bat/VBScript/PowerShellの単体テスト自動化

2015/1/7作成

佐野　尚之

アジェンダ

- セミナーのゴール
- 単体テストコードを書くために必要な知識
- テストコード(共通部分)
- bat
- VBScript
- PowerShell
- お勧めの本
- 参考URL

セミナーのゴール

- 今まで毎回手作業で行っていた単体テストがテストコードを書くことで2回目以降のテストの工数を削減できることを理解する。

単体テストコードを書くために必要な知識(1/6)

- **動作確認環境**
  - **Window 7 SP1**
  - **Visual Studio Professional 2013**
  - **PowerShell 4.0**
    - PowerShell/Windows7にPowerShell4.0をインストールする手順

http://win.just4fun.biz/PowerShell/Windows7%E3%81%ABPowerShell4.0%E3%82%92%E3%82%A4%E3%83%B3%E3%82%B9%E3%83%88%E3%83%BC%E3%83%AB%E3%81%99%E3%82%8B%E6%89%8B%E9%A0%86.html

単体テストコードを書くために必要な知識(2/6)

- **既存のプログラム**
  - 全てのコードがテストコードでテストしやすいように書かれているとは限らない。一部はテストしやすいように変える、または、手作業でテストしなければならない。
- **新規のプログラム**
  - 可能であれば最初からテストしやすい形で作る。

単体テストコードを書くために必要な知識(3/6)

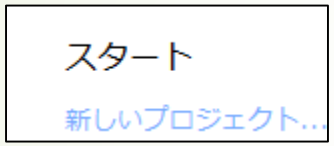
- **C#の文法で覚えておいたほうがいいこと**
- **var (※)**
  - var num = 1 // int型

※型推論（かたすいろん）

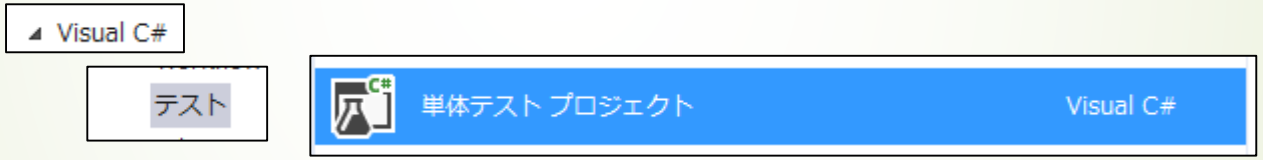
　プログラミング言語の機能の1つで、静的な型付けを持つ言語において、変数や関数の型を宣言しなくてもそれを導くのに使われた関数の型シグネチャなどから自動的に型を決定する機構のこと。

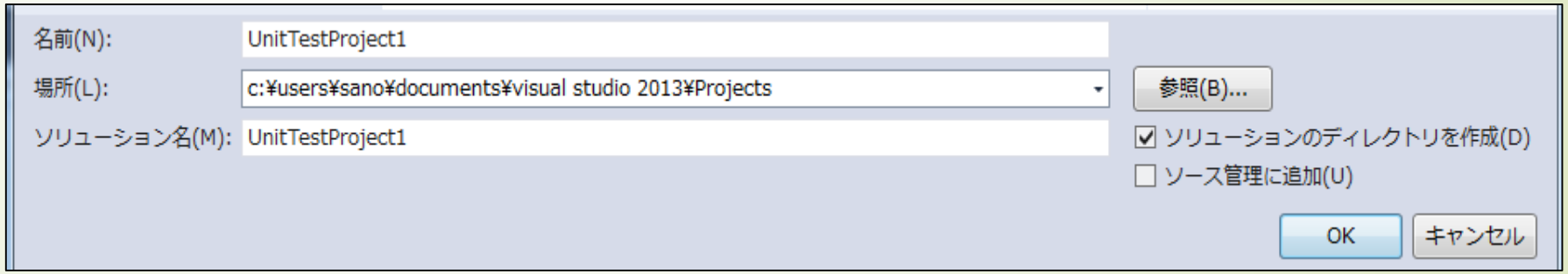
単体テストコードを書くために必要な知識(4/6)

