LowerSpec　下限値の設定
UpperSpec　上限値の設定

LowerSpec と UpperSpec に上下限値を設定し、OutOfSpec で判定を行います。
以下に各シートの参照関係を示します。ここでは Data1 と Data2 のみの判定となります。Data3 以上の判定を行う場合には、Data3 以降に Data2 のセルをコピーし、「=OR()」の式の範囲を拡大してください。
Data シートの C3 に記録されている -3.5 が LowerSpec の -3.1 以下となっているためエラーになっています。赤の塗りつぶしと白い文字の設定は Excel の「条件付き書式」で設定していますので、好みの設定に変更できます。

LowerSpec シート

	A	B	C
1	MeasureId	Data1	Data2
2	1	−0.1	−1
3	2	−3.1	−46

UpperSpec シート

	A	B	C
1	MeasureId	Data1	Data2
2	1	0.1	1
3	2	−2.9	−44

OutOfSec シート（数式表示）

	A	B	C	D
1	MeasureId	SpecOver	Data1	Data2
2	1	=OR(C2,D2)	=OR(Data!C2<LowerSpec!B2, UpperSpec!B2<Data!C2)	=OR(Data!D2<LowerSpec!C2, UpperSpec!C2<Data!D2)
3	2	=OR(C3,D3)	=OR(Data!C3<LowerSpec!B3, UpperSpec!B3<Data!C3)	=OR(Data!D3<LowerSpec!C3, UpperSpec!C3<Data!D3)

Data シート

	A	B	C	D
1	MeasureId	OutOfSpec	Data1	Data2
2	1	FALSE	0.05	0
3	2	TRUE	−3.5	−45

=OutOfSpec!B3

通常は数式表示させないため、OutOfSpec シートは次のような表示となります。

OutOfSec シート（数式表示なし）

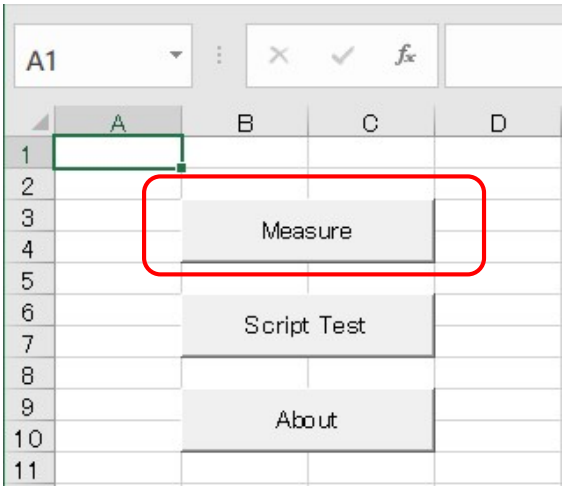
	A	B	C	D
1	MeasureId	SpecOver	Data1	Data2
2	1	FALSE	FALSE	FALSE
3	2	TRUE	TRUE	FALSE

Measure シートを変更したときの注意

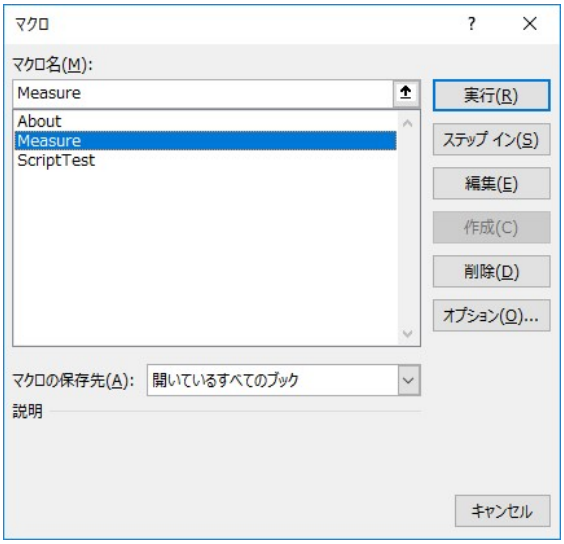
Measure シートを変更し、データの記録される位置が変わった場合、スペックを判定している 3 つのシートとの相対位置が変わってしまいます。この位置関係が変わらないようにするため、Measure シートに行を追加した場合はスペック判定を行っているそれぞれのシートの同じ位置にも、行を挿入してください。Measure シートの行を削除した場合はスペックシートの該当行を削除してください。

3-6 測定の実行

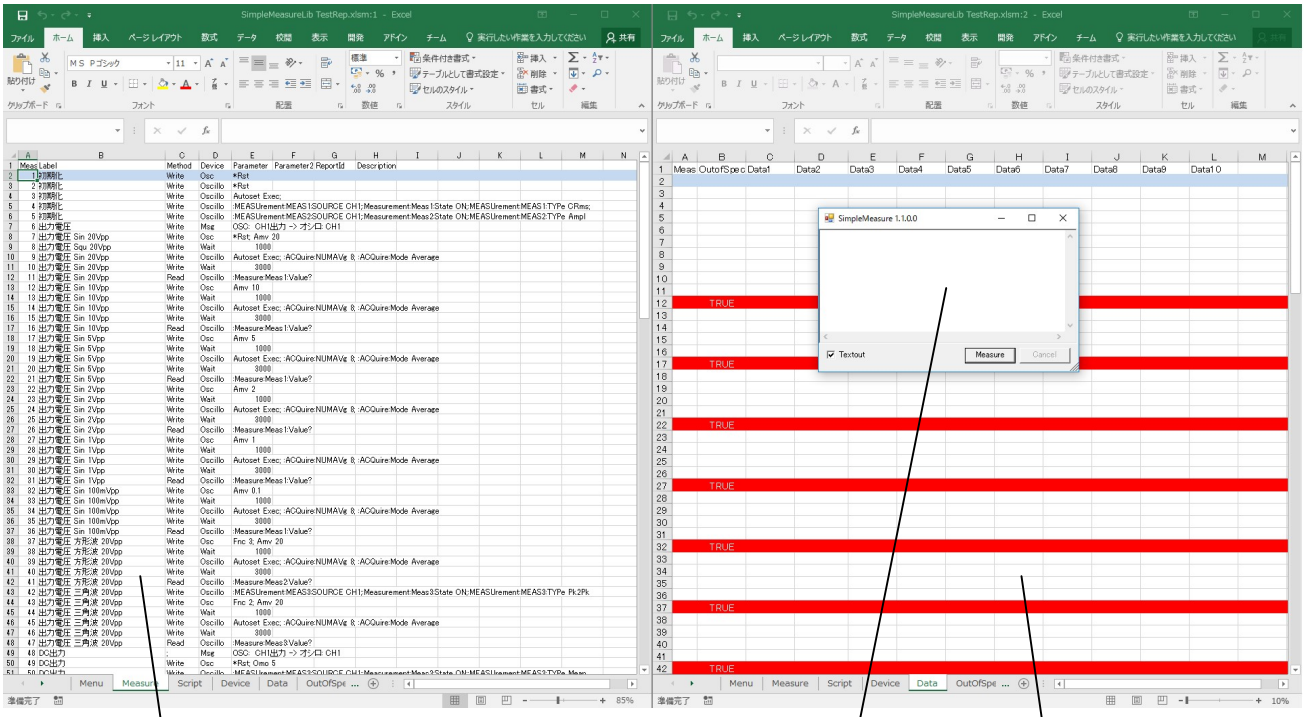
Menu シートの「Measure」ボタンを押すと Measure フォームが開きます。



このボタンは、標準モジュール「SimpleMeasure」の「Public Sub Measure()」を呼び出しています。[開発][マクロ]から呼びだすこともできます。



