Simple Measure

Version 1.1

C#マクロの使い方

Automatic Measurement Systems

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

内容

概要

Simple Measure での C#スクリプトと C#マクロは実行形態が異なるだけで、機能面では同じです。ここでは C#マクロの具体的な使用方法を説明します。

Excel は Visual Basic インタープリタ（VBA）をマクロ言語として使用しています。
Simple Measure は、より高度な処理が可能な C#マクロプラグインが実装されています。
このプラグインは Excel で C#マクロが使用できるようになります。

使用できるオブジェクトなどは「Simple Measure 取扱説明書」の「スクリプト」を参照してください。

マクロの有効化と実行

マクロの実行や操作を行うには VBA の IDE を使用します。その操作は[開発]タブから操作しますが、[開発]タブが表示されていない場合は、「[開発] タブを表示する」を参照して表示させてください。

マクロは、モーダルとモードレスで実行できます。モーダルでは実行中に Excel の操作は行えません。モードレスでは Excel の操作が行えますが、マウスやキーボード操作と衝突した場合、実行が中断される場合があります。
Excel2016 ではグラフ表示などをリアルタイムで行う場合、モードレスで実行する必要があります。
用途に応じてモードを指定してください。

モードの指定は、VBA の標準モジュール SimpleMeasure の SetConfig()で
MesObj.AddConfig "System.FormCsForApp.Modal", True　　'FormCsForApp のモーダル表示
を True でモーダル、False でモードレスに設定できます。

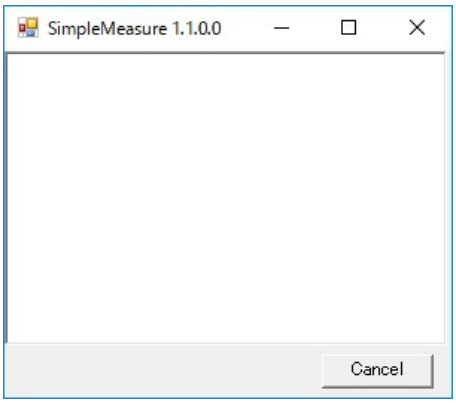
C#マクロのコードで、Console.WriteLine()などの標準出力への書き込みは、CsForApp フォームのテキストボックスに出力されます。

モーダルで実行

VBA の標準モジュール SimpleMeasure の CsForApp(script as string, param as string)を呼び出します。
引数 script にプログラムコード、必要に応じて param にその引数を設定します。

モードレスで実行

VBA の標準モジュール SimpleMeasure の CsForApp(script as string, param as string)を呼び出します。
このとき引数 script にプログラムを設定すると、そのコードをすぐに実行します。
空文字を指定すると CsForApp フォームが表示した状態になります。



このフォームが開かれている状態では、いつでも C#マクロが実行できます。

マクロの実行は、VBA の標準モジュール SimpleMeasure の CallScript(script As String, param As String)を呼び出します。script にプログラムを設定し、必要に応じて param にその引数を設定します。

サンプル

CsForApp フォームに文字を表示するサンプルは以下のようになります。

１．実行モードの指定
VBA の標準モジュール SimpleMeasure の SetConfig()で
MesObj.AddConfig "System.FormCsForApp.Modal", True　　'FormCsForApp のモーダル表示
を True でモーダル、False でモードレスに設定できます。

２．MacroSample シートを作成します。
そのシートのセル"B1"に次のコードを記述します。

```
// 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;
        }
    }
}
```

３．次に VBA で新しいモジュールを作成し（例えば Module1)、そこに次のコードを記述します。
このコードは、上のコードを呼び出すための処理です。この例ではセルに保存されたコードを実行しますが、ファイルなどに保存されたコードでも同じように実行できます。モーダルでは CsForApp()を、モードレスでは CsForApp()と CallScript()を呼び出します。モードレスで CsForApp()を呼び出さずに CallScript()を呼び出した場合、CsForApp("", "")が自動的に呼び出されます。

モーダルの場合
VBA の標準モジュール SimpleMeasure の SetConfig()で
MesObj.AddConfig "System.FormCsForApp.Modal", True　　'FormCsForApp のモーダル表示

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

モードレスの場合１
VBA の標準モジュール SimpleMeasure の SetConfig()で
MesObj.AddConfig "System.FormCsForApp.Modal", False　　'FormCsForApp のモーダル表示

```
Public Sub MacroSample1()
    'CsForApp("", "")が自動的に呼び出されます。
```