- **Visual Studioソリューションの作り方**
  - Visual Studioを起動後に「新しいプロジェクト」をクリック



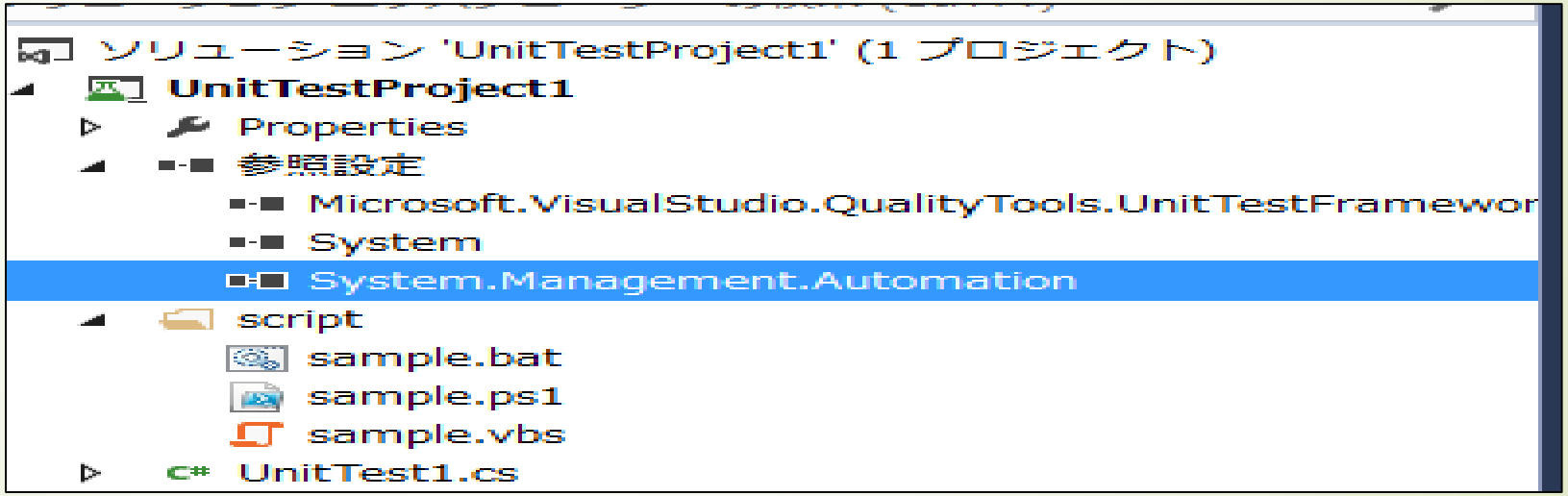
  - 「Visual C#」ー「テスト」ー「単体テスト プロジェクト」ー「OK」をクリック





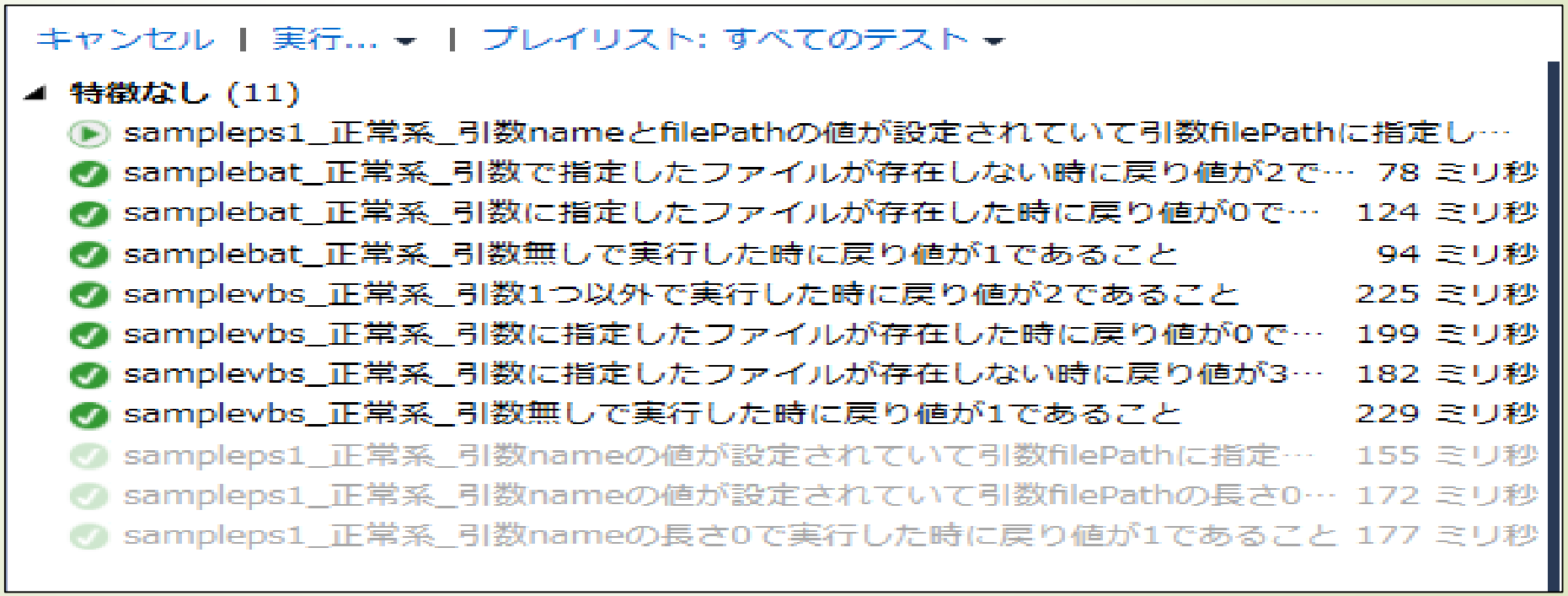
単体テストコードを書くために必要な知識(5/6)