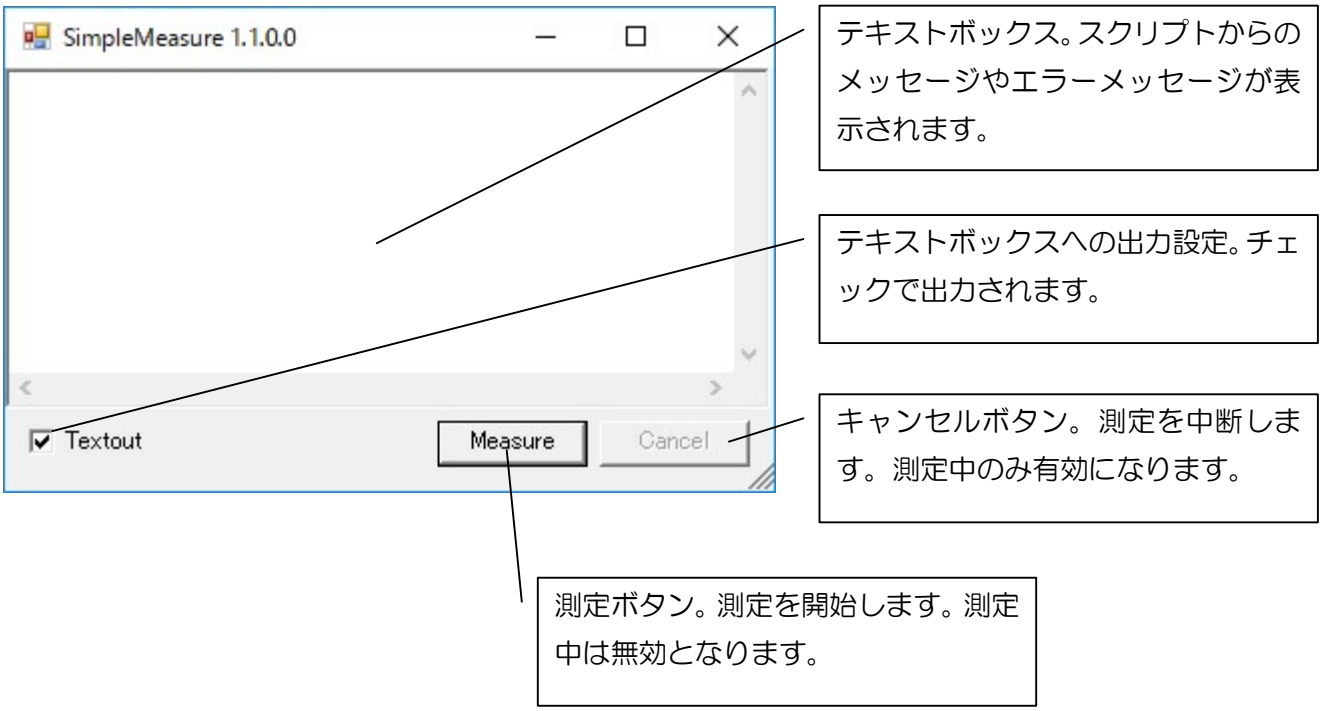
実行すると、Measure シートと Data シートが整列されて表示されます。[表示][新しいウインドウを開く]で複数のシートが表示されている場合は整列されません。



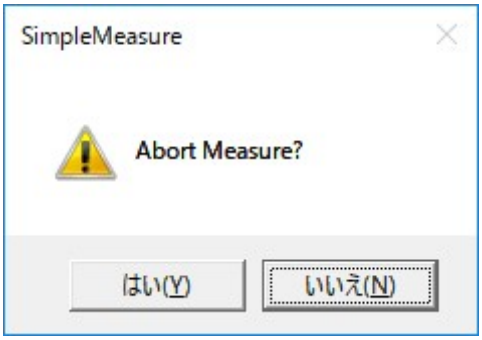
実行すると、Measure フォームが表示され、Measure シートと Data シートのカーソルは MeasureId = 1 の位置に移動します。フォーカスがテキストボックス以外にある場合、下矢印キーを押すと Measure と Data シートのカーソルが 1 つずつ下に移動します。PageDown を押すと 10 行ずつ下に、End キーで最終行へ移動します。最終行を超えての移動はできません。上矢印、PageUp、Home でそれぞれ上方向へのカーソル移動となります。

Measure フォームの「Measure」ボタンを押すと、カーソル位置から測定を始めます。各シートのカーソルは測定が進むにしたがって移動します。

Measure フォーム



測定を中断する場合は「Cancel」ボタンを押します。「Abort Measure」ダイアログが表示されますので、そこで「はい」を押すと測定は停止します。「いいえ」を押すと測定はそのまま実行を続けます。「Abort Measure」ダイアログが表示されている状態では、実行は中断されません。



3-7 動作確認

すべての準備が整ったら、動作確認を行います。
生産ラインで使用するようなシステムや再測定が困難な場合は、十分な動作確認を行う必要があります。
測定結果が正しいか数値的に判断します。

一般に、測定値のばらつきは正規分布します。したがって、同じ条件での測定を繰り返し、その測定結果の標準偏差を求めれば数値的に判断することができます。
n が十分大きい場合の標準偏差と確率の関係は次の表のようになります。

区間	$\pm 1.96\sigma$	$\pm 2.00\sigma$	$\pm 2.58\sigma$	$\pm 2.81\sigma$	$\pm 3.00\sigma$
区間内確率[%]	95.0	95.44	99.0	99.5	99.74
区間外確率[%]	5.0	4.56	1.0	0.5	0.26

目的の精度が得られない場合や正規分布しない場合は、測定条件の検討やグランドループ、ノイズの混入などの測定環境の確認を行います。
生産ラインでのトラブルは大きなコストとなりますので、十分な動作確認を行うよう心がけます。

測定器が故障し、その測定器が購入できない場合など、別のものに置き換えなければならないときにも、この検証結果が役に立ちます。測定精度が同じかどうかの比較ができるので置き換えの判断が的確にできるようになります。

４ スクリプト

Simple Measure では C#スクリプトと C#マクロが使用できます。ここでは、それらを総称してスクリプトとします。

C#スクリプトはMeasureシートに記述し、C#マクロはExcelのVBAから呼び出すプログラムコードです。これらの違いは、C#マクロは Excel のマクロ実行など、任意のタイミングで実行できるのに対し、C#スクリプトは Measre シートの測定を実行中の時にだけ実行されます。それ以外の機能的な違いはありません。
ただし、C#マクロでは測定実行中の固有値である Index や CurrentDevice、LastData などは意味を持ちません。

スクリプトを使用することによって、NI DAQmx などの VISA 以外のインターフェースの使用や測定値の平均化処理、データに測定日時などの付加情報を加えるなど、測定器から取得したデータを加工したい場合に使用します。

４-１ スクリプト言語仕様

スクリプト言語

言語：　　C#
コンパイラバージョン：　2.0、3.5、4.0

スクリプトのソースコードは、次に示すように MeasureScriptBase を基底クラスとする MeasureScript クラスの部分クラスとし、最初に呼び出される public override object[] Main(string value) を実装します。Main メソッドの引数 value には Measure シートの Parameter2 の値が設定されています。

```
1       // Hello world サンプルスクリプト
2       using System;
3       namespace SimpleMeasure
4       {
5           public partial class MeasureScript : MeasureScriptBase
6           {
7               public override object[] Main(string value)
8               {
9                   Console.WriteLine("Hello world");
10                  return null;
11              }
12          }
13      }
```

網掛け部分が最低限必要なコードです。戻り値を設定する場合は 10 行目の return で、objet の配列を返

します。各要素は Data シートの、object[0]は Data1 に object[1]は Data2 に返されます。
9 行目の Console.WriteLine()は Measure フォームなどのテキストボックスに出力されます。

MeasureScript クラスにコンストラクタを記述した場合、C#スクリプトでは最初にスクリプトが実行されたときに 1 回だけ呼び出されます。Measure フォームが閉じられるまでインスタンスは有効です。
また、まったく同じ内容（コメントを含む）のスクリプトは同じスクリプトのインスタンスが使用されます。（コンパイルされません。）
C#マクロでは CallScript()を呼び出すごとにコンストラクタが呼び出されます。

ループを作成した場合、「MeasureStatus クラス」の Status.Cancel が true になったとき、ループを抜けるようにしてください。無限ループにした場合、終了できなくなります。System.Threading.Sleep()も同様です。長時間の Sleep()を呼び出す場合は、同等の処理を行い Status.Cancel で中断する DevObj.DeviceList["Wait"].Write("waitTime") を使用してください。ここで waitTime は 0 以上の整数値で単位は[msec]です。

スクリプトのテストを行う「４-３ ScriptTest フォーム」ツールがあります。基本的な動作テストはこのツールを使用することができます。

Measure シートへのスクリプトの実装は、Method フィールドに”Script”を指定し、Parameter フィールドにスクリプトのコードを貼り付けます。引数がある場合には Parameter2 フィールドに設定します。

