Visual Basic .NETでプログラミング

コントロールの動的配列から継承、ActiveXコントロールまで

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_02ディレクトリに収録しています。

¥VBCTL
動的にコントロールを配置するサンプル

¥VBFORM
継承を使用してフォームを作成するサンプル

¥VBBROWSEW2
VB.NETで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）には、何種類かの言語が存在し、また、その共通ランタイムである.NET Frameworkで動作する.NET言語は、マイクロソフト社以外からも多数販売／提供されている。しかも、この何種類もある言語の中のどれを選んでも、実現できる機能は基本的には同じだ[注1]。では、その中からVisual Basic .NET（以下VB.NET）を選択する利点とはなんだろうか。その最も重要な利点とは、VB.NETがVBの文法を基本とした.NET言語であるという点につきると思う。VBが数多のWindows開発言語の中からクライアントアプリケーション構築のためのデファクトスタンダードに登りつめたのには、

①秀逸なIDE（統合開発環境）
②COMコンポーネント市場の存在
③BASIC言語の素性のよさ
④情報流通量

など、いくつかの理由がある。VBのIDEがVS.NETのIDEの母体として採用され、さらにCOMコンポーネント製品が.NET Framework対応しているために、VBの優位性の①や②については、.NET言語のすべてで共通利用できるようになったとはいえ、.NET言語の中にあってもVB.NETは③や④などの優位性を保ちつづけていると思う[注2]。

それでは、VBとVB.NETを比べたらどうだろうか。まだまだ、VBを選択してしまうケースが多いのではないだろうか。これは、Webアプリケーションなど VBには作れない（作りづらい）ものがあるとはいえ、Windowsアプリケーションでシステム構築するのならば従来のVBランタイムから.NET Frameworkに移る意義が見出せないのではないだろうか。

こういった現象が起こるのは「VB6

注1）VS.NETシリーズ以外の.NET言語については、IDEのすべての機能が使えない言語製品もあるので、どこまでIDEに対応しているかなどの情報収集は必須

注2）情報流通量ではC#がややリードか？

のままでもシステム構築できる」というのが大きいのだが、VBからVB.NETに移行すると考えたときに、そのギャップが大きいと感じているという点も見逃せない。

では、なぜギャップがあるのだろうか。ギャップはなくせなかったのだろうか。答えは「なくせなかった」のだ。VBはたしかに優秀な言語ではあったが、WindowsというOSの機能を使い切るには年々力不足になりつつあった。COMコンポーネントの追加や改変により延命はしていたが、本来持っていた便利な機能が、大規模システム開発に歯が立たなくなるときがあるという局面に対しては決定打とはなりえていなかった。

そこで登場したのが、.NET Frameworkという環境だ。この環境のすごいところは、セキュリティ管理やメモリ管理、プロセス管理やデータ管理といった、正しい形で実装しないとWindows自体にも影響をおよぼしてしまう部分を.NET Frameworkクラスライブラリという関数群として定義し、それを使うことで、誰もがある一定レベル以上の質を持つアプリケーションを構築できる点だ。

しかも、その「ある一定レベル」とは業務アプリケーションとして充分実用に耐えるレベルだ。表現を変えれば、アプリケーションの設計者やプログラマの技量に頼ることで実現していたWindowsシステムの安定稼動を、たとえスキル不足の設計者やプログラマであっても.NET Frameworkが面倒をみることでアプリケーションの安定稼動、ひいてはWindowsシステム自体が安定稼動するように設計されたものだと言えるだろう。

VBに施された言語拡張は、この.NET Frameworkクラスライブラリの恩恵を最大限に享受するためのものであり、大規模システムを構築してきたVBデベロッパーにとって、福音ともなるものだ。

VB.NETは、簡単にいってしまえば、VBの手軽さにさまざまな便利さを兼ね備えた言語だ。しかし、手軽な感じのするVBの雰囲気に比べると、クラス、クラスライブラリ、オブジェクト、オブジェクト指向といろいろ難しい言葉に隠れて、その本当の姿が見えにくくなっている。

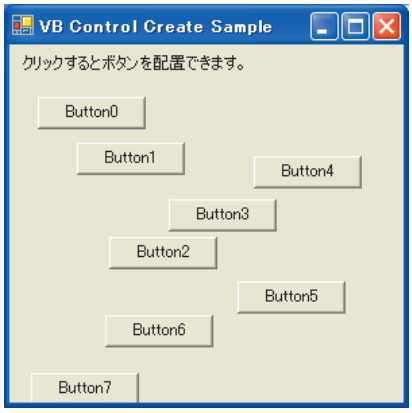
この特集では、サンプルを交えてVBでできなかったことが、VB.NETでは簡単にできるのだという例を紹介し、同時に、難しい言葉が難しい使い方を意味しているのではないことを紹介してゆきたい。

なぜならば、VBとは、実際に試して動きを理解しつつ学ぶものだからだ。

動的にコントロールを配置

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。VBでも、コントロール配列を使って、動的にコントロールを生成して配置することができた。しかし、VB.NETでは、まさに何もないところからコントロールを生成して配置することができる。なぜそのようなことが可能かといえば、コントロールも.NET Framework上で実現されている機能なので、当然、クラスとして定義されているからだ。設計時にIDEでフォーム上にコントロールを張ったとしても、内部的にはコントロールのクラスからオブジェクトを生成してフォーム上に配置するというコードが初期化ルーチンの中に定義され、実行時だけではなく、IDEのデザインウィンドウでフォームを表示するときにも呼び出されて、フォームの見た目を作り出している。そして、初期化ルーチンの中と同じコードを記述すれば、好きなときにコントロールが配置できるかもしれないと考えて、作ってみたサンプルがvbCtlサンプル（図1・リスト1）だ。

図1：ボタンコントロールの動的生成サンプル



それでは、コードの意味を追ってみよう。

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。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いけるようにしよう。ボタンコントロールを管理するので配列mactlBtnの型は、