はじめに

.NET Framework 2.0の開発にあたっては、入門／学習向けとしてVisual Studio Express Editionというツール群が提供されています。これらのツールは無料であっても、デザイン画面でのドラッグ＆ドロップによるコントロールの配置やウィザードによる開発、コード記述時のインテリセンスサポートといった機能を備えており、強力で高機能なツールであるということができます。ただ、残念なことに上位の他のVisual Studioと比較すると機能的に一部制限されている部分があります。Windowsアプリケーションを開発するうえではデータベースとの接続の制限が最も残念な点だと言えるでしょう。

Express Editionの中で、C#を利用してWindowsアプリケーションを開発できるものがVisual C# 2005 Express Edition（以下C# Express）です。C# Expressでもデータベースを利用するアプリケーションは開発できるのですが、C# Expressのウィザードから接続できるデータベースはローカルマシンに置かれたMDFファイルに限定されています。また、このMDFファイルへの接続には、SQL Serverの中でもSQL Server 2005 Express Edition（以下SQL Express）のみが持つユーザーインスタンスという機能が利用されています。このため、一見するとC# Expressでは、たとえばSQL Server 2000を利用するクライアント／サーバー（C/S）型のシステムは構築できないかのように思われてしまいます。

しかし、実はちょっとした工夫でC# ExpressでC/S型のシステムを開発することが可能になります。この記事では、まずローカルのMDFファイルを利用するWindowsアプリケーションを構築し、そのアプリケーションをSQL Server 2000を利用するC/Sシステムに拡張します。

また、このアプリケーションの開発の中で、データベースとの接続を行なう部分を独立したクラスライブラリとして作成し、そのライブラリの機能拡張をテストツールであるNUnitを利用しながら行なう方法を説明します。

レベル >>> Level

1 2 3 4 5

サンプル >>> Sample

この記事で取り上げたソースコードおよびサンプルプログラムは、
http://www.shoeisha.com/mag/windev/
からダウンロード可能です。

言語 >>> Language

- C#

ツール >>> Tool

- Visual C# 2005 Express
- NUnit 2.2.8
- SQL Server 2005 Express
- SQL Server 2000

事前の準備

ツールのインストール

この記事では以下のツールを利用しています。これらのツールのインストールは、ダウンロードしたファイルを実行し、ウィザードに従って設定を行なうことで簡単に終了します。

・C# Express
（http://www.microsoft.com/japan/msdn/vstudio/express/vcsharp/）
C# Expressのインストール時に同時にSQL Expressもインストールされます。ただし、SQL ExpressにはSP1が提供されていますので、Windows UpdateなどからSQL ExpressのSP1の適用を行なうようにしてください。

・NUnit
（http://www.nunit.org/）
NUnitは簡単なプログラムの記述でクラスの動作をテストすることができるツールです。執筆時点での最新バージョンは2.2.8となっています。.NET Framework 1.1用と2.0用があるので、2.0用（NUnit-2.2.8-net-2.0.msi）をダウンロードし、インストールします。

NUnitは主にテスト駆動と呼ばれる開発方法論と共に広く利用されています。テスト駆動開発では、まずクラスが実装するべき動作を検証するためのプログラムをテストとして記述します。テストを記述した後で実際のクラスの動作をプログラミングしていきます。まずテストの記述から開発作業が進められるため「テスト駆動」開発と呼ばれています。この記事の中でもテスト駆動開発を少しだけ取り入れています。

利用するデータベースの作成

作成するアプリケーションは、最終的にはSQL Server 2000上のデータベー

スを利用することとします。最初にこのデータベースと利用するテーブルを作成しておきましょう。

SQL Server Enterprise Managerを利用して「sampleDB」という名前のデータベースを作成します。その中に「商品番号」「商品名」「在庫数」を管理する「zaikoTable」を作成します（表1・図1）。

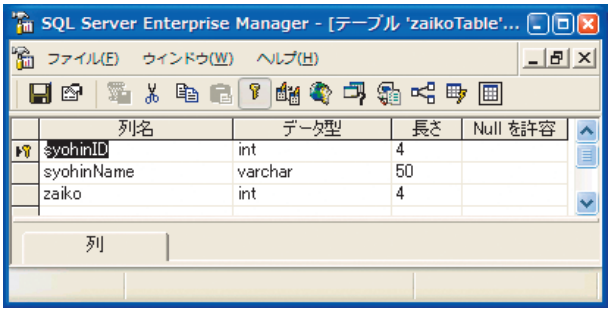
前述したとおり、C# Expressでは直接このsampleDBを利用することはできません。このため、sampleDBの実体であるMDFファイルをSQL Server 2000から取り出し、C# Expressで利用することとします。

MDFファイルを取り出すには、まずSQL Server 2000のサービスを停止します。通常「C:¥Program Files¥Microsoft SQL Server¥MSSQL¥Data」といったフォルダに「sampleDB_Data.MDF」という名前のMDFファイルができているはずです。これがsampleDBの実体になります。このsampleDB_Data.MDFファイルをどこか適当なフォルダにコピーしておきます。

表1：zaikoTable

項目名	データ型	サイズ	
syohinID	int	4	主キー
syohinName	varchar	50	
zaiko	int	4	

図1：zaikoTable



サンプルアプリケーションの構造

作成するサンプルアプリケーションは、ひとつのソリューションの中に3つのプロジェクトを含みます（図2）。

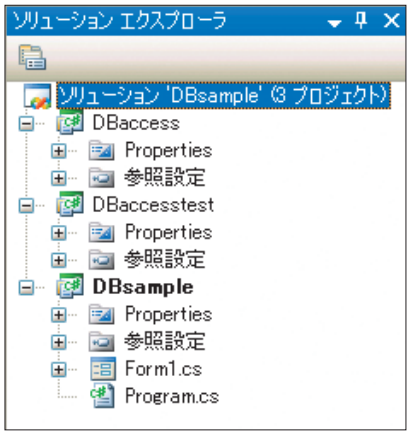
- DBにアクセスするクラスライブラリ（DBaccess）
- テストを実行するクラスライブラリ（DBaccesstest）
- Windowsアプリケーション（DBsample）

まず最初にWindowsアプリケーションであるDBsampleプロジェクトを作成し、そこに2つのクラスライブラリを追加します。

C# Expressを起動し、メニューから「ファイル」-「新しいプロジェクト」を選択します。ここでWindowsアプリケーションテンプレートを選択し、プロジェクト名を「DBsample」として作成します。

作成したソリューションに新しいプロジェクトを追加するには、ソリューションエクスプローラの「ソリューション 'DBsample'」を右クリックし、

図2：ソリューションの状態



[追加]-[新しいプロジェクト]を選択します。現在のプロジェクトを保存するよう求められますが、ここでは適当な場所を指定して保存しておきます。そして、新しいプロジェクトとしてクラスライブラリテンプレートを選択し、プロジェクト名を「DBaccess」として[OK]ボタンをクリックします。

同じようにして「DBaccesstest」という名前でもうひとつのクラスライブラリを追加します。

クラスライブラリとして追加したDBaccessプロジェクトとDBaccesstestプロジェクトには自動的にClass1.csというファイルが作成されますが、このファイルは使いませんのでそれぞれ削除しておきましょう。

sampleDBデータベースへのアクセス権について

この記事の中では説明を簡便にするために、sampleDBデータベースへの接続にユーザー名「sa」、パスワード「sapassword」を使用することとします。実際の運用ではセキュリティを十分に考慮して、適切なデータベース接続を利用するようにしてください。