1

これでわかる Visual Basic.NETとC#、マネージドC++プログラミング

.NET Frameworkクラスライブラリ概要

日向 俊二 *HYUGA, Shunji*

はじめに

Visual Studio.NETが登場して、多くの読者がこれまで親しんできたVisual Basic 6.0やVisual C++ 6.0（MFC、Win32プログラム）のプログラミング手法を、.NETのプログラミングの方法に変える必要性が増してきました。

従来のVisual BasicやVisual C++のプログラミングと、Visual Basic.NET、C#、マネージドC++のプログラミングは明らかに違います。しかし「プログラミング言語」と「.NET Frameworkクラスライブラリ」という2つの要素に分けて考えると、.NETプログラミングを容易に理解することができます。そのうえ、2つの要素に分けることで、プログラミングが容易になり、.NETのほかのプログラミング言語もより容易にマスターすることができるようになります。つまり、本稿を読むことで、Visual Basic.NETだけでなく、C#やマネージドC++のプログラミングもわかるというわけです。

これは読まなきゃ損だ！

本稿の前提となるもの

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

初級　中級　上級

.NETプログラムの構造

Visual Studio.NETのプログラムの構造は、これまでのWindowsのプログラムから大幅に変わりました。簡単にいえば、次のように考えることができます。

- バージョン6.0までのVisual BasicやVisual C++は、直接または間接的にWin32 APIを呼び出すことで機能していた
- Visual Basic.NETやC#、マネージドC++のプログラムは、.NET Frameworkクラスライブラリのオブジェクトを利用することで機能する

実際には.NETでもWin32 APIを呼び出すことはできるので、これはわかりやすさを最優先したかなり大胆な表現です。しかし、これが真実です。

実際、バージョン6.0までのVisual BasicやVisual C++（以下、従来のVisual Studio言語と呼びます）では、Visual Basic用のコントロールやVisual C++のMFCを使って開発するのが最も一般的な開発方法ですが、Visual Basicのコントロールや MFCの究極の目的はWin32 APIを、さらに便利に容易に使えるようにすることでした。しかし、Visual Basic.NETやC#、マネージドC++のプログラム（以下、.NET言語と呼びます）では、もはやWin32は必要なく、.NET Frameworkのライブラリと共通言語ランタイム（CLR）がプラットフォームになろうとしています。

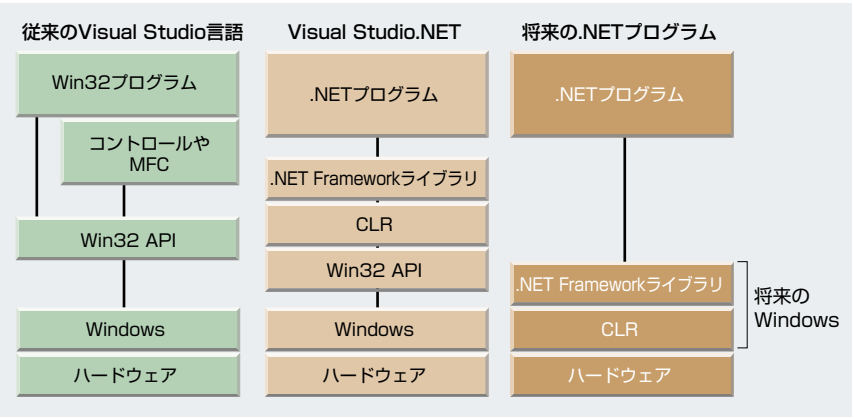
実際、近い将来、.NET Frameworkを直接サポートするWindowsが登場すると、Win32 APIは必要なくなるでしょう（図1）。

また、従来のVisual Studio言語は、言語ごとに異なる方法を使ってWin32 APIを活用していたのに対し、.NET言語では、どのプログラミング言語でも、

プログラミング言語
＋
.NET Frameworkライブラリ
＋
共通言語ランタイム（CLR）

という構造に整理されました。しかも、「.NET Frameworkライブラリ」や「共通言語ランタイム（CLR）」はプログラミング言語の種類には関係なく同じものを使うようになったと考えることができます（表1）。つまり、プログラミング言語はどれであれ、同じライブラリを同じように使うのが.NETプログラムであるといえます。

図1：Win32と.NETプログラミングの違い



注）図1は理解しやすいようにきわめて単純化しています。Win32 APIやCLRは、さらにWindowsの内部ルーチンであるネイティブAPIを呼び出していますし、その下にはデバイスドライバのようなソフトウェア層もあります。また、どのプログラムも何らかの形でそれぞれの言語のランタイムを利用していますが、それらの詳細は省略してあります。

表1：プログラミング言語とプログラミング手法

プログラミング言語	代表的なプログラミング手法
Visual Basic（6.0以前）	コントロールを介してWin32 APIを利用して機能する
Visual C++（6.0以前）	MFCを介してWin32 APIを利用して機能する
Visual Basic.NET	.NET Frameworkクラスライブラリのクラスを利用してCLRで機能する
C#	.NET Frameworkクラスライブラリのクラスを利用してCLRで機能する
マネージドC++	.NET Frameworkクラスライブラリのクラスを利用してCLRで機能する

このようになった背景には、複数の異なるプログラミング言語のプログラムをリンクしてひとつのプログラムを容易に実現できるようにするという目的もありますが、プログラムやWindows APIに対する要求が大きくなったという事実も重要です。

APIとプログラミングの変遷

Windowsが登場した当初は、複数のウィンドウに情報を表示したりそこで編集できるということだけでも大きな進歩でした。実際、Windows 2.0の頃には、サウンドもインターネットも一般には現在のように普及していませんでした。しかし、その後、Win32 APIにさまざまな機能が追加されて、現在ではサウンドやビデオ、インターネットはもちろん、XMLを活用したアプリケーションやさまざまな形態のWebアプリケーションなどなど、ソフトウェアの備える機能に対する要求が増しています。従来の非オブジェクト指向の構造のままプログラミングを続けてゆくのは、プログラムの生産性という面で大きな障害になることが明白になってきました。また、これまでのようにAPI関数を追加するという形でAPIを追加し、Windows APIを肥大させて現在と将来の状況に対処するのは現実問題として無理になってきています。