Measure シートからスクリプトを実行

MeasureId	Label	Method	Device	Parameter

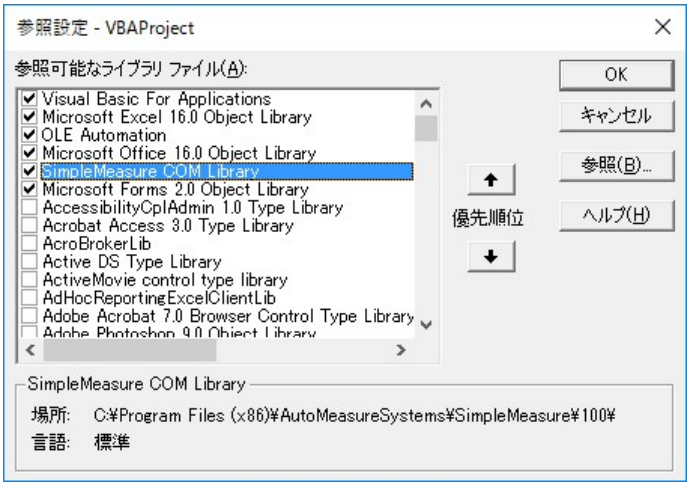
1	サンプル	Script	Dummy	// Hello world サンプルスクリプト
				using System;
				namespace SimpleMeasure
				{
				public partial class MeasureScript : MeasureScriptBase
				{
				public override object[] Main(string value)
				{
				Console.WriteLine("Hello world");
				return null;
				}
				}
				}

C#マクロではコードを適当なセルに記述し、CsForApp()または CallScript()の引数である script に文字列としてコードを設定し、呼び出します。

Excel の VBA からの呼び出しコード

VBA の参照設定

VBA の IDE で[ツール][参照設定]で SimpleMeasure COM Library を設定します。



SimpleMeasre オブジェクトの設定

標準モジュール SimpleMeasure の定義部分に次のコード、または同等のコードを記述します。このコードは必ず必要です。

Public MesObj As New SimpleMeasureLib

Excel オブジェクトの設定

次のように設定します。このコードは必ず必要です。

標準モジュール SimpleMeasure の SetConfig メソッドで MesObj.AddConfig メソッドで Application を設定します。

```
' Excel
' Application
MesObj.AddConfig "Excel.Application", Application
```

C#アセンブリ参照設定

Variant 配列に参照するアセンブリ名を設定してから登録します。指定しなかった場合は、次のコードを実行したときと同じ参照が設定されます。アセンブリ参照を追加する場合は次のコードを標準モジュール SimpleMeasure の SetConfig メソッドに書き加えてください。

MesObj.AddConfig メソッドで第 1 引数に "Script.References"、第 2 引数にアセンブリ名の配列を指定して呼び出します。

```
'Script で使用するアセンブリリスト
Dim asm(9) As Variant
asm(0) = "System.dll"
asm(1) = "System.Core.dll"
asm(2) = "System.Data.dll"
asm(3) = "System.Data.DataSetExtensions.dll"
asm(4) = "System.Drawing.dll"
asm(5) = "System.Windows.dll"
asm(6) = "System.Windows.Forms.dll"
asm(7) = "System.Xml.dll"
asm(8) = "System.Xml.Linq.dll"
asm(9) = "Microsoft.CSharp.dll"
MesObj.AddConfig "Script.References", asm
```

４-２ クラス

Simple Measure で使用する主なクラスです。

SimpleMeasureLib（COM）

ライブラリ名 SimpleMeasureCom

表示名 SimpleMeasure COM Library

メソッド

Sub　About()

バージョン情報を表示します。

Sub　AddConfig(key As String, value As Variant)

環境を設定します。

パラメータ

key

Type：String

環境変数名を設定します。

value

Type：Variant

環境変数に設定する値を設定します。

Sub　CallScript(script As String, param As String)

C#マクロを実行します。

CsForApp フォームがモードレスで実行されるときに使用します。

パラメータ

Script

Type：String

C#ソースコード。

param

Type：String

C#ソースコードの object[] Main(string value)の value へ渡される値。

Sub　CsForApp(script As String, param As String)

C#マクロを有効にします。CsForApp フォームを表示します。

script が設定されている（空文字以外の）場合はそのコードをフォーム表示後に実行します。

パラメータ

Script

Type：String

空白文字で CsForApp フォームを表示するのみ。C#ソースコードが設定されている場合はそのフォーム表示後コードを実行する。

param

Type：String

C#ソースコードの object[] Main(string value)の value へ渡される値。

Sub　Measure()

Measure フォームを表示します。

Sub　ScriptTest()

ScriptTest フォームを表示します。

プロパティ

Property　Busy As Boolean

実行状態を返します。

MeasureScriptBase クラス

名前空間 SimpleMeasure

メソッド

object[]　Main(string value)

スクリプトを実行したときに、最初に呼び出されるメソッドです。

パラメータ

value

Type：string

Measure シートの Parameter2 の値が設定されます。

戻り値

Type：object[]

Data シートに書き込む値を設定します。書き込まれるフィールドは object[0]が Data1 に、object[1]が Data2 になります。null を返すと書き込みは行われません。

プロパティ

string　LastData

最後に Device から読み取った値の取得。カンマまたはセミコロン区切りで分離される前の、ReadString()で読み取ったときと同じ値を返します。

スクリプトと Dummy デバイスを使用したときのみ、任意の文字列の設定が可能です。C#マクロでは無効です。

Dictionary<string, object>　Var

変数を管理するコンテナを取得します。DevVariable（Variable Device）はこのコンテナを使用します。DevVariable で操作できない場合は、このオブジェクトを直接使用します。

読み取り専用で設定はできません。

フィールド

Dictionary<string, object>　Config

環境設定コンテナです。VBA からも設定できます。プログラム実行時に使用されます。実行後は変更できません。読み取りのみ有効です。

DeviceObjects　DevObj

Device シートに登録されたデバイスが保存されているコンテナです。

dynamic　ExcelApp

Excel PIA の Application クラスオブジェクトです。
.NET 4.0 未満では object 変数になります。
プロパティ、メソッドについては Microsoft の Microsoft.Office.Interop.Excel のドキュメントを参照してください。

dynamic　ExcelBook

Excel PIA の Workbook クラスオブジェクトです。
.NET 4.0 未満では object 変数になります。
プロパティ、メソッドについては Microsoft の Microsoft.Office.Interop.Excel のドキュメントを参照してください。

MeasureStatus　Status

実行状態を管理するオブジェクトです。

object　Variable

システム予約オブジェクトです。

Device クラス

名前空間 SimpleMeasure

インスタンス DeviceObjects 内

概要

各デバイスの基底クラスです。Device シートで設定している Device は派生クラスとなります。デバイス名と派生クラス名は次のようになります。有効なプロパティとメソッドは「３-２ Device の詳細」で有効なもののみですが、DevVisa クラスは、ReadByte(int count)と Write(byte[] value)が使用できます。

デバイス名	クラス名
Dummy	DevDummy
End	DevEnd
Input	DevInput
Message	DevMessage
Picture	DevPictureMessage
Variable	DevVariable
VISA	DevVisa
Wait	DevWait

プロパティ

int Timeout { get; set; }

タイムアウトの取得・設定をします。

メソッド

void Clear()

デバイスをクリアします。

byte[] ReadByte(int count)

デバイスからバイナリーで count バイト読み取ります。

パラメータ

count

Type：int

読み取るバイト数。

戻り値

Type：byte[]

読み取った byte 配列。

string　ReadString()

デバイスから文字列を読み取ります。

戻り値

Type：string

読み取った文字列。

string　ReadString(string value)

デバイスに value を書き込んだ後に読み取ります。

パラメータ

value

Type：string

書き込む文字列。

戻り値

Type：object[]

読み取った文字列

void　Write(string value)

デバイスに文字列を書き込みます。

パラメータ

value

Type：string

書き込む文字列。

void　Write(byte[] value)

デバイスにバイト配列データを書き込みます。

パラメータ

value

Type：byte[]

書き込む byte 配列データ。

DeviceObjects クラス

名前空間 SimpleMeasure

インスタンス DevObj

概要

Device シートで登録されたオブジェクトと Measure シートの Device フィールドで設定されたオブジェクトを管理します。

フィールド

Device　CurrentDevice

現在のデバイス（Measure シートの Device フィールドに設定されたオブジェクト）を取得します。C#マクロでは無効です。

Dictionary<string, Device>　DeviceList

Device シートで登録されたオブジェクトが保存されているコンテナです。

MeasureStatus クラス

名前空間 SimpleMeasure

インスタンス Status

概要

プログラムの状態を管理します。現在実行中の Measure シートのインデックスやキャンセルの状態の取得・設定を行います。

プロパティ

bool　Cancel { get; set; }

現在行っている処理の制御状態の取得・設定をします。true に設定すると中断処理を行います。

Int　Index { get; set; }

現在測定を行っている Measure シートの Index 値の取得・設定をします。最初の項目を 1 とする連番が設定されます。値を設定する場合、その項目番号から 1 を減じた値に設定します。0 に設定すると最初の項目 1 に戻ります。C#マクロでは無効です。