- **Visual Studioソリューションの作り方**
  - 参照設定の追加(※)、「script」フォルダ追加、bat,ps1,vbsをscriptフォルダに追加
    - ※ C:¥Program Files (x86)¥Reference Assemblies¥Microsoft¥WindowsPowerShell¥3.0¥System.Management.Automation.dll



単体テストコードを書くために必要な知識(6/6)

- Visual Studioのテストコードの動作
  - 作成したテストメソッドは順番に動くのではなく、並列で動作することを理解して、テストコードを書く必要がある。テストメソッド毎に同じフォルダ／ファイルは使用しないほうが良い。



テストコード(共通部分) (1/2)

```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using System.Diagnostics;
using System.IO;
using System.Management.Automation;
using System.Management.Automation.Runspaces;

namespace UnitTestProject1
{
    [TestClass]
    public class UnitTest1
    {
        private string fileInputPath;
        private string fileOutputPath;

        private TestContext testContextInstance;
        /// <summary>
        ///Gets or sets the test context which provides
        ///information about and functionality for the current test run.
        ///</summary>
        public TestContext TestContext
        {
            get
            {
                return testContextInstance;
            }
            set
            {
                testContextInstance = value;
            }
        }
```

テストコード(共通部分) (2/2)

```
#region 追加のテスト属性
//
//テストを作成するときに、次の追加属性を使用することができます:
//
//クラスの最初のテストを実行する前にコードを実行するには、ClassInitialize を使用
//[ClassInitialize()]
//public static void MyClassInitialize(TestContext testContext)
//{
//}
//
//クラスのすべてのテストを実行した後にコードを実行するには、ClassCleanup を使用
//[ClassCleanup()]
//public static void MyClassCleanup()
//{
//}
//
//各テストを実行する前にコードを実行するには、TestInitialize を使用
[TestInitialize()]
public void MyTestInitialize()
{
    this.fileInputPath = TestContext.TestRunResultsDirectory;
    this.fileOutputPath = TestContext.DeploymentDirectory;
}
//
//各テストを実行した後にコードを実行するには、TestCleanup を使用
//[TestCleanup()]
//public void MyTestCleanup()
//{
//}
//
#endregion
```

bat (1/7)

- テスト対象のコード(sample.bat)

```
 1 @ECHO OFF
 2
 3 REM 引数の値チェック
 4 IF "%~1"=="" (
 5     GOTO NOT_ARGS_END
 6 ) ELSE (
 7     SET CHECKFILE=%1
 8 )
 9
10 REM ファイルの存在チェック
11 IF NOT EXIST %CHECKFILE% GOTO FILE_NOT_FOUND_END
12
13 REM ファイルが存在した場合の処理
14 EXIT /b 0
15
16 REM 引数無し場合の処理
17 :NOT_ARGS_END
18 EXIT /b 1
19
20 REM ファイルが存在しない場合の処理
21 :FILE_NOT_FOUND_END
22 EXIT /b 2
[EOF]
```

bat (2/7)

- テストコード①(C#)

```
[TestMethod]
public void samplebat_正常系_引数に指定したファイルが存在した時に戻り値が0であること()
{
    // Inフォルダに引数2に指定するファイルを作成
    // 例)
    // C:¥Users¥¥sano¥documents¥visual studio 2013¥Projects¥UnitTestProject1¥TestResults
    // ¥Deploy_sano 2015-01-05 00_54_37¥In¥SANO-PC
    var directoryName = Path.Combine(this.fileInputPath, "01");
    var fileName = Path.Combine(directoryName, "test.txt");
    Directory.CreateDirectory(directoryName);
    File.Create(fileName);

    var startInfo = new ProcessStartInfo();

    // 起動する実行ファイルのパス
    startInfo.FileName = System.Environment.GetEnvironmentVariable("ComSpec");
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin¥Debug
    var scriptPath = Directory.GetCurrentDirectory();
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin
    scriptPath = Path.GetDirectoryName(scriptPath);
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1
    scriptPath = Path.GetDirectoryName(scriptPath);
```

bat (3/7)

