けん太のプログラミング修行記

最終回 「けん太、マルチスレッドに挑戦する」の巻

マルチスレッドの使い方について考えてみる　ーその2ー

碗仔 けん太（Pochi Company）
WANKO, Kenta

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\KENTAディレクトリに収録しています。

\AREAVB
図形描画／面積計算プログラム（Visual Basic .NET版）

\AREACS
図形描画／面積計算プログラム（Visual C#版）

\CLOCKVB
時計プログラム（Visual Basic .NET版）

\CLOCKCS
時計プログラム（Visual C#版）

前回は、複数のプログラムを同時に実行するマルチスレッドプログラムについて調べました。マルチスレッドをうまく利用すれば、ほかの方法では困難なことも容易に実現できそうです。とはいえ、複数のプログラム部分が同時に実行されるわけですから、シングルスレッドのプログラムとは違った難しさもあるようです。今月号でも引き続いて.NETのマルチスレッドプログラミングについて調べてみます。

前回のあらすじ

ひとつのプログラムの中で、別のプログラム部分を同時に複数実行するプログラムを「マルチスレッドプログラム」と呼ぶことはよく知られています。調べてみると、プログラムの中で別のスレッドを起動する方法はとても単純でした。カンタンにいってしまえば、別のスレッドで実行したいメソッドを作り、そのアドレスを指定して、スレッドを作成するためのメソッドを呼び出すだけでOKです。

実際、ストップウォッチ付きのマルチスレッドの時計／計時プログラムをラクラク作ることができました（図1）。

でも、このプログラムを作る過程で、マルチスレッドはみかけほど単純ではなさそうであるということにも気づく結果になりました。

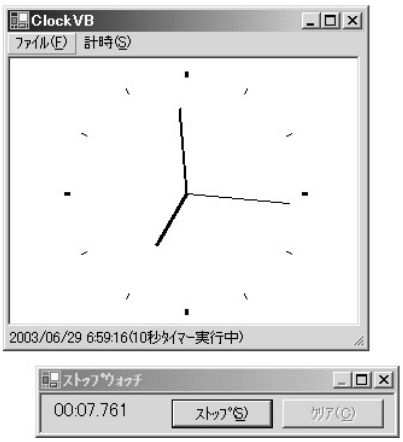
第一に、マルチスレッドの（はずの）時計／計時プログラムのストップウォッチは、実は、プログラムコードでスレッドを起動せず、ただフォームを表示しただけで、時計と並行して動作しました。ということは、フォームを使うときにはマルチスレッドは不要ということでしょうか？

加えて、ポチさんからいくつかの点を指摘されて考えてゆくうちに、新しい問題が出てきました。

スレッドから別のスレッドに情報を渡すときにはどうしたらよいのでしょうか？

さらに、ひとつの資源（たとえば、

図1：前回のサンプルClockの実行例



マウスやキーボード入力、変数を保存するためのメモリとか）を複数のスレッドで使おうとすると、「競合」と呼ばれる事態が発生することも容易に想像できます。そういう事態になったら、あるいは、そういう事態にならないようにするためには、どうすればよいのでしょうか？

いろいろ考えはじめると、マルチスレッドには、やはりマルチな（多数の、多様な……）問題がありそうです。

今回も単純なプログラムを作ってマルチスレッドについてさらに調べてみましょう。

シングルかマルチか？

前回は、ストップウォッチ付きの時計／計時プログラム、サンプルClock（ClockVBとClockCS）を作成しました。最初の計画では、ストップウォッチをメインウィンドウとは別のスレッドとして起動する予定でした。しかし、結果として、ストップウォッチのフォームはモードレスダイアログとして表示しただけで、ストップウォッチのためにとくに新しいスレッドを起動するということはしませんでした。それでも当初の目的が果たせました。そして、さらによく考えてみると、10秒を計測する部分も、別のスレッドにしなくても実現できそうな気がしてきました。そして、いま、プログラムをマルチスレッドにするかどうかということについて、根本的に考え直す必要に迫られています。

前回のサンプルですけど、自分でもどうも気に入らないんです。

マルチスレッドの勉強の材料としては、そう、まあ、悪くはないかな。

はあ（もしかして、よくもないってことか？）

とくによくはないけど。

はい（やっぱり）。だけど……。

だけど？

そもそも、あるプログラムのある部分を別のスレッドで実行するかどうか、どうやって決めるんですか？

そうだなー、与えられた条件の中で、どーしても別のスレッドにしなけりゃならない理由が出てきたら、マルチスレッドにするかなー。

はあ？　それじゃあ、必要がなかったら？

必要がなければ特別なことはなにもしないのが一番さ。

はあ（ま、そりゃそうかもしれない）。

最後の決め手は経験かもね。やってみなけりゃわかんないことも多いし、やってみた結果、どっちでも良いし、どっちにしても悪くないという場合もあるし。

そんなモンですか？

そんなモンさ。

とにかく、時間のあるときにあれこれイロイロやってみるんだね。

はあ。

というわけで、今回もあれこれ考えつつ、いろいろやってみることにします。

最初に、スレッドとユーザーの関係について最初から考えなおしてみましょう。

スレッドの存在

プログラムをマルチスレッドにする大きな理由のひとつは、バックグラウンドで何かほかの作業を行なえるようにしたいということでしょう。

普段使っているプログラムの中にも、バックグラウンドでさまざまな作業を行なえるものがあります。たとえば、Microsoft Wordは、バックグラウンドで実行するように指定すると、バックグラウンドで印刷を行ない、バックグラウンドで改ページ位置を修正し、ドキュメントを保存するようにできます。でも、ユーザーからはバックグラウンドのスレッドの存在は目に見えません。そして、バックグラウンドで実行するように指定しなければ、それはバックグラウンドではないところ、つまりフォアグラウンドで実行されます。フォアグラウンドは、通常、プライマリスレッドであり、プログラムはシングルスレッドのプログラムとして機能する