モデル記述のためのプログラミング 2

Microsoft Visual C++ 6.0 の使い方

07186　井原 智彦*

平成 14 年 5 月 17 日

1　はじめに

Windows 環境で C++を学ぶ場合，Microsoft 社の Visual C++か Borland 社の Borland C++ Builder（もしくは Borland C++）を使用することがほとんどである．研究室では，GUI を手早く作成したい（RAD ツールを使いたい）という場合を除けば，Visual C++を使用するのが一般的である．

Visual C++は，Visual C++ .NET 2002 が最新版であるが，従来のバージョンと比較して仕様が大きく変更されたため，また発売されてから間もないため，今でも，Visual C++ 6.0 を用いる人の方が多い．今回は，研究室でよく用いられている Visual C++ 6.0 の操作方法について説明する．

以下は，Visual C++ 6.0（開発ツール）および msdn Library（ヘルプ）をインストールした環境について記述している．できるだけ実際に手を動かしてやってみて欲しい．ちなみに，msdn Library はハードディスクの容量が許す限りフルインストール構成でインストールした方がヘルプが引けて便利．また，Visual C++ 6.0 に関しては，Service Pack 5（修正パッチ，無償）が，Microsoft 社から発表されているので，Visual C++ 6.0 インストール後に，Service Pack 5 を当てるのが望ましい．

2　コンソールアプリケーション

コマンドプロンプト[*1]上で動作する無味乾燥なプログラム（通常，GUI を持たない）をコンソールアプリケーションと呼ぶ．

研究の場合，面倒な GUI 作成に力を入れるより，むしろ数値計算部に注力する必要があるのが普通であり，どうしても GUI が必要というのでなければ，コンソールアプリケーションで済むことが多い．