Excel オブジェクトを使用する場合の注意

.NET Version v4.0 の場合で Script から ExcelApp や ExcelBook などの Excel object を使用するには参照に次のアセンブリを追加する必要があります。

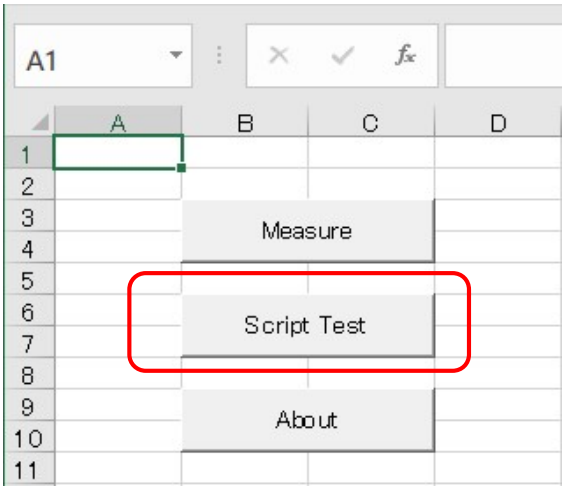
```
System.dll
System.Core.dll
Microsoft.CSharp.dll
```

.NET Version が v4.0 未満の場合、dynamic 型は使用できないので、次のように Invoke()でメソッドを呼び出す必要があります。また、参照アセンブリの System.Core、Microsoft.CSharp は設定できません。

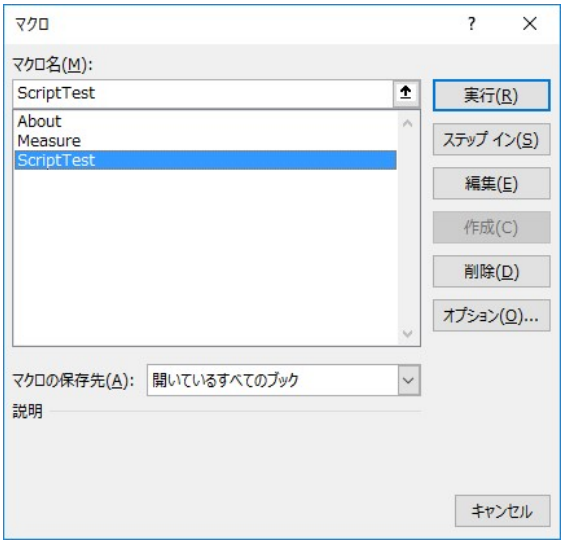
```
MethodInfo info = myType.GetMethod("Function_Name");
info.Invoke(excelObject, new object[] { Parameter_List });
```

4-3 ScriptTest フォーム

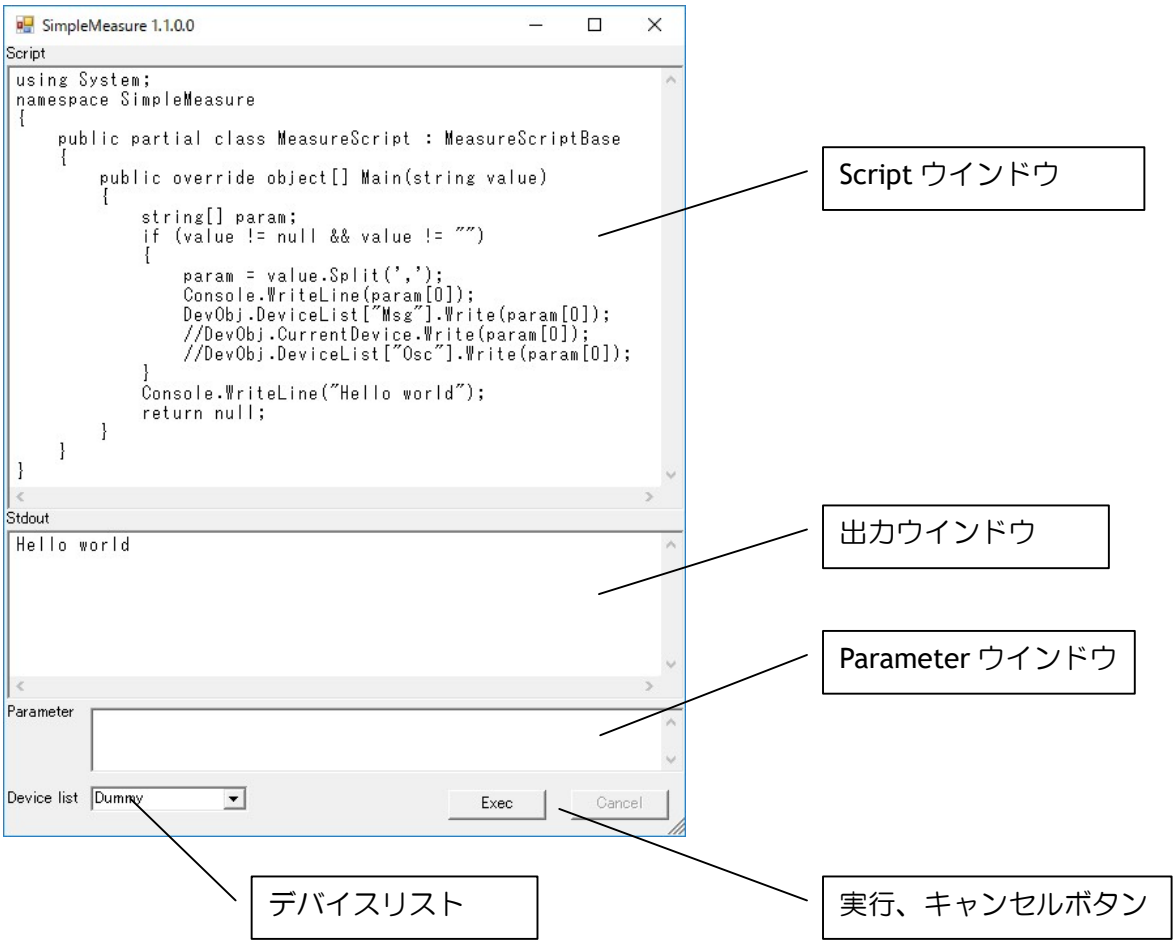
スクリプトのテストやリモートコマンドの確認などに使用します。
Menu シートの「Script」ボタンを押すと Script フォームが開きます。



このボタンは、標準モジュール「SimpleMeasure」の「Public Sub ScriptTest()」を呼び出しています。[開発][マクロ]から呼びだすこともできます。



ScriptTest フォーム



Script と Parameter ウインドウはそれぞれ、Measure シートの Parameter と Parameter2 フィールドに設定した時と同じ動作をします。
出力ウインドウは Measure フォームの出力ボックスに相当します。Console.WriteLine()等で出力することができます。
デバイスリストは Device シートに登録したオブジェクトがリストされ、選択されたオブジェクトが CurrentDevice に登録されます。Measure シートの Device に相当します。

４-４ C#マクロ

C#をマクロとして使用します。Measure のスクリプトとの違いは、Measure フォームを表示し実行中でなければスクリプトを実行できませんが、C#マクロは VBA のマクロと同じように任意のタイミングで実行できます。
VBA では実現が困難なコードが.NET ライブラリを使用して実現できます。

