特集1
SPECIAL FEATURE
2
>>>

共通のフォームを使って「簡単、早い、かっこいい」

継承をうまく使って コンポーネントプログラミングを加速する

初音 玲 HATSUNE, Akira

はじめに

システム開発の現場では、似たような作業の繰り返しを、いかに効率良くこなすかが重要な事項のひとつとして挙げられる。

たとえば、あるシステムの業務画面をデザインする場合、“操作性の統一”という観点からデザインに共通性を持たせることが多い。統一されたデザインのアプリケーションを開発するには、開発チームにデザインガイドのようなガイドラインを用意するだけではなく、最初に共通のデザインを施した「ひな型」と呼ばれる画面などを用意しておき、その「ひな型」を元にして、業務画面を次々に作っていくことで、作業の効率化をはかる。

この方法は一見すると効率がよさそうだが、システムがある程度出来上がってきた段階で、共通デザイン部分が変更になると、業務画面の数だけ手間がかかってしまうという弱点がある(図1)。

今回は、コンポーネントプログラミングの考え方を使って、あとからの変更にも強い共通デザインの実現を考えてみたい。

レベル >>> Level
1 2 3 4 5

言語 >>> Language
- Visual Basic

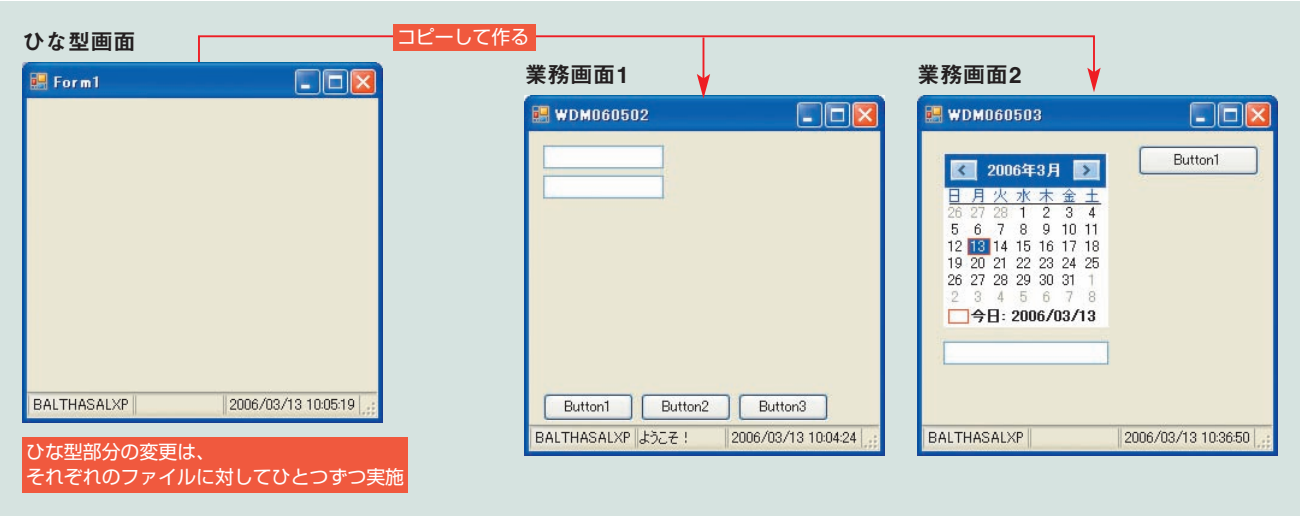
ツール >>> Tool
- Visual Studio 2005 Professional

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。

簡単、早い、かっこいい

コンポーネントプログラミングとは、同じ設計思想で作られたテスト済みの

図1：後からの変更は共通部分でも手間がかかる



コンポーネントを組み合わせてプログラミングを行なう方法で、効率よく、実際にコードを記述した内容以上の機能を持ったアプリケーションを作り上げることができる。

このようなことが可能なのは、コンポーネントの定義に次の2つの要素があるからだ。

・同じ設計思想
・テスト済み

同じ設計思想

同じ設計思想という要素は、コンポーネントプログラミングにどのような意味を与えているのだろうか。

Visual Basic 2005には、いろいろなコントロールが付属しているが、このコントロールもコンポーネントの一種だ。そのため、どのコントロールもBackColorプロパティを変更すれば背景色が変更できるし、Textプロパティに文字を設定すれば、その文字が表示できるようになっている。

このように、設計思想が同じであれば、異なるコンポーネントであっても、同じ意味を持つインターフェイス（この場合でいえば、コントロールのプロパティ／メソッド／イベント）は同じ名前になるし、デフォルト値も同じになる。

これは、コントロールを使う側にとっては非常に重要だ。初めて使うコントロールであったとしても、それまでのコントロールの使い方の知識が活用できる。つまり、

「新しく覚えること」＝
「新しいコントロール特有の機能」

であるため、覚える知識量も少なくて済むし、類似知識の重複なども避けられる。

Visual Studio 2005対応の市販コントロールにおいても、付属コントロールと同じ意味の機能には、同じ名前が割り当てられているが、あまりメジャーではないところのものだと、名前が異なっていたり、動作が異なったり、デフォルト値が異なっていたりする場合もあるので、注意が必要だ。

Visual Basic 6.0のころの実例で申し訳ないが、VB添付のラベルコントロールの機能拡張版である、ある市販コントロールは、ラベルコントロールと同

図2：VB標準コントロールと別動作の困ったコントロールの例

様にBackStyleプロパティを「透明」にすると、図2のように自分が張り付いているFrameコントロールではなく、フォームの背景色で表示されるという不整合があった。同じ名前なのに、これだけ動きが違うときの使いづらさは、簡単に想像がつくと思う。

もし、そのようなコントロールに出会ったら製造元に相違点を通知して仕様を変更してもらうようにしたほうがいいだろう。それが受け入れられないようであれば、その製品の使用中止を検討してみる必要もあるだろう。すでに使ってしまっていて中止が難しいときでも、徐々に違うコントロールに置き換えていくなどのアプローチもある。

なぜこのように厳しい方針を打ち出すかといえば、付属コントロールと同