Visual C++ 6.0 では次のようにして，コンソールアプリケーションを作成する．

2.1　コンソールアプリケーションの作成手順

2.1.1　プロジェクトの作成

まず，Visual C++ 6.0 を起動すると，図 2.2 のような画面が現れる．

* 東京大学大学院工学系研究科地球システム工学専攻博士課程，　E-mail ihara@globalenv.t.u-tokyo.ac.jp

[*1] Windows9x 系 OS では，コマンドプロンプトの代わりに MS-DOS プロンプトが存在する．

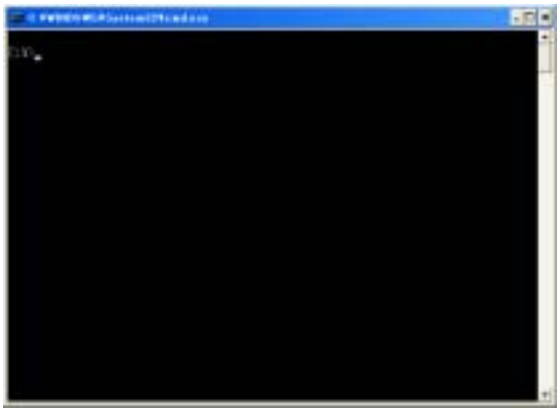

図 2.1　コマンドプロンプト

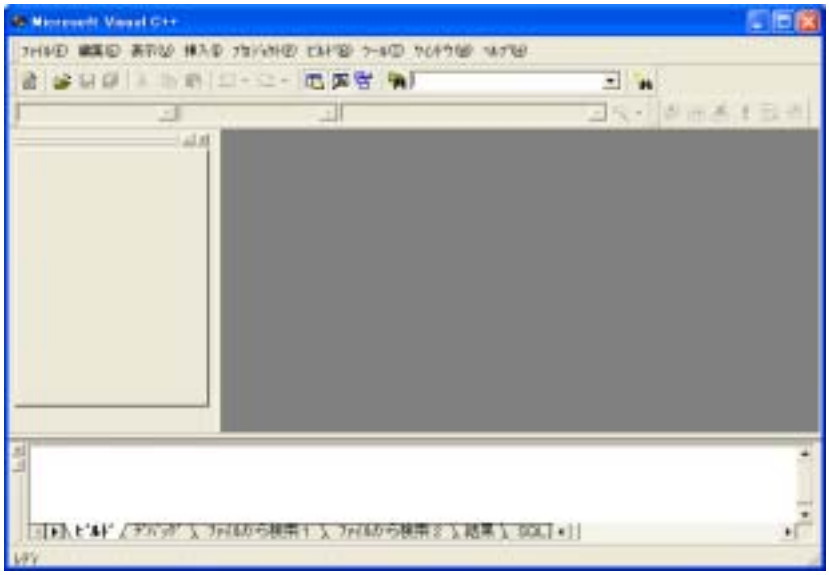

図 2.2　コンソールアプリケーションの作成 1

[ファイル (F)]→[新規作成 (N)...] を選択すると，図 2.3 のようにダイアログが現れる．始めから [プロジェクト] タブが選択されているはずである．今回は，コンソールアプリケーションの作成が目的であるので，[Win32 Console Application] を選択する．

[プロジェクト名 (N):] に適当なプロジェクト名（たとえば testApp）を入力し，[位置 (C):] に適切なパス（たとえば C:¥Documents and Settings¥someone¥My Documents¥Program¥testApp）を設定する[*2]．

続いて，出てくるダイアログにて，[空のプロジェクト (E)] をチェックし，[終了 (F)] ボタンを押す．

その次に出てくる [新規プロジェクト情報] というダイアログに，作成するプロジェクトの情報が出てくるが，ここで [OK] ボタンを押すと，図 2.4 のような画面が現れ，プロジェクトが作成された．

[*2] 先に位置を設定し，その後で，プロジェクト名を入力すると，位置の末尾が自動的にプロジェクト名と同名のフォルダとなり，便利．

図 2.3　コンソールアプリケーションの作成 2

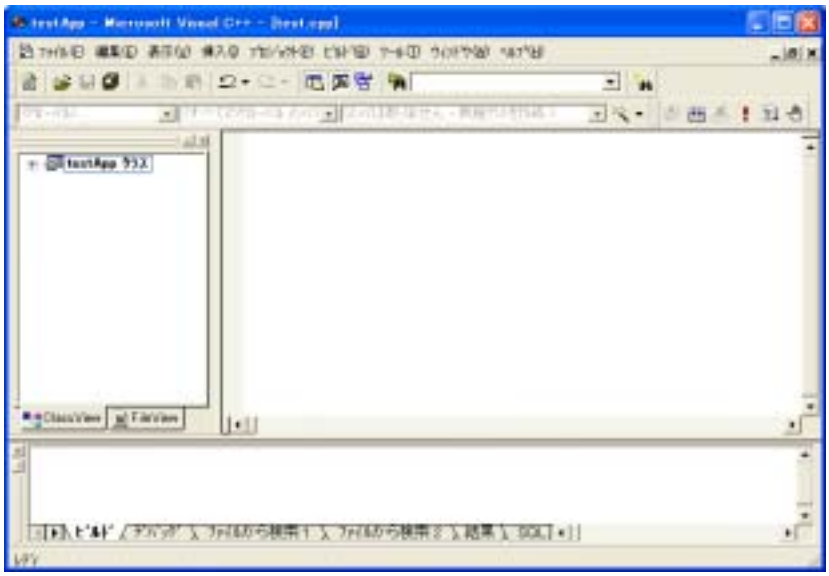

図 2.4　コンソールアプリケーションの作成 3

以上で，プロジェクトが作成されたが，これは，文字通り空のプロジェクトであり，ソースファイルもヘッダファイルも含んでいない．これにソースファイルを（必要に応じてヘッダファイルも）追加する必要がある．

