C# .NETのメリットを採る
.NET Frameworkを知るには最適な言語

初音 玲
HATSUNE, Akira

Technology Tools

- ❑ Visual Basic .NET
- ☑ Visual C# .NET
- ❑ SQL Server 2000
- ❑ Oracle 9i
- ❑ Access 2002
- ❑ ASP.NET
- ❑ Internet Information Services
- ❑ Other:

Level

★ ★ ☆ ☆ ☆

Samples

・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、付録CD-ROMの¥DOTNET¥FEATURE01_03ディレクトリに収録しています。

¥CSCTL
コントロールを動的に配置するサンプル

¥CSFORM
継承を使用してフォームを生成するサンプル

¥CSBROWSER2
ActiveXコントロールを使用するサンプル

¥CSBROWSER3
ActiveXコントロールを使用するサンプル

はじめに

Microsoft Visual Studio.NET（以下VS.NET）に搭載されている言語の中でも、Visual C#.NET（以下C#）の情報量が、書籍やネットワーク上で他の言語のそれと同等か凌駕しているのは、なぜだろうか。

さまざまな理由が考えられるが、以下の2点が大きいのではないだろうか。

・.NET Frameworkのネイティブ言語
・厳密な文法で教育向けとしての側面もある

実際に使用して他の言語と比較すると、この2点によって.NET Frameworkクラスライブラリを「正しく使う」という面ではアドヴァンテージがあるように思える。

たしかに、VB.NETやVC++.NETでも.NET Frameworkのクラスライブラリを使いこなすことはできるが、.NET Frameworkクラスライブラリの利点を活かしたコード以外にも、過去との互換性や言語仕様のしがらみから、別の書き方も出来てしまうという点で、誰もが同程度のよい品質のコードを書けるかどうかは怪しい。

C# .NETならば、慣れないうちは、まいってしまうくらい文法や変数型などもかなり厳密にチェックされるため、.NET Frameworkを勉強するにはよい言語だろう。この雰囲気は、Pascal言語の雰囲気すら漂わせている。学習用や熱狂的な一部開発者が使い続ける言語になるか、はたまた.NET言語の本流になるか、まだわからないが、VB.NETの開発者はぜひ.NET Frameworkクラスライブラリの使い方を確認する手段としてC# .NETを利用してみることをお勧めする。

どうしてこのようなアプローチを勧めるかは、VB.NET編で取り上げた3つのサンプルをC#.NETで記述するときに、.NET Frameworkクラスライブラリをどのように使っているかに注目してもらえればわかってもらえると思う。

図1：ボタンコントロールの動的生成サンプル

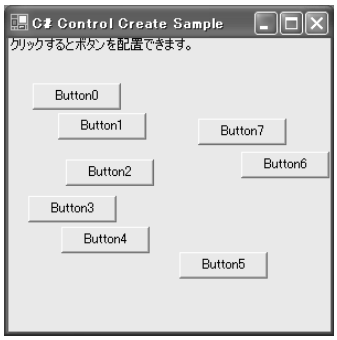
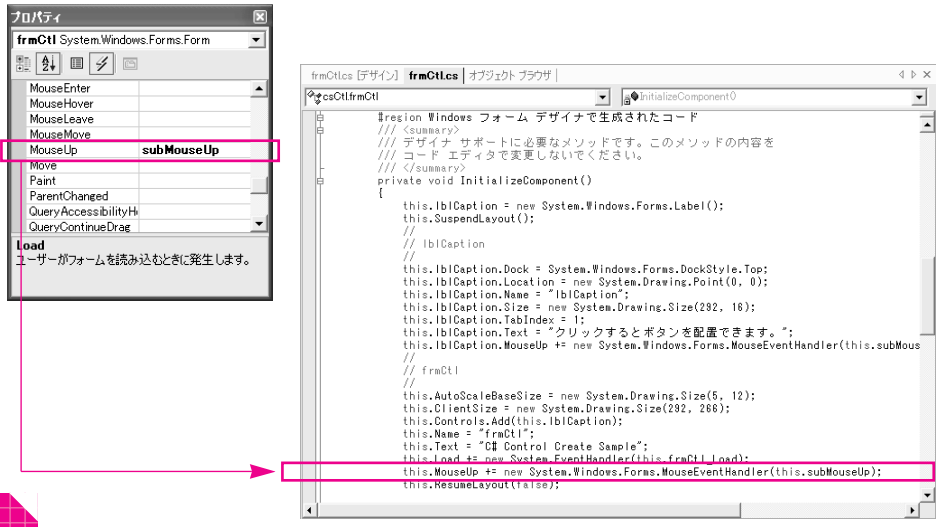


図2：イベントプロシージャの定義



動的にコントロールを配置する

最初のサンプルは、実行時に動的にコントロールを生成して配置するサンプルだ（図1）。このサンプルでは、VBやVB.NETで実現していたRedim文を使った動的配列をどのように.NET Frameworkクラスライブラリを使って実現するのか、また、イベントプロシージャはどのように定義するのかについて注目して欲しい。

イベントプロシージャの定義

VB.NETの場合、フォームやコントロールの標準イベントプロシージャは、たとえば、フォームであれば、フォームをダブルクリックしてForm_Loadイベントプロシージャを表示してから、コードウィンドウの上部右側の［プロシージャ］コンボボックスをドロップダウンすれば、フォームに定義されている標準的なイベントプロシージャの一覧が表示される。

C#では、［プロシージャ］コンボボックスに表示されるのは、現在、定義されているプロシージャのみであって、たとえフォームのイベントプロシージャにカーソルがあっても、フォームに定義されているイベントプロシージャの一覧は表示されない。その代わり、プロパティウィンドウにイベントボタンがあり、それをクリックすることで、プロパティウィンドウにイベントの一覧が表示され、そこに任意の名前でイベントプロシージャ名を記述できる方式を採用している。プロパティウィンドウで記述したイベントプロシージャの割り当て定義は、コードウィンドウの初期化ルーチンに即座に反映されるようになっている（図2）。

この2つの機能の差は、単純に操作性の違いのように捉えられるが、実は、この違いが、.NET Framework上でのよりよいイベントを選択することに一役かっている。具体的にはサンプル2の説明時に行なうとして、サンプル1（CSCTLサンプル）では動的配列について注目してゆこう。

それでは早速、サンプルのコードを見てみよう（リスト1）。

管理用の変数の定義

動的に生成したボタンコントロールを管理するために、mactlBtnという変数名で配列を宣言する。

```
private System.Windows.Forms.Button[]
  mactlBtn;
private int mintBtnCnt;
```

配列の要素数は未定で、変数mintBtnCntを使って管理し、動的に要素を拡張してゆけるようにしよう。配列mactlBtnと変数mintBtnCntは、フォームの中での共通管理部分なので、特定のサブプロシージャや関数の中には記述しない。

画面表示時の初期化

画面が表示されたときには、ボタンコントロールの数は「0」なので、変数