- テストコード①(C#)

```
    // 作業フォルダの設定
    startInfo.WorkingDirectory = Path.Combine(scriptPath, "script");

    // コマンドライン引数を指定
    startInfo.Arguments = string.Format(@"/c {0} {1}", "sample.bat" , @"""" + fileName + @"""");

    // プロセス用の新しいウィンドウを作成せずにプロセスを起動する場合は true、
    // それ以外の場合は false。 既定値は、false です。
    startInfo.CreateNoWindow = true;

    // プロセスを起動するときにシェルを使用する場合は true、
    // プロセスを実行可能ファイルから直接作成する場合は false。 既定値は、true です。
    startInfo.UseShellExecute = false;

    // 戻り値が「0」の場合、テスト成功
    using (var process = Process.Start(startInfo))
    {
        process.WaitForExit();
        var expectedCode = 0;
        Assert.AreEqual(expectedCode, process.ExitCode);
    }
}
```

bat (4/7)

- テストコード②(C#)

```
[TestMethod]
public void samplebat_正常系_引数無しで実行した時に戻り値が1であること()
{
    var startInfo = new ProcessStartInfo();

    // 起動する実行ファイルのパス
    startInfo.FileName = System.Environment.GetEnvironmentVariable("ComSpec");
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin¥Debug
    var scriptPath = Directory.GetCurrentDirectory();
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin
    scriptPath = Path.GetDirectoryName(scriptPath);
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1
    scriptPath = Path.GetDirectoryName(scriptPath);

    // 作業フォルダの設定
    startInfo.WorkingDirectory = Path.Combine(scriptPath, "script");
```

bat (5/7)

- テストコード②(C#)

```
    // コマンドライン引数を指定
    startInfo.Arguments = string.Format(@"/c {0}", "sample.bat");

    // プロセス用の新しいウィンドウを作成せずにプロセスを起動する場合は true、
    // それ以外の場合は false。 既定値は、false です。
    startInfo.CreateNoWindow = true;

    // プロセスを起動するときにシェルを使用する場合は true、
    // プロセスを実行可能ファイルから直接作成する場合は false。 既定値は、true です。
    startInfo.UseShellExecute = false;

    // 戻り値が「1」の場合、テスト成功
    using (var process = Process.Start(startInfo))
    {
        process.WaitForExit();
        var expectedCode = 1;
        Assert.AreEqual(expectedCode, process.ExitCode);
    }
}
```

bat (6/7)

- テストコード③(C#)

```
[TestMethod]
public void samplebat_正常系_引数で指定したファイルが存在しない時に戻り値が2であること()
{
    var startInfo = new ProcessStartInfo();

    // 起動する実行ファイルのパス
    startInfo.FileName = System.Environment.GetEnvironmentVariable("ComSpec");
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin¥Debug
    var scriptPath = Directory.GetCurrentDirectory();
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin
    scriptPath = Path.GetDirectoryName(scriptPath);
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1
    scriptPath = Path.GetDirectoryName(scriptPath);

    // 作業フォルダの設定
    startInfo.WorkingDirectory = Path.Combine(scriptPath, "script");
```

bat (7/7)

- テストコード③(C#)

```
        // コマンドライン引数を指定
        startInfo.Arguments = string.Format(@"/c {0} {1}", "sample.bat", "C:¥¥test.txt");

        // プロセス用の新しいウィンドウを作成せずにプロセスを起動する場合は true、
        // それ以外の場合は false。 既定値は、false です。
        startInfo.CreateNoWindow = true;

        // プロセスを起動するときにシェルを使用する場合は true、
        // プロセスを実行可能ファイルから直接作成する場合は false。 既定値は、true です。
        startInfo.UseShellExecute = false;

        // 戻り値が「2」の場合、テスト成功
        using (var process = Process.Start(startInfo))
        {
            process.WaitForExit();
            var expectedCode = 2;
            Assert.AreEqual(expectedCode, process.ExitCode);
        }
    }
```

VBScript (1/9)