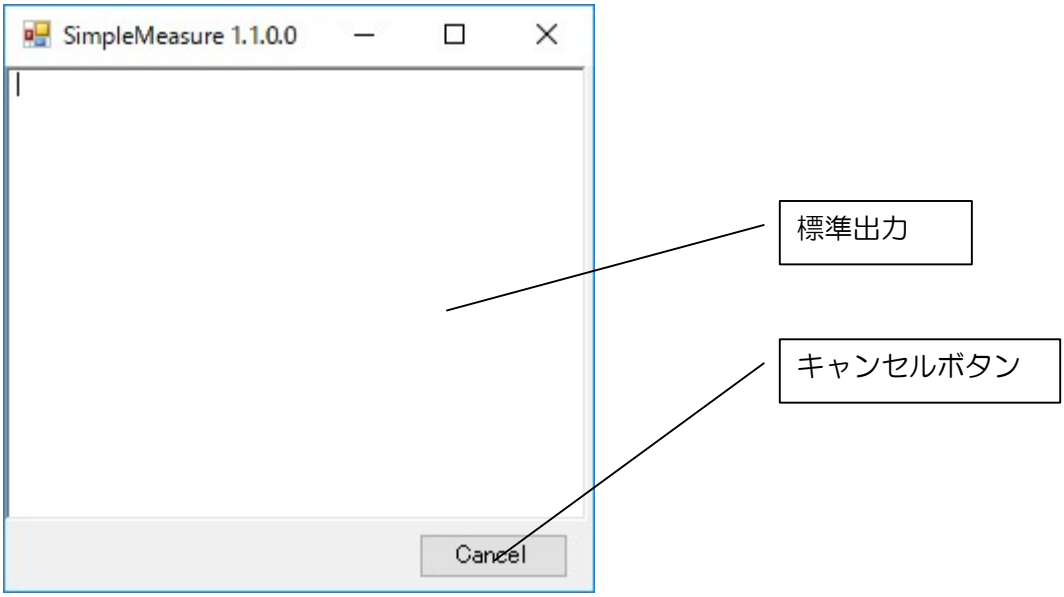
マクロの実行モードはモーダルとモードレスモードがあります。CsForApp フォームが表示された状態で、Excel の操作ができないのがモーダル、操作できるのがモードレスです。
モードの設定は「４-５ 動作環境設定」で
MesObj.AddConfig "System.FormCsForApp.Modal", False
で、True でモーダル、False でモードレスに設定されます。

マクロの実行は、モーダルでは CsForApp()を、モードレスは CsForApp()と CallScript()を呼び出します。

C#マクロの実行（モーダル）

VBA の CsForApp(script, param) を呼び出します。script に C#マクロのソースコード、必要に応じて param にその引数を設定します。CsForApp フォームが表示後、マクロが実行されます。

テキストボックスは標準出力で、Console.WriteLine()等からの出力が表示されます。
Cancel ボタンは Status.Cancel を true に設定します。ループを中断させるときに押します。
中断を有効にするには、次のように Status.Cancel が true になったら中断するようにします。
While (!Status.Cancel) { 処理 … }



標準モジュール：

SimpleMeasure

メソッド

Sub　CsForApp(script As String, param As String)

C#マクロを有効にします。CsForApp フォームを表示します。

script が設定されている（空文字以外の）場合はそのコードをフォーム表示後に実行します。

パラメータ

Script

Type：String

C#ソースコード。

param

Type：String

C#ソースコードの object[] Main(string value)の value へ渡される値。

C#マクロの記述は「4-1 スクリプト言語仕様」「4-2 クラス」を参照してください。

C#マクロサンプル

1. ワークシート「MacroSample」を作成して B1 に以下のコードを貼り付けます

	A	B

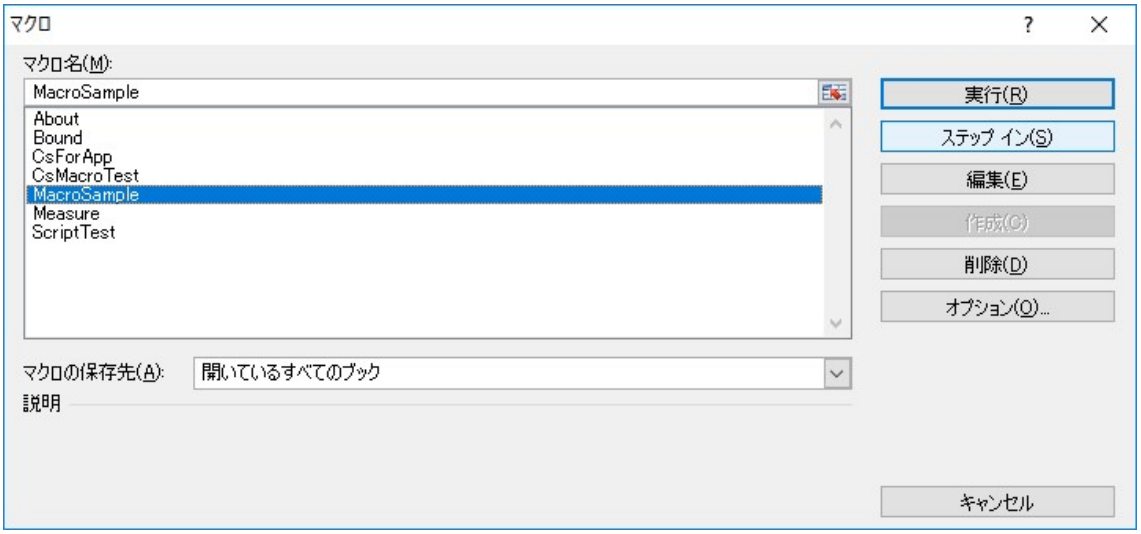
1	マクロコード	// Hello world サンプルスクリプト
		using System;
		namespace SimpleMeasure
		{
		　　public partial class MeasureScript : MeasureScriptBase
		　　{
		　　　　public override object[] Main(string value)
		　　　　{
		　　　　　　Console.WriteLine(value);
		　　　　　　return null;
		　　　　}
		　　}
		}

2		

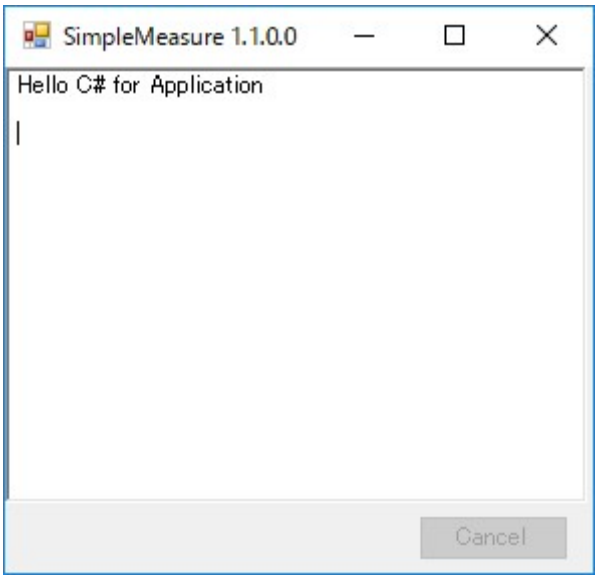
2. VBA の標準モジュールを作成して以下のコードを貼り付けます。

```
Public Sub MacroSample()
    CsForApp Sheets("MacroSample").Range("B1").Value, "Hello C# for Application"
End Sub
```

3. [開発][マクロ]で「MacroSample」を選択して[実行]ボタンを押します。



4. 実行すると CsForApp のフォームに以下のように表示されます。



ここに表示されている文字は、
CsForApp Sheets("MacroSample").Range("B1").Value, "Hello C# for Application"
の第 2 引数で指定した文字列が表示されます。

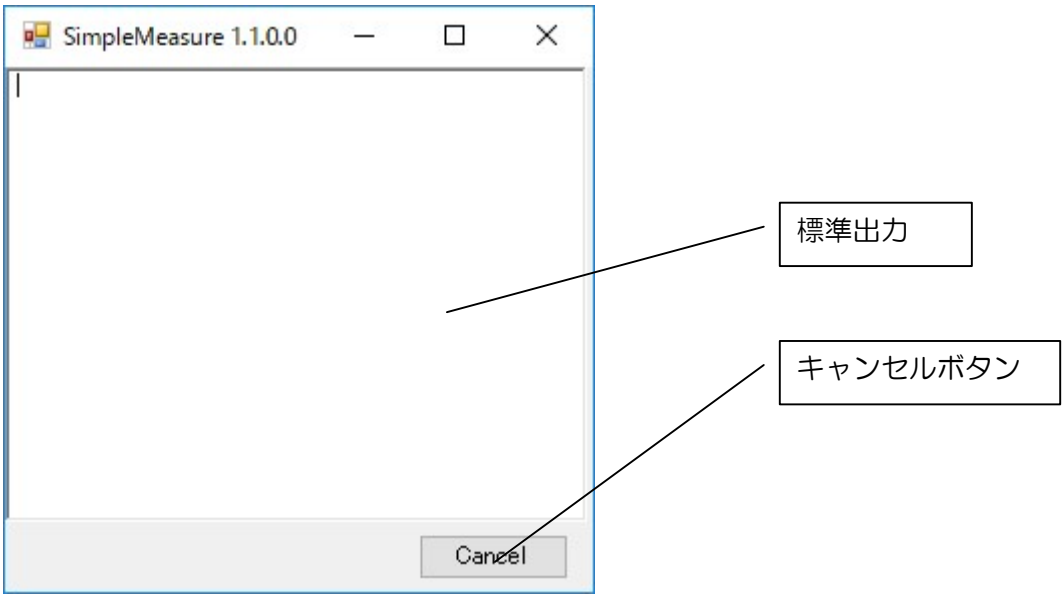
C#マクロの実行（モードレス）

VBA の CsForApp(“”, “”) を呼び出すと、次のフォームが表示されます。このフォームが開かれているときに C#マクロが実行できます。テキストボックスは標準出力で、Console.WriteLine()等からの出力が表示されます。