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

モードレスの場合 2

VBA の標準モジュール SimpleMeasure の SetConfig()で

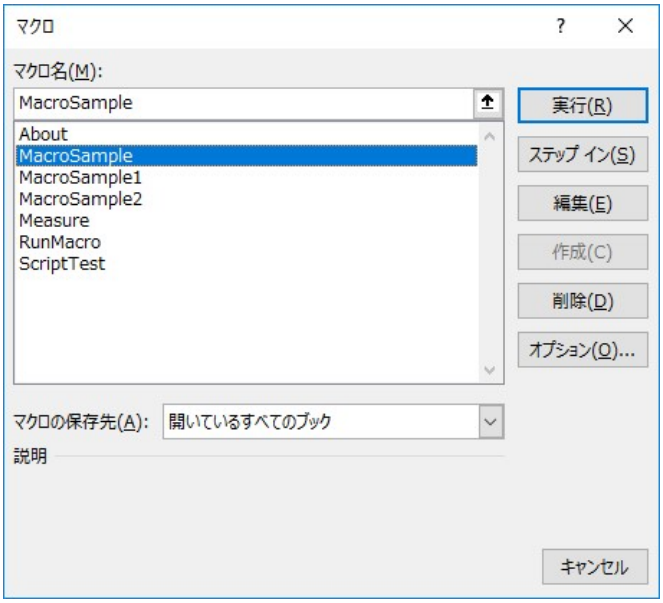
```
MesObj.AddConfig "System.FormCsForApp.Modal", False          'FormCsForApp のモーダル表示

'マクロを有効化
Public Sub MacroSample2()
    CsForApp "", ""
End Sub
'マクロの実行
Public Sub RunMacro()
    CallScript Sheets("MacroSample").Range("B1").Value, "Hello C# for Application"
End Sub
```