- テスト対象のコード(sample.vbs)

```
Dim oParam
Set oParam = WScript.Arguments

'引数の数が0の場合、戻り値「1」で処理を終了
If oParam.Count = 0 Then
    Set oParam = Nothing    'オブジェクトの破棄
    Set oParam = Nothing    'オブジェクトの破棄
    WScript.Quit(1)
End If

'引数の数が1ではない場合、戻り値「2」で処理を終了
If oParam.Count <> 1 Then
    Set oParam = Nothing    'オブジェクトの破棄
    WScript.Quit(2)
End If

'引数に指定された値(ファイル)が存在しない場合、戻り値「3」で処理を終了
Set objFileSys = CreateObject("Scripting.FileSystemObject")
If objFileSys.FileExists(oParam(0)) = False Then
    Set oParam = Nothing        'オブジェクトの破棄
    Set objFileSys = Nothing    'オブジェクトの破棄
    WScript.Quit(3)
End If
Set objFileSys = Nothing        'オブジェクトの破棄

'正常終了
Set oParam = Nothing            'オブジェクトの破棄
WScript.Quit(0)
```

VBScript (2/9)

- テストコード①(C#)

```
[TestMethod]
public void samplevbs_正常系_引数に指定したファイルが存在した時に戻り値が0であること()
{
    // Inフォルダに引数に指定するファイルを作成
    // 例)
    // C:¥Users¥¥sano¥documents¥visual studio 2013¥Projects¥UnitTestProject1¥TestResults
    // ¥Deploy_sano 2015-01-05 00_54_37¥In¥SANO-PC
    var directoryName = Path.Combine(this.fileInputPath, "10");
    var fileName = Path.Combine(directoryName, "test.txt");
    Directory.CreateDirectory(directoryName);
    File.Create(fileName);

    var startInfo = new ProcessStartInfo();

    // 起動する実行ファイルのパス
    startInfo.FileName = @"cscript";
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin¥Debug
    var scriptPath = Directory.GetCurrentDirectory();
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin
    scriptPath = Path.GetDirectoryName(scriptPath);
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1
    scriptPath = Path.GetDirectoryName(scriptPath);

    // 作業フォルダの設定
    startInfo.WorkingDirectory = Path.Combine(scriptPath, "script");
```

VBScript (3/9)

- テストコード①(C#)

```
    // コマンドライン引数を指定
    startInfo.Arguments = string.Format("//B //Nologo {0} {1}", "sample.vbs", @"""" + fileName + @"""");

    // プロセスの起動時のウィンドウを、最大化、最小化、通常 (最大化も最小化もしていない状態)、
    // または非表示のどの状態にするかを示す列挙値のいずれか。 既定値は、Normal です。
    startInfo.WindowStyle = ProcessWindowStyle.Hidden;

    // 戻り値が「0」の場合、テスト成功
    using (var process = Process.Start(startInfo))
    {
        process.WaitForExit();
        var expectedCode = 0;
        Assert.AreEqual(expectedCode, process.ExitCode);
    }
}
```

VBScript (4/9)

- テストコード②(C#)

```
[TestMethod]
public void samplevbs_正常系_引数無しで実行した時に戻り値が1であること()
{
    var startInfo = new ProcessStartInfo();

    // 起動する実行ファイルのパス
    startInfo.FileName = @"cscript";
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin¥Debug
    var scriptPath = Directory.GetCurrentDirectory();
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin
    scriptPath = Path.GetDirectoryName(scriptPath);
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1
    scriptPath = Path.GetDirectoryName(scriptPath);

    // 作業フォルダの設定
    startInfo.WorkingDirectory = Path.Combine(scriptPath, "script");
```

VBScript (5/9)

- テストコード②(C#)

```
    // コマンドライン引数を指定
    startInfo.Arguments = string.Format("//B //Nologo {0}", "sample.vbs");

    // プロセスの起動時のウィンドウを、最大化、最小化、通常 (最大化も最小化もしていない状態)、
    // または非表示のどの状態にするかを示す列挙値のいずれか。 既定値は、Normal です。
    startInfo.WindowStyle = ProcessWindowStyle.Hidden;

    // 戻り値が「1」の場合、テスト成功
    using (var process = Process.Start(startInfo))
    {
        process.WaitForExit();
        var expectedCode = 1;
        Assert.AreEqual(expectedCode, process.ExitCode);
    }
}
```

VBScript (6/9)

- テストコード③(C#)

```
[TestMethod]
public void samplevbs_正常系_引数1つ以外で実行した時に戻り値が2であること()
{
    var startInfo = new ProcessStartInfo();

    // 起動する実行ファイルのパス
    startInfo.FileName = @"cscript";
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin¥Debug
    var scriptPath = Directory.GetCurrentDirectory();
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin
    scriptPath = Path.GetDirectoryName(scriptPath);
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1
    scriptPath = Path.GetDirectoryName(scriptPath);

    // 作業フォルダの設定
    startInfo.WorkingDirectory = Path.Combine(scriptPath, "script");
```