Cancel ボタンは Status.Cancel を true に設定します。ループを中断させるときに押します。

中断を有効にするには、次のように Status.Cancel が true になったら中断するようにします。

While (!Status.Cancel) { 処理 … }



コードの実行

C#マクロの実行は、VBA の Public Sub CallScript() を呼び出します。
メソッド CsForApp()については「C#マクロの実行（モーダル）」を参照してください。

標準モジュール：
SimpleMeasure
メソッド
Sub CallScript(script As String, param As String)

C#マクロを実行します。
C#マクロが起動していない状態で CallScript()を呼び出した場合、自動的に CsForApp(“”, “”)が呼び出されます。
C#マクロの記述は「４-１ スクリプト言語仕様」「４-２ クラス」を参照してください。

パラメータ
script：C#マクロのソースコード。
Type：String
C#ソースコード。
Param
Type：String
C#ソースコードの object[] Main(string value)の value へ渡される値。

C#マクロサンプル

1. ワークシート「MacroSample」を作成して B1 に以下のコードを貼り付けます

	A	B

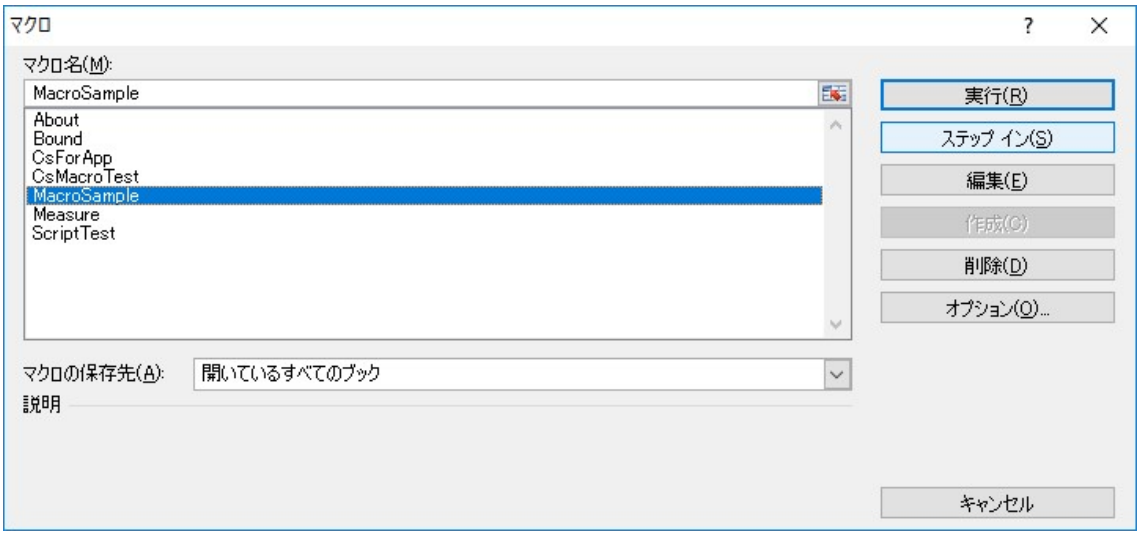
1	マクロコード	// Hello world サンプルスクリプト
		using System;
		namespace SimpleMeasure
		{
		public partial class MeasureScript : MeasureScriptBase
		{
		public override object[] Main(string value)
		{
		Console.WriteLine(value);
		return null;
		}
		}
		}

2		

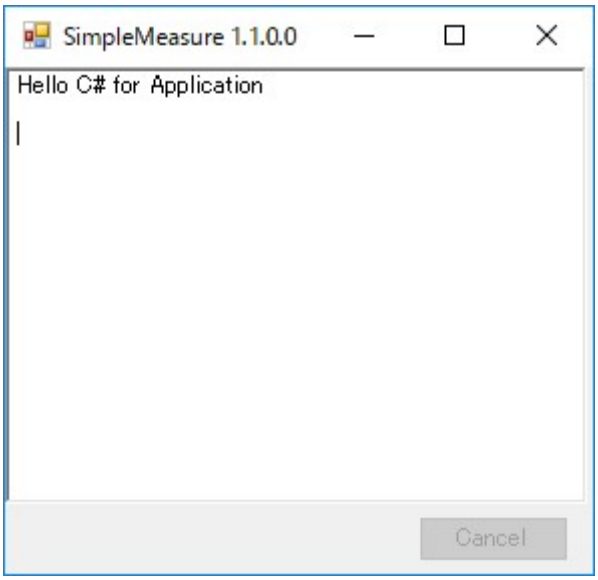
2. VBA の標準モジュールを作成して以下のコードを貼り付けます。

```
Public Sub MacroSample()
    CallScript Sheets("MacroSample").Range("B1").Value, "Hello C# for Application"
End Sub
```

3. [開発][マクロ]で「MacroSample」を選択して[実行]ボタンを押します。



4. 実行すると CsForApp のフォームに以下のように表示されます。



ここに表示されている文字は、
CallScript Sheets("MacroSample").Range("B1").Value, "Hello C# for Application"
の第 2 引数で指定した文字列が表示されます。

4-5 動作環境設定

Measure フォームの最上位表示やカーソル移動の有無などが設定できます。
[開発][Visal Basic]で VBA の IDE を開きます。

以下のコードは標準モジュールの SimpleMeasure のメソッド Private Sub SetConfig() 内に記述してください。

System

```
MesObj.AddConfig "System.StdoutMaxlength", CLng(65536)  'Textbox に出力する最大文字数

'表示サイズ 0:規定値、1:最小、2:最大
'FormMeasure の表示サイズを指定する
MesObj.AddConfig "System.FormMeasure.FormWindowState", CLng(0)
'FormScript の表示サイズを指定する
MesObj.AddConfig "System.FormScript.FormWindowState", CLng(0)
'FormCsForApp の表示サイズを指定する
MesObj.AddConfig "System.FormCsForApp.FormWindowState", CLng(0)

MesObj.AddConfig "System.FormMeasure.TopMost", False    'FormMeasure を最上位に表示する
MesObj.AddConfig "System.FormScript.TopMost", False     'FormScript を最上位に表示する
MesObj.AddConfig "System.FormCsForApp.TopMost", False   'FormCsForApp を最上位に表示する

MesObj.AddConfig "System.FormMeasure.Modal", True       'FormMeasure のモーダル表示
MesObj.AddConfig "System.FormScript.Modal", True        'FormScript のモーダル表示
MesObj.AddConfig "System.FormCsForApp.Modal", False     'FormCsForApp のモーダル表示

MesObj.AddConfig "System.Measure.Cursol", True          'Measure のカーソルの有効化
MesObj.AddConfig "System.Data.Cursol", True             'Data のカーソル有効化

MesObj.AddConfig "System.FormMeasure.TimeInterval", CLng(1000)  'FormMeasure の更新間隔
MesObj.AddConfig "System.FormScript.TimeInterval", CLng(200)    'FormScript の更新間隔
MesObj.AddConfig "System.FormCsForApp.TimeInterval", CLng(200)  'FormCsForApp の更新間隔
```

VISA

```
'VISA ReadString()の引数で、入力バッファサイズの規定値。
MesObj.AddConfig "Visa.ReadBufferSize", CLng(65536)
```

Script

```
‘コンパイラ・ワーニング・レベル。-1 でデフォルトのレベル 4。
MesObj.AddConfig "Script.WarningLevel", CLng(-1)
‘Script で使用する.NET フレームワークのバージョン。v2.0、v3.5、v4.0 が設定可能。
MesObj.AddConfig "Script.FrameworkVersion", "v4.0"
‘Script フォームで起動時に表示されるコード
MesObj.AddConfig "ScriptTest.DefaultText", _
    "using System;" & vbCrLf & _
    "namespace SimpleMeasure" & vbCrLf & _
    "{" & vbCrLf & _
    "    public partial class MeasureScript : MeasureScriptBase" & vbCrLf & _
    "    {" & vbCrLf & _
    "        public override object[] Main(string value)" & vbCrLf & _
    "        {" & vbCrLf & _
    "            if (value != null && value != """")" & vbCrLf & _
    "            {" & vbCrLf & _
    "                Console.WriteLine(""Parameter="" + value);" & vbCrLf & _
    "                DevObj.CurrentDevice.Write(value);" & vbCrLf & _
    "            }" & vbCrLf & _
    "            else" & vbCrLf & _
    "            {" & vbCrLf & _
    "                Console.WriteLine(""Hello world"");" & vbCrLf & _
    "            }" & vbCrLf & _
    "            return null;" & vbCrLf & _
    "        }" & vbCrLf & _
    "    }" & vbCrLf & _
    "}"
```