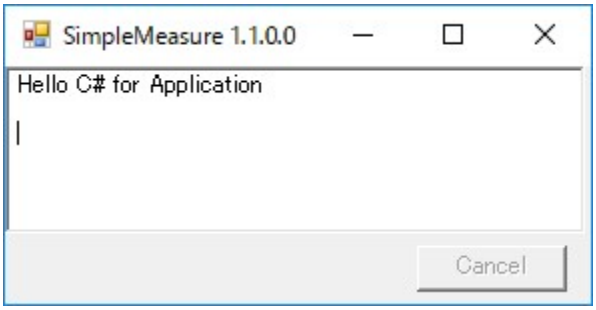
４．マクロを実行する

モーダルの場合

[開発][マクロ]で[MacroSample]を選択して[実行]ボタンを押す。



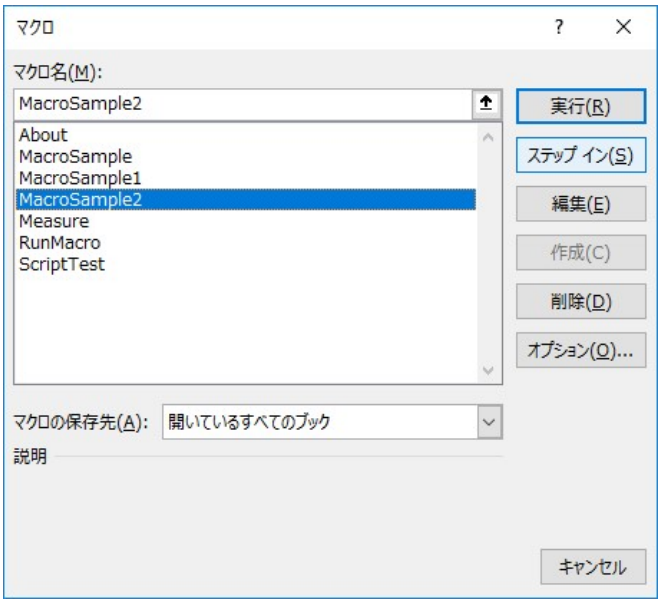
実行結果は次のようになります。



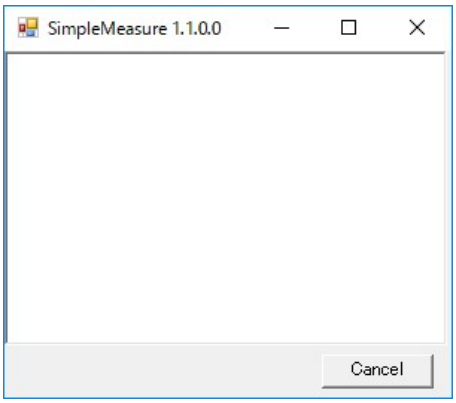
モードレスの場合

マクロを CsForApp()、CallScript()の 2 段階で呼び出します。

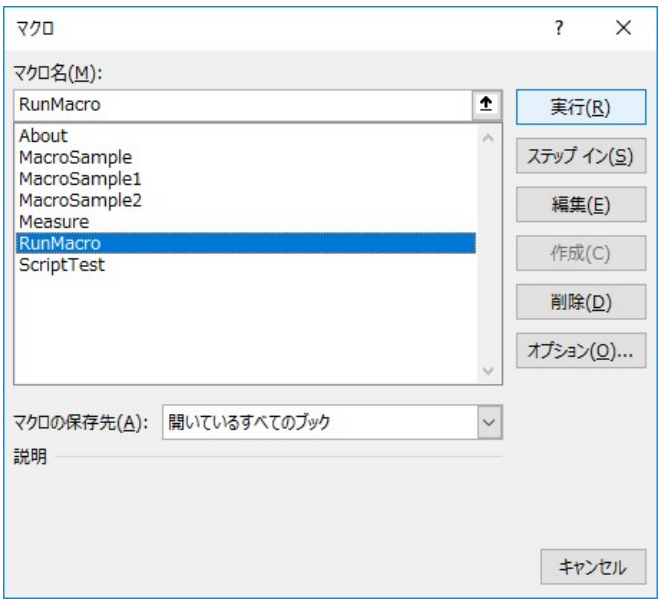
[開発][マクロ]で[MacroSample2]を選択して[実行]ボタンを押す。



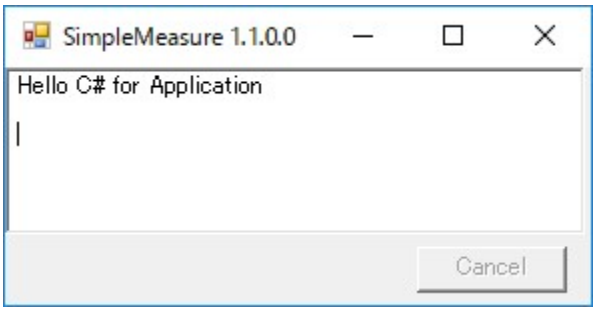
CsForApp フォームが表示されます。



RunMacro()を呼び出します。



実行結果は次のようになります。



[開発] タブを表示する

Excel 2016

[ファイル]［オプション］[リボンのユーザー設定］を開き、［メイン タブ］の下の［開発］チェック ボックスをオンにします。

Excel 2007

[オプション］［基本設定]で「 Excel の使用に関する基本オプション」の枠の「[開発]タブをリボンに表示する」にチェックを入れます。

Excel オブジェクトを使用する

Excel の Application と WorkBook オブジェクトがそれぞれ ExcelApp と ExcelBook 変数に割り当てられています。
これらのオブジェクトについては Microsoft Office Interop Excel を参照してください。Simple Measure ではこれらのオブジェクトを dynamic 変数へ保存しています。C# Version が v4.0 未満の場合は object 変数に保存されます。dynamic では実行時バインドを自動処理しますが、object 変数の場合は Invoke()を使って呼び出す必要があります。

次のコードはセルに値を書き込むサンプルです。
ワークシート Sheet1 のセル"A1"に"C# for Application Test"を書き込みます。

```
using System;
namespace SimpleMeasure
{
    public partial class MeasureScript : MeasureScriptBase
    {
        public override object[] Main(string value)
        {
            ExcelBook.Sheets["Sheet1"].Range["A1"].Value = "C# for Application Test";
            return null;
        }
    }
}
```

VISA デバイスを使用する

次の表のように Excel の Device シートに Osc が設定されている場合、このデバイスを使うサンプルコードです。
Osc に"Frq 1234"を書き込みます。

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		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	MsgTest	システムデバイス。メッセージボックスを表示する。
5	Var	Variable		システムデバイス。変数オブジェクト。
6	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]
7	Osc	VISA	GPIB0::22::INSTR	発振器
8	Oscillo	VISA	GPIB0::2::INSTR	オシロスコープ

```
using System;
namespace SimpleMeasure
{
    public partial class MeasureScript : MeasureScriptBase
    {
        public override object[] Main(string value)
        {
            // Osc に書き込む
            DevObj.DeviceList["Osc"].Write("Frq 1234");
            return null;
        }
    }
}
```

VISA を使用しないで C#マクロを使用する

Device シートで VISA のオブジェクトが設定されていると、その VISA インターフェースが接続されていない場合、初期化でエラーが表示され実行が中断します。
VISA の行を削除するか、Name フィールドの 1 文字目にセミコロン";"をつけて無効化してください。