VBScript (7/9)

- テストコード③(C#)

```
    // コマンドライン引数を指定
    startInfo.Arguments = string.Format("//B //Nologo {0} {1}", "sample.vbs", "1 2");

    // プロセスの起動時のウィンドウを、最大化、最小化、通常 (最大化も最小化もしていない状態)、
    // または非表示のどの状態にするかを示す列挙値のいずれか。 既定値は、Normal です。
    startInfo.WindowStyle = ProcessWindowStyle.Hidden;

    // 戻り値が「2」の場合、テスト成功
    using (var process = Process.Start(startInfo))
    {
        process.WaitForExit();
        var expectedCode = 2;
        Assert.AreEqual(expectedCode, process.ExitCode);
    }
}
```

VBScript (8/9)

- テストコード④(C#)

```
[TestMethod]
public void samplevbs_正常系_引数に指定したファイルが存在しない時に戻り値が3であること()
{
    var startInfo = new ProcessStartInfo();

    // 起動する実行ファイルのパス
    startInfo.FileName = @"cscript";
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin¥Debug
    var scriptPath = Directory.GetCurrentDirectory();
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin
    scriptPath = Path.GetDirectoryName(scriptPath);
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1
    scriptPath = Path.GetDirectoryName(scriptPath);

    // 作業フォルダの設定
    startInfo.WorkingDirectory = Path.Combine(scriptPath, "script");
```

VBScript (9/9)

- テストコード④(C#)

```
    // コマンドライン引数を指定
    startInfo.Arguments = string.Format("//B //Nologo {0} {1}", "sample.vbs", "C:¥¥test.txt");

    // プロセスの起動時のウィンドウを、最大化、最小化、通常 (最大化も最小化もしていない状態)、
    // または非表示のどの状態にするかを示す列挙値のいずれか。 既定値は、Normal です。
    startInfo.WindowStyle = ProcessWindowStyle.Hidden;

    // 戻り値が「3」の場合、テスト成功
    using (var process = Process.Start(startInfo))
    {
        process.WaitForExit();
        var expectedCode = 3;
        Assert.AreEqual(expectedCode, process.ExitCode);
    }
}
```

PowerShell (1/9)

- テスト対象のコード(sample.ps1)

■PowerShell実行イメージ

PS C:¥Users¥sano¥Desktop> ./sample.ps1 -name "ss" -filePath "c:¥test¥test.txt"

```
Param([string] $name, [string] $filePath)

# 名前付き引数「name」の値の長さが0の場合、戻り値：1を指定して処理終了
if ($name.Trim().Length -eq 0)
{
    return 1
}
# 名前付き引数「filePath」の値の長さが0の場合、戻り値：2を指定して処理終了
if ($filePath.Trim().Length -eq 0)
{
    return 2
}
# 名前付き引数「filePath」(ファイル)が存在しない場合、戻り値：3を指定して処理終了
if (-not(Test-Path $filePath))
{
    return 3
}

# 戻り値：0を指定して処理終了
return 0
```

PowerShell (2/9)

- テストコード①(C#)

```
[TestMethod]
public void samplepsl_正常系_引数nameとfilePathの値が設定されていて引数filePathに指定したファイルが存在する時に戻り値が0であること()
{
    // Inフォルダに引数に指定するファイルを作成
    // 例)
    // C:¥Users¥¥sano¥documents¥visual studio 2013¥Projects¥UnitTestProject1¥TestResults
    // ¥Deploy_sano 2015-01-05 00_54_37¥In¥SANO-PC
    var directoryName = Path.Combine(this.fileInputPath, "20");
    var fileName = Path.Combine(directoryName, "test.txt");
    Directory.CreateDirectory(directoryName);
    File.Create(fileName);

    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin¥Debug
    var scriptPath = Directory.GetCurrentDirectory();
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin
    scriptPath = Path.GetDirectoryName(scriptPath);
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1
    scriptPath = Path.GetDirectoryName(scriptPath);
```

PowerShell (3/9)

