bject BLVD オブジェクトの散歩道

例題でわかる！ .NET Framework

第2回 「ファイルディレクトリ」

吉田 弘一郎 *YOSHIDA, Koichiro*

はじめに

今回も、非常に基本的なところから始めます。でも、基本的であると言うことと初歩的であると言うのは少々別の話で、話題としては上級向けであると思います。

お題は、ファイルシステム内のディレクトリの散歩の仕方です。地味な話題のようですが、意外と盲点になっている分野に思えます。

本稿で前提となるもの

OS	Windows 2000 Professional（SP3）以降
開発環境	Visual Studio.NET .NET Framework 1.0.3705.288（SP2） Internet Explorer 6.0.2800.1106

初級 ― 中級 ― 上級

この記事で解説したサンプルプログラムは、付録CD-ROMの¥DMAG¥SAMPOフォルダ以下に収録しています

SAMPO02.SLN：今回紹介したサンプルのソリューションファイル

例題1：Environmentクラス

最初に、Visual Studio.NETでC#のコンソールアプリケーションプロジェクトを新規作成して、まずは自動生成された骨格プログラムを見てみましょう。

「[STAThread]」は気にしないことにして、Main関数のパラメータに注目してください。これはプログラムのパラメータである「コマンドパラメータ」を保持するストリング配列argsです。

```
using System;
namespace CS01
{
  class Class1
  {
    [STAThread]
    static void Main(string[] args)
    {
    }
  }
}
```

たとえば、次のようにこのプログラムを起動した場合、“ABC”と“123”がコマンドパラメータになります。

```
C:¥>CS01 ABC 123
```

コンソールプログラムの場合には、このコマンドパラメータを用いることが多いですね。C#の場合は、コードがVisual C/C++と非常によく似ています。パラメータの個数とパラメータそのものの読み出し方法は次のとおり。

static void Main(string[] args)の
パラメータの個数はargs.Length、
n番目のパラメータはargs[n]。

これを用いれば、たとえばリスト1の

リスト1：例題1、コマンドパラメータを読み込む（サンプルCS01）

```
' コンソールプログラムでコマンドラインパラメータを読み込んで
' 用いる例題。たとえば
' C:¥>CS01 吉田
' と打てば、「吉田」がコマンドラインパラメータとなる

using System;

namespace CS01
{
    /// <summary>
    /// Class1の概要の説明
    /// </summary>
    class Class1
    {
        /// <summary>
        /// アプリケーションのメインエントリポイント
        /// </summary>
        [STAThread]
        static void Main(string[] args)
        {
            Console.WriteLine("  引数の個数：{0}",args.Length);
            int i;
            for(i=0;i<args.Length;i++)
                Console.WriteLine("   arg[{0}]={1}",i, args[i]);
        }
    }
}
```

図1：リスト1実行例1

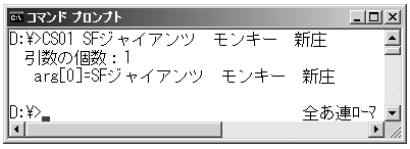
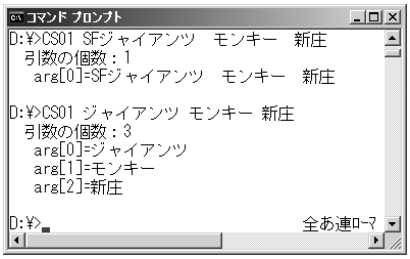


図2：リスト1実行例2



ように、コマンドパラメータをプログラム内に取り込むことができます。

米メジャーリーグ、ワールドシリーズに新庄が出ていたので、次のようなパラメータにしてみましょう。

```
D:¥>CS01 SFジャイアンツ　モンキー　新庄
```

妙だ！　プログラムに間違いはないはずなのに、まったく機能していません（図1）。実は、パラメータを“全角の空白”で区切ってもだめなのです。図1では、“SFジャイアンツ”と“モンキー”、“モンキー”と“新庄”の間にそれぞれ全角の空白を入れて実行しています。これを半角の空白にして実行すると、図2のようにきちんと動作してくれます。一瞬、あせりました。

サンプルCS01をVisual Basic.NETで書き換える

さて、本誌のメジャー言語は、まだまだVisual Basic。そこで、リスト1と同様のことをVisual Basic.NETで行なってみましょう。Visual Studio.NETが自動生成してくれるVisual Basicのコンソールアプリケーションの骨格プログラムは次のとおりで、コマンドパラメータはどこにもありません。

```
Module Module1

  Sub Main()

  End Sub

End Module
```

そこで、ヘルプであれこれ探してみると、「Environment」というクラスを用いればよいことがわかりました。図3のように、Environmentクラスには GetCommandLineArgsというメソッドがあります。これを用いて前掲のサンプルCS01をVisual Basicで書き直したものがサンプルVB01（リスト2）です。

C#の場合と異なるのは、コマンドそのものまで含めてパラメータとしている点です。

実行結果は、図4のようになり、パラメータ数のカウント以外はC#の場合とまったく同一です。このEnvironmentクラスは、コマンド行のみならず、リスト3のように、さまざまな環境情報を提供くれるので非常に便利です（図5）。こういうクラスが準備されていると言うのも、.NETの特徴のひとつではないでしょうか。レジストリを意識することなく、システム情報などを非常に簡単に読み出せるのがありがたいです。

図3：EnvironmentクラスのGetCommandLineArgsメソッド