bject BLVD オブジェクトの散歩道

例題でわかる！ .NET Framework

第3回 「テキストファイルの読み書き」

吉田 弘一郎 *YOSHIDA, Koichiro*

はじめに

ストリングの基本を理解し、ディレクトリの情報を読めるようになったら、次はファイルそのものへのアクセスです。ファイルへデータを出力したり、ファイルからデータを読み込むのです。このようなファイルとのデータのやり取りは、プログラムのもっとも基本的な仕事のひとつですね。ファイルという用語を拡大解釈してゆけば、それはデータベースにもなれば、ネットワークにもなるのです。

ファイルへの読み書きには何通りものパターンがあります。その多くは、データをテキストにしないで、直接読み書きするものです。でも、今月は、テキストファイルの読み書きに限定します。みなさんが.NETで開発したプログラムで、テキストファイルに収まっている既存のデータにアクセスする必要が生じたという状況設定です。

数値をテキストの形でファイルに格納するなんて、効率が悪い時代遅れの方法であるという見方もあるでしょう。ここでは、「なぜかテキストファイルに収まっているデータがあるから」と逃げることもできます。

しかし、プログラムを開発する場合には、直接中身を見て理解できるテキストファイルは便利ですし、ここまでハードディスクが廉価、大容量かつ高速になれば、別に気にする必要もないように思えます。それどころか、XMLという「データの格納効率／転送効率」などを最初から無視する技術がもてはやされているではありませんか。とにかく、テキストファイルの読み書きは、.NETの世界においても、非常に基本的かつ重要なものなのです。

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。

SAMPO03.SLN：今回紹介したサンプルのソリューションファイル

テキストファイルへの読み書きの方法

さて、ファイルへのテキスト出力は、とくに問題になる事項はありません。もちろん、.NETは高度にオブジェクト指向した世界ですから、数値データの書式に関しては貧弱な機能しかサポートしてくれません。ですから、この連載では、始めから迷わず「C言語のprintfを用いる」ことにしています。この段階で「Managed」か否かは問題で

はありません。とにかく、割り切って「printf」を用いることにより、書式付きWriteは問題なくできます。これは、出力先がファイルであろうが、ストリングであろうが同じです。

では、書式付きで整然とファイルに書き出されたデータをC#やVisual Basic.NETで読み込むにはどうすればよいのでしょう。

私が調べた限り、何もサポートされていないのです。ヘルプにもありません。Googleで探しても見つかりません。「テキストファイル」から文字単位、ないしは行単位で読み込むことはできても、「1行から“x,y,z”の3つの数値を読み込む」ような機能はないのです。

「ひとつの数値がASCIIコードとして収まっている文字列」からその数値を取り出すことは簡単にできます。しかし、たとえば次のような行、

```
4  4.00000  6.9756E-002
```

から、3つの数値を抽出する術はないのです。それには「正規表現をサポートするRegExクラス」を用いればよいという記事もありました。しかし、この程度の作業に仰々しく正規表現云々というのも大人気ない気がします。そして数日間あれこれ迷ったあげく、簡単なクラスを作ることにしました。簡単な入力行パーサーですが、これが今月拡張した自作ライブラリKYLib（KyLib.cs）の機能です。

この入力パーサーでは、Consoleへの出力にWriteメソッドおよびWriteLineメソッドを用いました。ここで、Consoleの代わりに、StreamWriterクラスを用いれば、テキストファイルへ出力できるのです。つまり、StreamWriterオブジェクトを得るだけの話です。これには、次のように何通りもの方法があります（これらの方法はすべてC#で紹介します）。

テキスト出力ファイルの作成法①

ファイル名を与えて直接StreamWriterオブジェクトを作ることができます。次の例をご覧ください。

```
StreamWriter sw=new
    StreamWriter("ABC.txt");
    sw.WriteLine("今日は");
    sw.Close();
```

とにかく簡単なので、私はもっぱらこの方法を用いています。

テキスト出力ファイルの作成法②

先月習ったFileInfoクラス経由でStreamWriterオブジェクトを作ることができます。FileInfo型オブジェクト“f”が与えられれば、そのメソッドCreateTextを用いて、次のようにするのです。

```
StreamWriter sw=f.CreateText();
```

また、FileInfo型オブジェクトfを与えられたファイル名から生成し、それを用いることもできます。

```
FileInfo f=new FileInfo("ABC.txt");
StreamWriter sw=f.CreateText();
```

単に手間がかかるだけのように見えますが、FileInfo型オブジェクトを用いてそのファイルのさまざまな情報を得る必要がある場合もあるのです。

テキスト出力ファイルの作成法③

FileStreamクラス経由でStreamWriterオブジェクトを作ることもできます。

```
FileStream fs=new
    FileStream("ABC.txt",FileMode.Create);
    StreamWriter sw=new StreamWriter(fs);
```

C言語で、fopenしてFILE型のポインタを得るのに似ています。これが最も一般的なのではないかと思います。

例題１：テキストファイルへの出力

それではひとつ目の例題にかかります。

C#の例題CS01

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、次のようなデータをもつファイル（Data01.txt）を生成するプログラムを作成しましょう。

```
 0   0.00000  0.0000E+000
 1   1.00000  1.7452E-002
 2   2.00000  3.4899E-002
 3   3.00000  5.2336E-002
 4   4.00000  6.9756E-002
 5   5.00000  8.7156E-002
 6   6.00000  1.0453E-001
 7   7.00000  1.2187E-001
 8   8.00000  1.3917E-001
 9   9.00000  1.5643E-001
10  10.00000  1.7365E-001
```