- テストコード①(C#)

```
    using (var rs = RunspaceFactory.CreateRunspace())
    {
        rs.Open();
        using (var ps = PowerShell.Create())
        {
            var psCmd = new PSCommand();
            psCmd.AddScript(
                string.Format(@"& {0} {1}",
                    @""""" + Path.Combine(scriptPath, "script", "sample.ps1") + @""""",
                    "-name 'テスト太郎' -filePath " + @""""" + fileName + @""""""));
            ps.Commands = psCmd;
            ps.Runspace = rs;
            var psoList = ps.Invoke();

            // PowerShell内でエラーが発生した場合はテスト失敗
            var errors = ps.Streams.Error;
            if (errors.Count > 0)
            {
                Assert.Fail();
            }

            // 戻り値が「0」の場合はテスト成功
            var acual = "";
            foreach (var pso in psoList)
            {
                if (pso != null)
                {
                    acual = pso.ToString();
                }
            }
            var expectedCode = "0";
            Assert.AreEqual(expectedCode, acual);
        }
    }
}
```

PowerShell (4/9)

- テストコード②(C#)

```
[TestMethod]
public void samplePs1_正常系_引数nameの長さ0で実行した時に戻り値が1であること()
{
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin¥Debug
    var scriptPath = Directory.GetCurrentDirectory();
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin
    scriptPath = Path.GetDirectoryName(scriptPath);
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1
    scriptPath = Path.GetDirectoryName(scriptPath);
```

PowerShell (5/9)

- テストコード②(C#)

```
        using (var rs = RunspaceFactory.CreateRunspace())
        {
            rs.Open();
            using (var ps = PowerShell.Create())
            {
                var psCmd = new PSCommand();
                psCmd.AddScript(
                    string.Format(@"& {0} {1}",
                        @""""" + Path.Combine(scriptPath, "script", "sample.ps1") + @""""",
                        "-name '' -filePath ''"));
                ps.Commands = psCmd;
                ps.Runspace = rs;
                var psoList = ps.Invoke();

                // PowerShell内でエラーが発生した場合はテスト失敗
                var errors = ps.Streams.Error;
                if (errors.Count > 0)
                {
                    Assert.Fail();
                }

                // 戻り値が「1」の場合はテスト成功
                var acual = "";
                foreach (var pso in psoList)
                {
                    if (pso != null)
                    {
                        acual = pso.ToString();
                    }
                }
                var expectedCode = "1";
                Assert.AreEqual(expectedCode, acual);
            }
        }
    }
```

PowerShell (6/9)

- テストコード③(C#)

```
[TestMethod]
public void samplepsi_正常系_引数nameの値が設定されていて引数filePathの長さ0で実行した時に戻り値が2であること()
{
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin¥Debug
    var scriptPath = Directory.GetCurrentDirectory();
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin
    scriptPath = Path.GetDirectoryName(scriptPath);
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1
    scriptPath = Path.GetDirectoryName(scriptPath);
```

PowerShell (7/9)

- テストコード③(C#)

```
        using (var rs = RunspaceFactory.CreateRunspace())
        {
            rs.Open();
            using (var ps = PowerShell.Create())
            {
                var psCmd = new PSCommand();
                psCmd.AddScript(
                    string.Format(@"& {0} {1}",
                        @""""""" + Path.Combine(scriptPath, "script", "sample.ps1") + @""""""",
                        "-name 'テスト太郎' -filePath ''"));
                ps.Commands = psCmd;
                ps.Runspace = rs;
                var psoList = ps.Invoke();

                // PowerShell内でエラーが発生した場合はテスト失敗
                var errors = ps.Streams.Error;
                if (errors.Count > 0)
                {
                    Assert.Fail();
                }

                // 戻り値が「2」の場合はテスト成功
                var acual = "";
                foreach (var pso in psoList)
                {
                    if (pso != null)
                    {
                        acual = pso.ToString();
                    }
                }
                var expectedCode = "2";
                Assert.AreEqual(expectedCode, acual);
            }
        }
    }
```

PowerShell (8/9)

- テストコード④(C#)

```
[TestMethod]
public void samplepsl_正常系_引数nameの値が設定されていて引数filePathに指定したファイルが存在しない時に戻り値が3であること()
{
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin¥Debug
    var scriptPath = Directory.GetCurrentDirectory();
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1¥bin
    scriptPath = Path.GetDirectoryName(scriptPath);
    // 例)
    // C:¥Users¥sano¥Documents¥Visual Studio 2013¥Projects¥UnitTestProject1¥UnitTestProject1
    scriptPath = Path.GetDirectoryName(scriptPath);
```

PowerShell (9/9)