DAQmx を使用する

機能

DAQmx デバイスの NI USB-6501 にデータを Port1 から読み込み、その値を Port0 に書きこみます。
Port0 の出力は内部の 4.7kΩでプルアップされたオープンコレクタに設定されます。
このサンプルでは DAQmx .NET API を使用していますが、NI I/O Trace ではサポートされていません。
送受信 データのキャプチャは行えません。

参考
http://zone.ni.com/reference/ja-XX/help/370466AD-0112/daqhelp/troubleshooting/

引数（Parameter2）
なし。

参照設定の追加

Excel VBA モジュール SimpleMeasure のメソッド Private Sub SetConfig()内で、asm(10)、asm(11)への代入を追加してください。（すでに 10,11 を使用している場合は、別のインデックスにしてもかまいません。）
このライブラリは、DAQmx のドライバをインストールするときに、「アプリケーション開発サポート」で「.NET Framework 言語サポート」を指定しないとインストールされません。また、インストールされたライブラリの場所は、デフォルトでの設定です。変更した場合はその場所を指定してください。
次のコードではファイルパスの途中で改行が入っていますが、ソースコードでは改行を入れないでください。

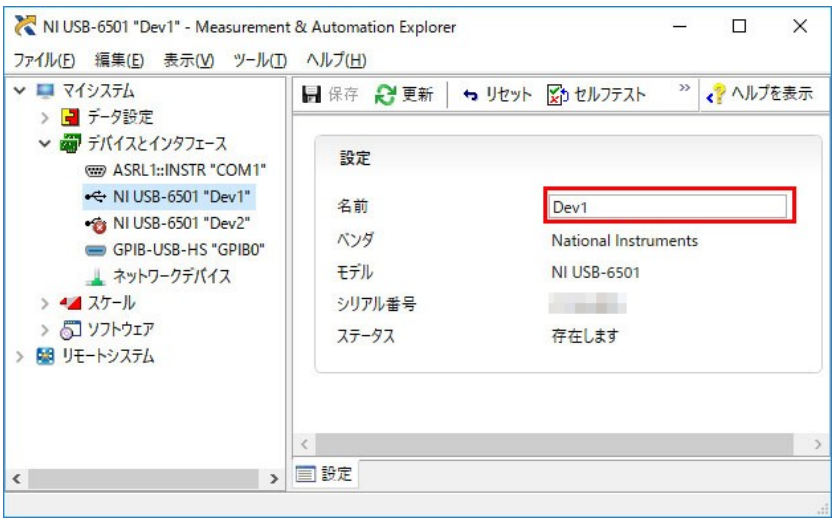
モジュール：SimpleMeasure

```
Private Sub SetConfig()
... 中略

    'Script で使用するアセンブリリスト
    'SimpleMeasure.dll はこのリストに指定する必要なし。
    Dim asm(15) 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"
    asm(10) = "C:¥Program Files (x86)¥National
Instruments¥MeasurementStudioVS2012¥DotNET¥Assemblies¥Current¥NationalInstruments.Common.dll"
    asm(11) = "C:¥Program Files (x86)¥National
Instruments¥MeasurementStudioVS2012¥DotNET¥Assemblies¥Current¥NationalInstruments.DAQmx.dll"
    MesObj.AddConfig "Script.References", asm
... 中略
```

DAQmx のデバイス名の確認

NI MAX を起動し、デバイスとインターフェースを展開すると次の図のように名前のところにデバイス名（Dev1）が表示されます。編集して名前を変更することもできます。



スクリプト

DAQmx デバイスの指定はデバイス名を使用します。次のコードでは ”Dev1” となります。

```
// DAQmx
// USB-6501
using System;
using NationalInstruments.DAQmx;
namespace SimpleMeasure
{
    public partial class MeasureScript : MeasureScriptBase
    {
        public override object[] Main(string value)
        {
            string dev = "Dev1";

            Task wtsk = new Task();
            Task rtsk = new Task();
            DOChannel cho = wtsk.DOChannels.CreateChannel(dev + "/Port0/Line0:7", "",
                        ChannelLineGrouping.OneChannelForAllLines);
            DIChannel chi = rtsk.DIChannels.CreateChannel(dev + "/Port1/Line0:7", "",
                        ChannelLineGrouping.OneChannelForAllLines);
            // cho.OutputDriveType = DOOutputDriveType.ActiveDrive;  // Active Drive
            DigitalSingleChannelWriter write = new DigitalSingleChannelWriter(wtsk.Stream);
            DigitalSingleChannelReader read = new DigitalSingleChannelReader(rtsk.Stream);

            int d = read.ReadSingleSamplePortInt32();
            write.WriteSingleSamplePort(true, d);
            Console.WriteLine("{0}", d.ToString("x"));
                        return new object[]{d};
        }
    }
}
```