Javaユーザーのための C#入門

乗り換えをもくろむあなたに

第2回 コンソールアプリケーションを使った基本的なプログラミング

矢嶋 聡 *YAJIMA, Satoshi*
NRIラーニングネットワーク株式会社
MSDN Regional Director

はじめに

前回は、C#を使った.NET向けアプリケーションの特徴や、それらに関連した基礎的な用語について解説しました。また、Visual Studio.NETを使って簡単なWindowsアプリケーションの作成を行ない、Visual Studio.NETを使うことで、効率的な開発作業ができることを体感していただけたことでしょう。

2回目の今回は、コマンドプロンプト上で動作するプログラムである、コンソールアプリケーションの作成を通して、C#の構文や構造に関する特徴、また、Javaにはない大きな特徴のひとつである属性宣言について説明しようと思います。前回も触れたように、C#の文法そのものの解説ではなく、

・何が作れるのか
・どう作るのか

に重点を置いているので、Visual Studio .NETを使いながら、どんなアプリケーションができるのか、どう作るのかを見てゆきます。誌面の都合もあり、C#の文法の詳細など、C#のすべてが取り上げられるわけではありませんが、今後C#を修得する上での予備知識となれば幸いです。

本稿で前提となるもの

OS	Windows 2000 Professional（SP3）以降
開発環境	Visual Studio.NET Visual C# .NET Compiler 7.00.9466 .NET Framework 1.0 Ver 1.0.3705（SP2） Internet Explorer 6.0（SP2）

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¥CSHARPフォルダ以下に収録しています

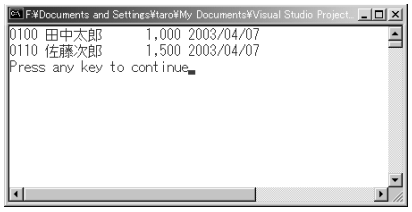
¥MYAPP：今回説明したサンプル

今回のサンプル

今回作成するサンプルプログラムは、コマンドプロンプトで実行するテキストベースのアプリケーションである「コンソールアプリケーション」です。このサンプルプログラムは、メモリ上に顧客を表わすオブジェクトを2つ作り、その内容を画面に表示し、XMLファイルとして出力する簡単な処理を行なうものですが、C#の基本的な内容やJavaとは異なる特徴が盛り込まれています。

今回のサンプルでは、顧客による購買をポイントとして管理するために、ひとりの顧客をひとつのオブジェクトとして表現するので、顧客を表わすクラス“Customer”を作ります。このプ

図1：コマンドプロンプトに顧客情報を表示



リスト1：出力されたXMLファイル

```
<?xml version="1.0"?>
<Customer xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <ID>100</ID>
  <Total>1000</Total>
  <EndValid>2003-03-24</EndValid>
</Customer>
```

リスト2：今回のサンプルでメインとなるプログラム（Class1.cs）

```
using System;
using System.Xml.Serialization;
using System.IO;

namespace MyApp
{
    class Class1
    {
        static void Main(string[] args)
        {
            // 顧客オブジェクトインスタンス用の変数
            Customer ob1, ob2;                                          ①
            // インスタンスを作成
            ob1 = new Customer(100, "田中太郎", 0);                      ②
            ob2 = new Customer(110, "佐藤次郎", 0);
            // ポイントの追加
            ob1.AddPoint(1000);                                         ③
            ob2.AddPoint(1500);

            // 簡単なデータの表示
            Console.WriteLine(                                          ④
                "{0:0000} {1,-8} {2,6:#,##0} {3:yyyy/MM/dd}",
                ob1.ID, ob1.Name, ob1.Point, ob1.EndValid );
            Console.WriteLine(
                "{0:0000} {1,-8} {2,6:#,##0} {3:yyyy/MM/dd}",
                ob2.ID, ob2.Name, ob2.Point, ob2.EndValid );

            // XMLデータの出力
            XmlSerializer xl = new XmlSerializer( typeof(Customer) );   ⑤
            FileStream fs1 = new FileStream("Test1.xml",
                                FileMode.Create );
            xl.Serialize(fs1, ob1);
            fs1.Close();
        }
    }
}
```

ログラムを実行すると、2つの顧客オブジェクトを作成し、個々の顧客に適当なポイントを加算して、その結果をコマンドプロンプトに出力します（図1）。また、ひとりの顧客の情報をXMLファイルとして出力する処理を行ないます（リスト1）。

今回のソースコードはリスト2とリスト3の2つのソースファイルから構成されています。それぞれのファイルには、

class Class1

の記述で始まるクラスと、

class Customer

の記述で始まるクラスがあり、合計2つのクラスから構成されます。前回触れたように、C#もオブジェクト指向言語であるので、このようなクラスのブロックからプログラムが形成されます。コードの個々の意味については、この後、Visual Studio.NETで作成しながら確認してみましょう[注1]。

コンソールアプリケーションプロジェクトを作る

前回説明したように、Visual Studio .NETでプログラムを開発するには、まず最初に適切なプロジェクトを選択して作成することから始めます。

ではVisual Studio.NETを起動して、メニューから［ファイル］-［新規作成］-［プロジェクト］を選んでください。「新しいプロジェクト」ダイアログボックスが表示されます（図2）。

ここで、次のように設定してください（他の部分は既定のまま）。今回は、コンソールアプリケーションを選んでいる点に注意してください。

注1）もしVisual Studio.NETが手元になければ、.NET Framework SDKでも作成できます。この場合、サンプルのファイルをメモ帳などで作成して、以下のようにコマンドラインから実行してください。これで、MyApp.exeが作成されます。

> csc /out:MyApp.exe Class1.cs Customer.cs