世界はオブジェクトの海に浮かぶ

.NET Frameworkで楽しむオブジェクト指向

第12回 C++/CLI －導入準備編－

επιστημη
えぴすてーめー

VC++ 2005 Express Edition

Visual Studio 2005、一足お先に期間限定ダウンロードの無償版Express Editionがリリースされました。VC#、VB、VC++、VJ#、VW（Visual Web Developer）それぞれのExpressが、

http://www.microsoft.com/japan/msdn/vstudio/express/

からダウンロードできますね。これからWindowsでのプログラミングを楽しもう／苦しもうというビギナーにはとっても嬉しいプロダクトです。コンパイラ／デバッガその他もろもろのツールにIDE（統合環境）までついてタダなんだから、マイクロソフトの太っ腹に感謝しつつ有難く頂戴してください。僕も早速頂きに参りました。ともかく全部なんもかんも、CDイメージを拾ってスピンドルで買い置きしておいたCD-Rに焼いちゃいました。

さてさてタダで遊べるVisual Studioの実力はどんなもんでしょう。僕の最も得意とするのはC++ですから、まずは“Visual C++ 2005 Express Edition”をインストールします。立ち上がるなりプロジェクトの新規作成を試みます。そこに用意されたプロジェクトの種類を眺めてみると……なんてことでしょう！　Win32アプリケーションが作れません（図1）。Win32アプリケーションが作りたければ別途Win32 Platform SDKをインストールしろとのこと。そこにある指示にしたがって設定をいじったら、Win32アプリケーションが作れるようになりました。が、肝心のリソースエディタがないのです。リソースエディタがないってことは、ボタンやラベルなどなどのコントロールを台紙に落として貼り付ける、ビジュアルなフォームデザインができないってこと。またまたどこからかリソースエディタを見つけて来にゃならんのですが、そこまでの技量をビギナーに求めるのは無理というもの（多分）。

「GUIアプリケーションは.NETでおやんなさい」というMicrosoftからのメッセージなのでしょう。VC++ 2005 Express EditionでGUIアプリを作るには“Windowsフォームアプリケーション”を選択することになります。

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#
- C++/CLI

ツール >>> Tool

- Visual Studio 2005 Professional
- Visual C++ 2005 Express
- NUnit 2.2.6

図1：Win32アプリが作れません……

さてここでまた問題。“Windowsフォームアプリケーション”は.NET Frameworkを使います。標準のC++はCLI（Common Language Infrastructure）に対応していません。そこでC++とCLIとを結びつけるのが、このバージョンで新たに導入されたC++/CLIです。C++/CLIは従来Visual Studio .NET 2003で用意されていたManaged C++の（互換性皆無な）後継で「これはホントにC++なの？」と思えるほど大幅な言語拡張がなされています。これはもうC++じゃない、C++/CLIという新たな言語と言っていいんじゃないかしら。そうなるとC/C++を多少なりともかじっているビギナーは、それに加えて新たな言語C++/CLIに慣れないことにはWindowsアプリケーションが書けないってことになります。ちょ、ちょっと辛いよね……。

とは言えC++/CLIは大幅な言語拡張がなされているものの土台はC++の上に乗っかっていますから、そんなに恐れることもありません。「C++とうまくかみ合うように調整されたC#の亜種」じゃないかと思っています。C++は“手続き型”と“オブジェクト指向”、そして“ジェネリックプログラミング”をサポートするハイブリッド言語なんですが、今回さらに“.NET”を加えた「なんでも屋」になりました。いまのところC++とC++/CLIは同居はすれども融合／統合にまでは至らない部分が多く“摺り合わせの悪さ”みたいなものを感じます。これからのVisual C++に期待しています。

あらら、お題に入る前にキレイにまとめてしまいました。

えーと、今回のお題は「επιστημηのC++/CLI導入準備」です。C++/CLIで開発を始める前に僕がやったこと――グランド整備と準備運動のお話。

“実体”と“見てくれ”を分けましょう

いきなりコード書いてコンパイルして実行して変な動きに気づいてデバッグ？　そんなの効率悪いっすよ。とくにGUIを伴うものでそんな開発スタイルじゃかったるくてやってらんない。何らかの機能をGUIアプリとして実現するとき、機能の“本体”と利用者とを繋ぐ“見てくれ（つまりGUI）”とに区別できますよね。全部作って実行／テストだと“見てくれ”を通して“本体”をテストすることになります。実行結果もまた“見てくれ”を通過したものですから、予想／期待と異なる結果が現われたとき、その原因が“本体”にあるのか“見てくれ”にあるのか、切り分けが難しくなります。加えてGUIによるテストは多くのテスト項目を通すためにテスト担当者（アナタのことです）のキーボード／マウス操作と目視チェックが必要です。自動化できません。テスト項目が100個あれば100回の操作と確認をしなきゃなりません。

この段階ではデバッガも役には立ちません。デバッガはおかしな動きを見つけたそのあとで原因を究明するためのものであって、“おかしな動き”であるか否かはテスト担当（アナタ）が判断するしかありません。おかしな動きを見つけ、原因を探り、修正を加えたらホントに正しく動くか再度テストしなければなりません。場合によってはその修正が他の部分を壊しているかもしれませんから、完璧を期するなら100個用意したテスト項目をもう一度最初からやり直すことになります。また別の不具合を見つけたら？　原因を探り修正をかけ100項目のテストを最初から……窓の外が明るくなってきました。新聞配達のバイクの音が聞こえてきます。やれやれ。

もっと楽をしましょう。“本体”のテストは入力データを揃えて関数呼んで期待する結果と照合するコードを実行すればいいからボタンひと押しで100項目のテストを自動的に（一気に）実行できます。照合に失敗したテ