- テストコード④(C#)

```
        using (var rs = RunspaceFactory.CreateRunspace())
        {
            rs.Open();
            using (var ps = PowerShell.Create())
            {
                var psCmd = new PSCommand();
                psCmd.AddScript(
                    string.Format(@"& {0} {1}",
                        @""""" + Path.Combine(scriptPath, "script", "sample.ps1") + @""""",
                        "-name 'テスト太郎' -filePath 'C:¥¥test.txt'"));
                ps.Commands = psCmd;
                ps.Runspace = rs;
                var psoList = ps.Invoke();

                // PowerShell内でエラーが発生した場合はテスト失敗
                var errors = ps.Streams.Error;
                if (errors.Count > 0)
                {
                    Assert.Fail();
                }

                // 戻り値が「3」の場合はテスト成功
                var acual = "";
                foreach (var pso in psoList)
                {
                    if (pso != null)
                    {
                        acual = pso.ToString();
                    }
                }
                var expectedCode = "3";
                Assert.AreEqual(expectedCode, acual);
            }
        }
    }
```

お勧めの本

- **Windows コマンドプロンプト**
  - Windows コマンドプロンプト ポケットリファレンス(3,002円。円技術評論社。2011/12/9)
- **VBScript / WSH**
  - [改訂版] VBScriptポケットリファレンス(2,030円。技術評論社。2006/5/30)
  - WSHクイックリファレンス 第2版(4,104円。オライリージャパン。2006/10/24)
- **PowerShell**
  - 【改訂新版】 Windows PowerShell ポケットリファレンス(3,002円。技術評論社。2013/2/23)
- **C#の文法**
  - C#ポケットリファレンス(2,786円。技術評論社。2011/12/3)
- **VB.NET / C#の入門書**
  - C#の絵本(1,922円。翔泳社。2008/2/5)
  - Visual Basicの絵本(1,922円。翔泳社。2011/4/9)

参考URL(1/3)

PowerShellのセキュリティポリシーを変更してスクリプトファイルを実行できるようにする

http://qiita.com/kmr_hryk/items/6d3a63d84fd7feca2826

DOS/VBScript

http://rururu.sakura.ne.jp/doc/DOS_VBScript.pdf

VBScriptについて

http://rururu.sakura.ne.jp/doc/VBScript%E3%81%AB%E3%81%A4%E3%81%84%E3%81%A6.pdf

Windowsコマンドプロンプト基礎文法最速マスター

http://windows.g.hatena.ne.jp/cx20/20100203/p1

VBScript 基礎文法最速マスター

http://vbscript.g.hatena.ne.jp/cx20/20100131/1264906231

PowerShell基礎文法最速マスター

http://winscript.jp/powershell/202

C#基礎文法最速マスター

http://anond.hatelabo.jp/20120813121640

参考URL(2/3)

C# からbatファイルを呼ぶにはSystem.Diagnostics.Processを使う

http://c4se.hatenablog.com/entry/2012/07/28/192511

C＃アプリケーションでVBScriptファイルを呼び出す方法？

http://www.freeshow.net.cn/ja/questions/d3d5bdb7cbb6a00b413ad71f948dd066197f476cc19faf9c01f0b5294f8ab389/

C# Process

http://www.dotnetperls.com/process

PowerShell C#でInvoke-commandのリモート処理の戻り値を取得する方法

https://social.technet.microsoft.com/Forums/ja-JP/e9084418-626b-4b94-aeed-b9ab7686a321/powershell-cinvokecommand?forum=powershellja

Microsoft TechNet Windows PowerShell

https://social.technet.microsoft.com/Forums/ja-JP/home?forum=powershellja

Powershell retrieving pipeline errors in C#

https://social.msdn.microsoft.com/Forums/exchange/en-US/b2bece71-72d7-4305-ad81-02139959e643/powershell-retrieving-pipeline-errors-in-c

参考URL(3/3)

【Windows PowerShell】スクリプトの途中でスクリプトを強制終了する

http://munibus.hatenablog.com/entry/2014/01/22/053159

PowerShell/Windows7にPowerShell4.0をインストールする手順

http://win.just4fun.biz/PowerShell/Windows7%E3%81%ABPowerShell4.0%E3%82%92%E3%82%A4%E3%83%B3%E3%82%B9%E3%83%88%E3%83%BC%E3%83%AB%E3%81%99%E3%82%8B%E6%89%8B%E9%A0%86.html

コマンドレットの作成方法

http://csharper.blog57.fc2.com/blog-entry-55.html

PowerShell を C# から実行する

http://tech.tanaka733.net/entry/2013/12/10/powershell-from-csharp

【C#】C# から PowerShell を使うには

http://blogs.yahoo.co.jp/dk521123/archive/2013/11/25

Windows、バッチファイルの引数から、引数を囲んでいるダブルクォーテーションを除去する

hhttp://piyopiyocs.blog115.fc2.com/blog-entry-801.html