一昔前と異なり，最近のプログラムは多くのソースファイルから構成されることが多い．Visual C++や Borland C++ Builder では，プロジェクトという概念を導入し，複数のソースファイルを 1 つのプロジェクトファイルでまとめている．

エクスプローラなどファイル管理ツールで見れば分かると思うが，一連の動作により，[位置] で設定したパスに，testApp.dsp というプロジェクトファイルと testApp.dsw というプロジェクトワークスペースが作成されたはずである．次にこのプロジェクトを開きたい場合は，testApp.dsw というファイルを開けばよい．

ちなみに，Visual C++ 6.0 の場合，1 つのプロジェクトファイルで，複数のソースファイル・ヘッダファイルのみならず，リソース（画像ファイルやプログラム情報）をも含めて管理している．

図 2.5　コンソールアプリケーションの作成 4

2.1.2　ソースファイルの追加

新規に作成したプロジェクト（testApp）を開いたままの状態で，さきほどと同じように，[ファイル (F)]→[新規作成 (N)...] を選択すると，図 2.5 のようにダイアログが現れる．今度は [ファイル] タブが選択されているはずである．

ここで，今回は，[C++ ソース ファイル] を選択し，[ファイル名 (N):] に test と入力する．ちなみに，デフォルトで，[プロジェクトへ追加 (A):] にチェックが入った上で対象プロジェクトが testApp となっており，かつ，[位置 (C)] はプロジェクトを新規に作成したパスが設定されているはずである．

こうすることで，プロジェクト testApp に，C++ソースファイル test.cpp が追加された．左側ペインで [FileView] をクリックすると，今開いているプロジェクトに含まれるファイルの一覧が出てくるので，そこで確認して欲しい（図 2.6）．

2.2　コンソールアプリケーションの作成・実行

2.2.1　ビルド

一番安易であるが，どの参考書にも掲載されている，いわゆる “Hello, world.” プログラムを図 2.7 のように入力する．

これを記入した後，[ビルド (B)]→[コンパイル (C)] とすれば，test.cpp がコンパイルされる．コンパイルとは，人間が読めるソースファイルをコンピュータが理解できるオブジェクトファイル（マシン語で書かれている）に翻訳することである．この役割を担うのがコンパイラである．

また，[ビルド (B)]→[ビルド (B)] とすれば，複数のオブジェクトファイルをリンクさせて，1 つの実行ファイルを生成する．この役割を担うのがリンカである．Visual C++ 6.0 の場合，いきなり，このビルドを実行しても，コンパイル → リンクの順に進めてくれる．開発ツールによっては，ビルドではなく，メイクと呼ぶ場合もある．

