SKILL UP!

将来にわたって使えるスキルを身につけるために

オブジェクト指向ではじめるプログラミング

日向 俊二
HYUGA, Shunji

第2回

C#で取り組む本格オブジェクト指向プログラミング（その2）

Technology Tools

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

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¥CSHARPディレクトリに収録しています。

¥CIPHER
今回作成したサンプル

※この記事では.NET Framework SDKを利用して解説していますが、C#Builder、SharpDevelopでも利用できます（C#コンパイラならすべて利用できます）。

はじめに

C#はWindowsの開発ツールのプログラミング言語として使われるだけではなく、オブジェクト指向プログラミングをマスターするための最適なプログラミング言語のひとつでもあります。

前回はC#でプログラムを作るために必要なことについて説明し、「こんにちワン、C#」という文字列を出力する小さなプログラムを作って実行してみました。今回は、C#プログラムを作る際の基本事項「プログラムを作るために最初に知っておくべきこと」について説明します。

言語仕様とは？

C#はプログラミング言語です。

どのプログラミング言語も、やみくもに生まれるわけではありません。普通は、そのプログラミング言語の用語や概念の定義、構文の構造、主要な要素、適用範囲などについて明確に規定した文書が最初に作成されます。そして、この文書（仕様書）をもとにして、プログラムをコンパイルして実行するために必要なコンパイラやライブラリなどを作成します。この種のプログラミング言語の仕様のことを**言語仕様**といいます。

C#にも言語仕様があります。プログラマは、原則として、この言語仕様に従ってプログラムを書かなければなりません[★1]。言語仕様に従っていないプログラムは、通常、コンパイルしたとき（あるいはインタープリタで実行したとき）に、言語仕様に従っていない部分がエラーとして報告されます[★2]。

これは原則です。「原則」と断る理由は、言語仕様の解釈で立場によって違いがあったり、コンパイラが言語仕様に完全に忠実でなかったり、処理系（コンパイラやインタープリタなど）が仕様を独自に拡張している場合などがあるので、言語仕様どおりでなくてもエラーにならない場合もありえます。その端的な例が「HTMLドキュメント」

★1 専門家や経験者は、通常、プログラムコードは、入力する（enter、input）とはいわず、書くあるいは記述する（write）といいます。本稿でも、以降はプログラムを書くまたは記述するといいます。

★2 コンパイルした結果、エラーが表示されると、がっかりするかもしれません。しかし、コンパイルしたときにエラーが報告されたら、特に初心者のうちは、素直に喜んじゃいましょう。なにしろ、あなたの間違いをコンパイラが見つけてくれたわけですから。間違いをクールに指摘してくれるコンパイラがあるという点で、プログラマは幸せですね。

です。HTMLドキュメントは、特定のバージョンのHTML仕様に準拠していても、Webブラウザの種類によっては、独自のタグを使うことができたり、厳密には構文や書式に間違いがあっても許容されることがあります。

C#はプログラミング言語ですから、決められた仕様と違うことはめったに許容されません。つまり、C#のプログラムは言語仕様に忠実に従って作成しなければなりません。その理由は、プログラミングにおいて「プログラムの安定性や安全性を確保するため、厳密な決まりを守ること」が重要だからです。しかし、世の中のあらゆるものと同様に、C#の言語仕様も唯一絶対のものではありません。C#の言語仕様にも、特定のバージョンのC#コンパイラの処理と仕様との間で異なる部分や、将来のために予約されていることなどがあります。

C#の仕様とガイドライン

C#はMicrosoftのプログラミング言語であるVisual Basicとは大きく異なる点がいくつかあります。

ひとつは、「プログラミング言語C#とライブラリ部分（普通は.NET Framework）が完全に分離されている」ということです。これにより、C#はプログラミング言語としてシンプルで明快なものになります。

もうひとつは、C#の言語仕様はECMA（European Computer Manufacturer's Association）の標準仕様（ECMA-334）であるという点です。標準仕様なので、Microsoftが独自の都合と判断で臨機応変に仕様を変更することはありえません。しかし、標準仕様であるがために、コンパイラが変更されることもあります。実際、ECMAのC#の仕様に完全に準拠するために、C#のコンパイラは最初のバージョン（Visual C# .NET 2002）から、現在のバージョン（Visual C# .NET 2003）になった段階でいくつかの点が変更されています。これについては記事末の「参考リソース」に示したドキュメントで解説されているのでここでは説明しませんが、これからもコンパイラやツールの詳細が多少変更される可能性があるでしょう。

なお、Microsoftは言語仕様のほかに、名前付けやクラスライブラリの設計などにガイドラインを設けています。

言語仕様やガイドラインに注意を払うことはとても大切です。しかし、コンパイラや開発ツールを開発するのでなければ、言語仕様の詳細をすべて知っていないといけないわけではありませんし、特に理由がない限り、ガイドラインを絶対に守らなければならないわけではありません。本稿の目的は仕様やガイドラインの解説ではないので、これらについての詳細は取り上げません。必要なときに少しずつ説明することにします。

最初の段階で知っておくべきことは、プログラミング言語の構文や式の形式、コメントの形式、空行や空白、文字や改行などです。以下で前回のサンプルと新しいサンプルを使ってこれらの点を説明します。

コメントの形式

C#でサポートされているコメントは、基本的には次の2種類です。

- **1行コメント**（// コメント）
- **区切り記号付きコメント**（/* 〜 */コメント）

1行コメントは、“//”で始まり、そのソース行の行末まで続くコメントです。Microsoftのドキュメントでは「単一行コメント」とも呼ばれています。また、この形式のコメントはC言語にC++が導入されたときに導入された形式なので、「C++形式のコメント」と呼ぶ人もいます。

この種のコメントは、たとえば次のように使います。

```
// これは単純な1行コメントの例

// これは、よくみかける
// 複数行のコメントの例

///////////  この形式や  //////////

////////////////////////////////////
// このような形式で範囲や区切りを
// 明確にするために使うこともある
////////////////////////////////////
```

区切り記号付きコメントは、“/*”で始まり、“*/”で終わるコメントです。このコメントは途中に改行が含まれていてもよいので、複数行にわたる長いコメントや範囲を明確にしたいコメント、見出しや説明などを記述するとき、一連のコード行を無効にしたいときなどに、頻繁に使われる傾向があります。また、この形式のコメントはC言語で