Script で使用するアセンブリリスト

```
Dim asm(9) As Variant
asm(0) = "System.dll"
asm(1) = "System.Core.dll"
asm(2) = "System.Data.dll"
asm(3) = "System.Data.DataSetExtensions.dll"
asm(4) = "System.Drawing.dll"
asm(5) = "System.Windows.dll"
asm(6) = "System.Windows.Forms.dll"
asm(7) = "System.Xml.dll"
asm(8) = "System.Xml.Linq.dll"
asm(9) = "Microsoft.CSharp.dll"
MesObj.AddConfig "Script.References", asm
```

データシートの表示更新

Excel 2013 からシングル・ドキュメント・インターフェース（SDI）表示になり、シートの表示の切り替えに時間がかかるようになりました。この対策としては「表示の更新レートを低くする」、「シートを切り替えない」の 2 通りの方法があります。

表示の更新レートを低くする

```
MesObj.AddConfig "System.FormMeasure.TimeInterval", CLng(1000)
```

の値を大きくしてください。この値を大きくすると、カーソルが数行ごとに移動する動作になりますが、実行速度の低下は少なくなります。単位は[msec]です。

カーソル移動を無効化

```
MesObj.AddConfig "System.Data.Cursol", False
```

データシートのカーソル移動を無効化します。Measure シートのカーソルは有効です。
表示の更新レートも

```
MesObj.AddConfig "System.FormMeasure.TimeInterval", CLng(300)
```

程度にでき、スムーズにカーソルが移動します。単位は[msec]です。

グラフの更新

Excel 2007 ではシートのデータを更新すると、グラフも更新されていましたが、Excel 2016（2010、2013 は未確認）では更新されなくなりました。
Measure フォームをモードレス表示にすることで対応できますが、副作用が多いので、「どうしてもグラフのリアルタイム表示をしたい」という場合に限定して設定してください。

```
MesObj.AddConfig "System.FormMeasure.Modal", False
```

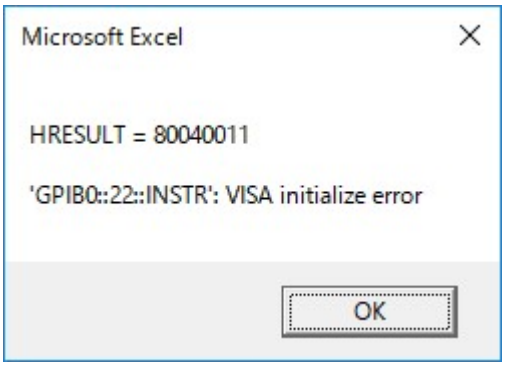
マウスやキーボードから Excel の操作が測定実行中に行えるため、データの書き込みやカーソル表示などの操作が衝突したときは実行を停止しますので注意してください。

５ エラー

５-１ Measure または ScriptTest を起動した直後に発生するエラー

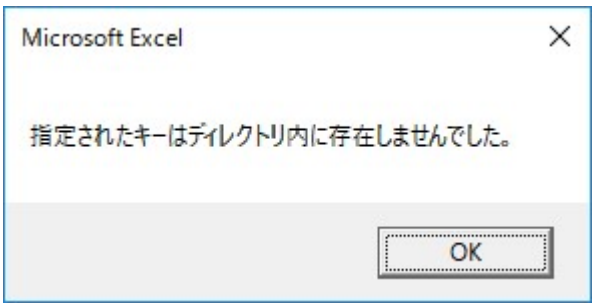
Measure()または ScriptTest()マクロを呼び出した直後に発生するエラーで、それぞれのフォームが表示される前に実行が中断します。

VISA 初期化エラー



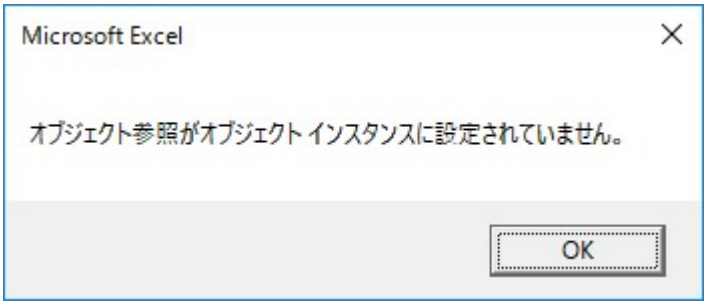
Error 0x80040011: Specified expression does not match any devices.
VISA デバイスが接続されていない場合に発生します。Device シートに登録されているデバイスの接続確認を行ってください。
この例では GPIB ボード 0、アドレス 22 のデバイスの初期化でエラーが発生しています。
”GPIB::22::INSTR”は Excel の Device シートの Resource フィールドで設定している値です。
USB などのほかのインターフェースの場合はこの部分の表示が変わります。

Device 名が存在しない



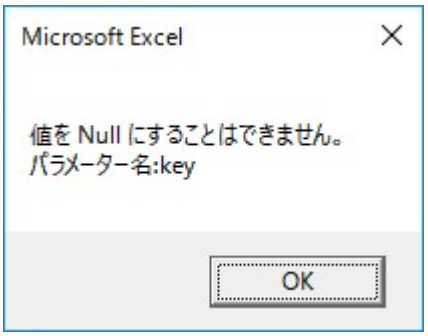
Device シートの Device フィールドで設定されているデバイスが存在しない場合に発生します。
設定できる値は「３-２ Device の詳細」を参照してください。

Device シートの Name フィールドが空欄



Device シートの Name フィールドに重複しない値を設定してください。この項目は省略できません。

Device シートの Device フィールドが空欄

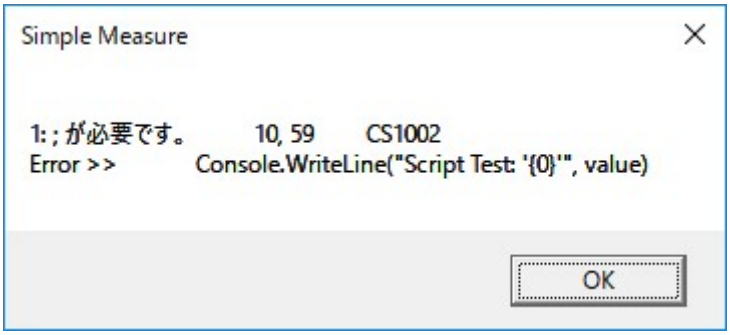


Device シートの Device フィールドに値を設定してください。設定できる値は「３-２ Device の詳細」を参照してください。この項目は省略できません。

5-2 Measure または ScriptTest 実行したときに発生するエラー

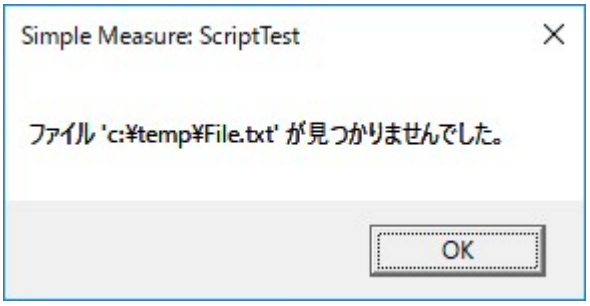
Measure フォームの Measure ボタン、または ScriptTest の Exec ボタンを押した後に発生するエラーです。

コンパイルエラー



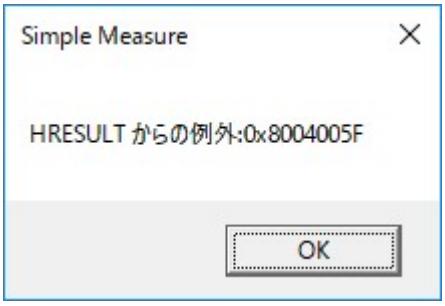
スクリプトまたは C#マクロのコードにエラーがある場合表示されます。
エラーの内容についてはソースコードの内容によって変わります。
1 行目がエラー内容で、
　　項目番号:　 内容　 行, 列　 エラーコード
2 行目はエラーの発生したコードが表示されます。
エラーコードの CSxxxx(x は数値)の詳細につきましては、MSDN の「C#コンパイルエラー」を参照してください。
複数のエラーが発生した場合は、エラーの数だけ表示されます。