なお，コンパイルやリンクの最中にエラーが発生した場合，下側ペインのアウトプットウィンドウに，エ

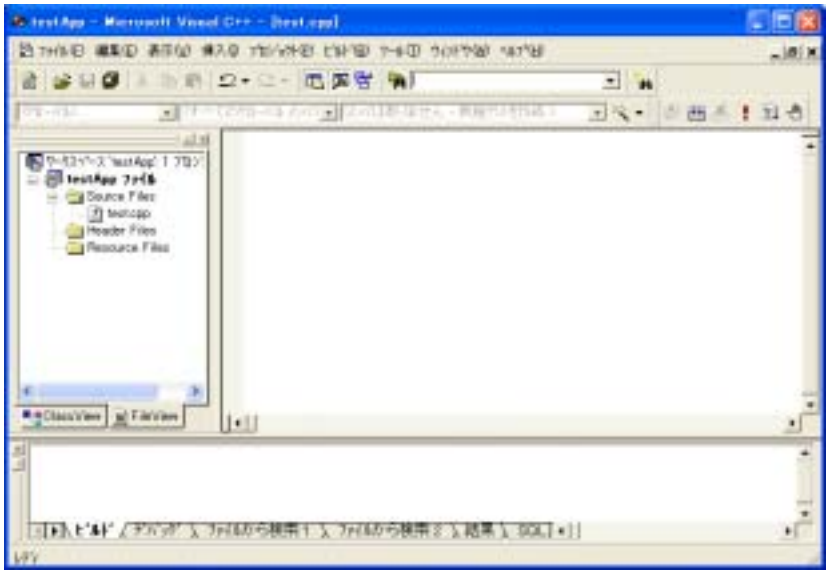

図 2.6　コンソールアプリケーションの作成 5

ラーの原因と行番号が出てくるので，慌てずに対処すること．該当箇所でダブルクリックすると，エラーの原因である行にジャンプできる．また，該当箇所で F1 キーを押せば，エラー番号と対応するヘルプ（msdn Library）のウィンドウが出現する．

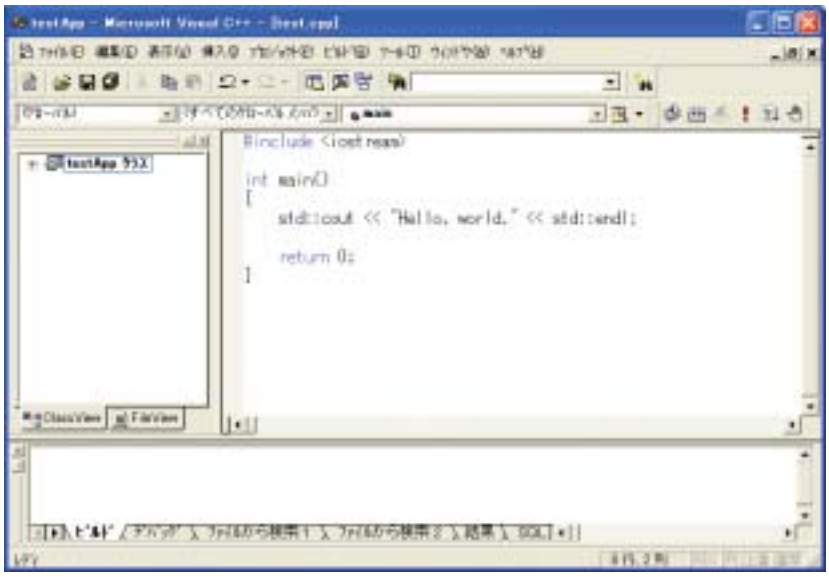

図 2.7　コンソールアプリケーションの作成 6

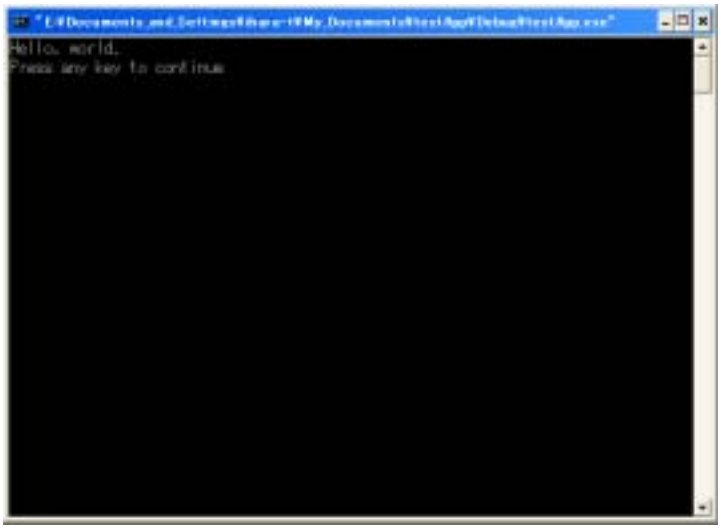

図 2.8　コンソールアプリケーションの作成 7

2.2.2　実行

最後に，[ビルド (B)]→[実行 (X)] とすると，プログラムが実行され，図 2.8 のような画面が出てくる．
……お疲れ様です．

3　その他

3.1　クラスの追加

1 クラスを 1 つのソースファイルとヘッダファイルの組で管理している人は多いと思うが，Visual C++もそれに対応して，クラスの追加を実行すると，クラスと同名のソースファイルとヘッダファイルを生成してくれる．