実行時の一般エラー

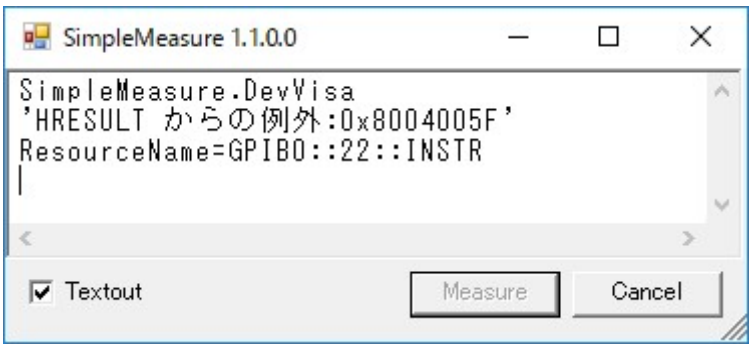


スクリプトまたは C#マクロの実行時に例外が発生した場合表示されます。
エラーの内容はその例外内容によって変わります。
このエラーは次のコードでファイルが存在しない場合のエラーです。
FileStream fs = File.Open(@"c:¥temp¥File.txt", FileMode.Open);

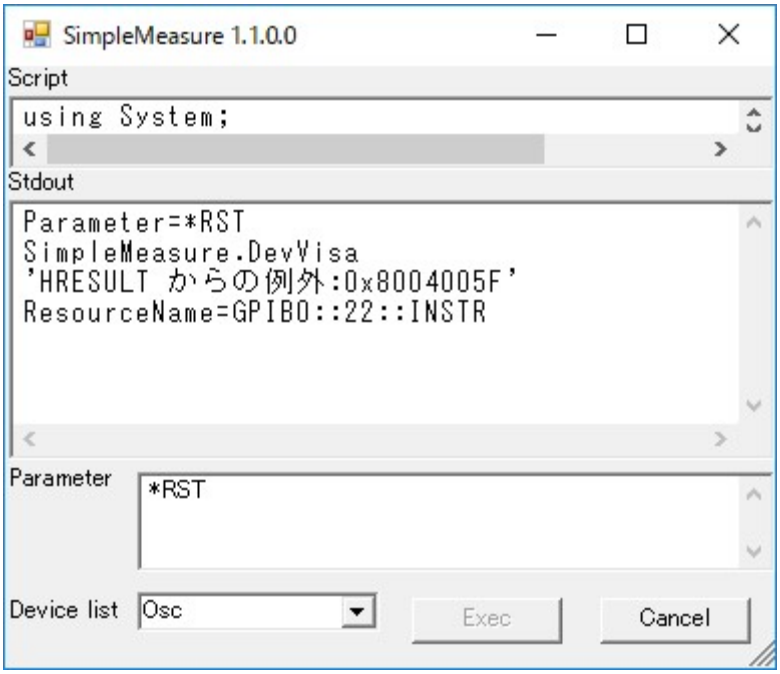
VISA デバイスエラー（GPIB）



Measure フォーム



ScriptTtest フォーム



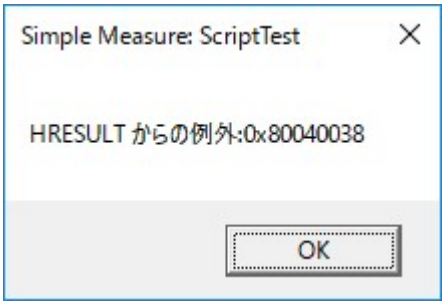
Error 0x8004005f: No listeners condition is detected (both NRFD and NDAC are deasserted).

VISA の GPIB デバイスに Write または Read 操作を行ったときに発生します。

デバイスが接続されていない場合にエラーが発生します。

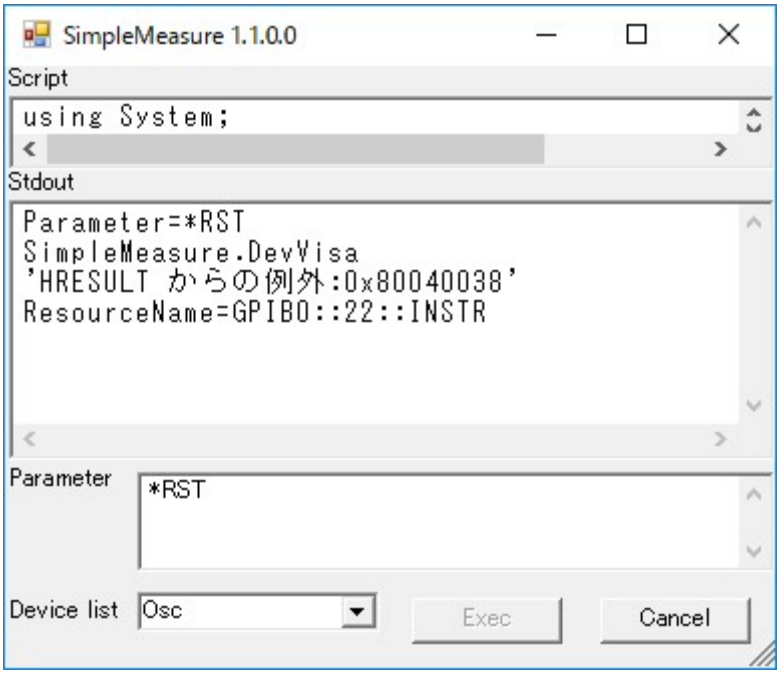
Device シートの Resource フィールドの設定と接続されている測定器の接続を確認してください。

VISA デバイスエラー



Measure フォーム

ScriptTest フォーム

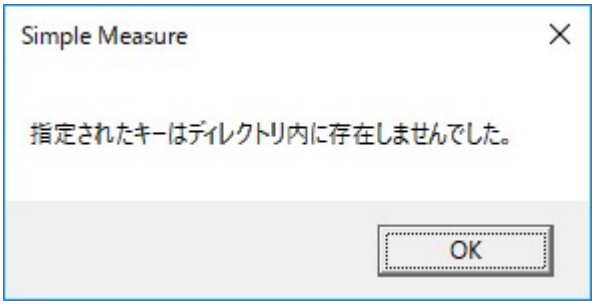


Error 0x80040038: Bus error occurred during transfer

VISA デバイスが接続されていない、または GPIB のアドレスや USBTMC の Serial、LAN の IP アドレスなどが設定と異なる場合にエラーが発生します。

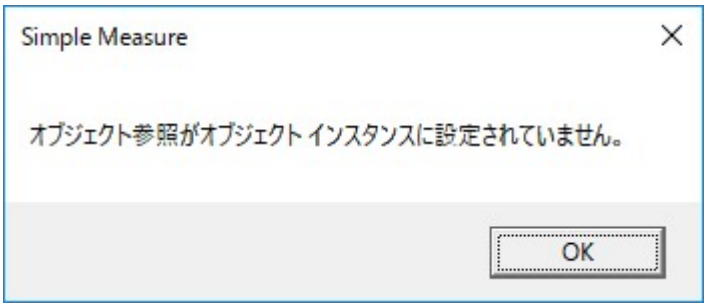
Device シートの Resource フィールドの設定と接続されている測定器の設定が一致しているか確認してください。

Measure シートの Method または Device が存在しない



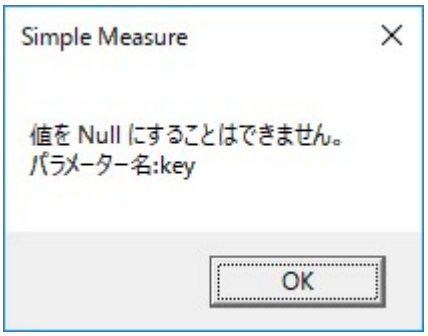
Measure シートの Method フィールドや Device フィールドが設定できない値になっている場合に発生します。設定できる Method は「3-3 試験項目の設定（Measure シート）」を参照してください。
また、Measure シートの Device フィールドに設定できる値は、Device シートの Name フィールドで設定されている値です。Measure シートの Device フィールドと Device シートの Name フィールドの値が一致しているか確認してください。

Measure シートの Method フィールドが空欄になっている



Method フィールドに値を設定してください。この項目は省略できません。

Measure シートの Device フィールドが空欄になっている



Device フィールドに値を設定してください。この項目は省略できません。