方法は，左側ペインで ClassView を選択し，testApp クラスの上で右クリックすると，[クラスの新規作成 (N)...] というサブメニューアイテムが出現するので，それをクリックする．図 3.1 で，適当に設定してやれば，作成したクラスに応じてソースファイルとヘッダファイルが生成される．図 3.1 では，CBuilding というクラスを作成してみた．

なお，Visual C++では，クラスの接頭辞に C を使うことを想定しており，クラス名から接頭辞の C を取り除いた名前でソースファイルとヘッダファイルを生成する．

3.2　コンソールアプリケーション以外の作成

さきほど説明したプロジェクトの新規作成画面（図 2.2）にて，[Win32 Application] を選択すれば，GUI を伴った Windows 用のアプリケーション開発用のプロジェクトが生成される．ただし，Win32 Application は，純粋な C/C++および Win32 API のみから記述することを前提にしており，高速な動作が期待できる代わり

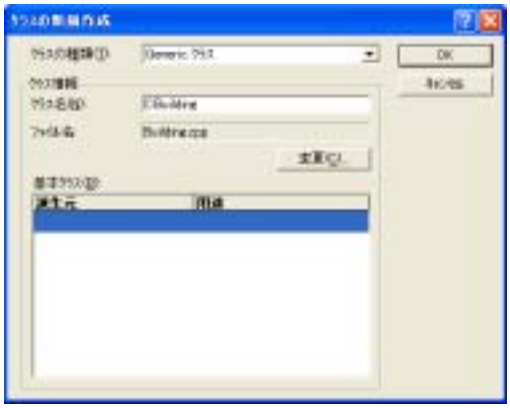

図 3.1　クラスの作成

に，かなり高度である[*3].

Visual C++ 6.0 では，不必要に面倒な作業を避けるために，MFC ライブラリを提供しており，それを用いると，C/C++ + Win32API という構成に比べると大分ましな環境で開発が可能となる．通常の開発では，[MFC AppWizard (exe)] を選択すればよい．この場合，スケルトンプログラムが生成され，少なくともスケルトンに関しては自分で開発する必要がなくなる．

4　情報源

Windows プログラミングを勉強するにあたって，情報源となるものをいくつか挙げておく．

4.1　書籍

■入門書　基本的に，まず C++を覚えることが前提となるはず．C++に関しては，前回資料を参照.

■一次資料　Visual Studio や Visual C++などに付属の msdn Library は必須である．msdn の会員になっていれば年に数回，最新版の msdn Library が提供される．また，Visual Studio など開発ツールを購入すれば，msdn Library が付属してくる．

■おすすめ　特にお薦めの本はなし．何冊か購入したが，比較的真面目に読んだ本は，以下の 2 冊だけ．MFC アプリケーションを作成する場合に便利かも．

- 林晴比古.
  新 Visual C++ 6.0 入門 ビギナー編.
  ソフトバンク パブリッシング，初版，1998.
- 林晴比古.
  新 Visual C++ 6.0 入門 シニア編.
  ソフトバンク パブリッシング，初版，1998.

[*3] ゲームプログラミングでは，独自のインターフェイスを提供するため，MFC は不要であり，Win32 Application がよく使われる．

ゲームプログラミングの場合は，全く別の系統の本が必要となる．これに関しても何冊か保有しているが，まともに読んだ本は全くないため，省略．

4.2 インターネットリソース

前回資料参照．

純粋な C++の勉強の場合は，参考書で事足りるが，MFC アプリケーションのような Windows に特化した事例では，参考書で足りない場合が少なくない．そのような場合は，メーリングリストが便利．