V B . N E T

Visual Basic .NETのツボ

第10回 Visual Basic.NETの例外処理

西田 雅昭
NISHIDA, Masaaki

本連載の8回目で作成したフォーム「DataForm1」では、最初に［読み込み］ボタンをクリックすることでデータベースからデータが読み込まれました。そこで、その［読み込み］ボタン（btnLoad）のClickイベントハンドラを見てみましょう（リスト1）。

中身はTryで始まりEnd Tryで終わるブロックで構成されています。Visual Basic 6.0（以下VB6.0）に慣れた方にとって、このTry…Catch…End Tryは、はじめて見るステートメントだと思います。このステートメントは、Visual Basic.NET（以下VB.NET）ではじめて導入された“構造化例外処理”なのです。

構造化例外処理

VB6.0のOn Error XXXによるエラー処理は、少し不便でした。デバッグが面倒な場合があるほか、GoToで特定の場所に飛ばすのもコードの構造化に関しては面白くありません。

VB.NETのTryで始まる構造化例外処理は、特別な制御構造（Select CaseやWhileに似た構造）と、例外の処理、コードの保護ブロック、およびフィルタを組み合わせたもので非常に優れた内容です。

構造化例外処理の中心はTry…Catch…Finallyステートメントで、書式は図1のとおりです。

はじめてTry…Catch…Finallyステートメントを使う方もおられると思いますので、ここで簡単なプログラムを作成して実験してみましょう。

VB.NETの開発環境を起動してメニューから、

［ファイル］-［新規作成］-［プロジェクト］

と選択して新規のプロジェクトを作成します。

「新しいプロジェクト」ダイアログボックスで「テンプレート」から［Windowsアプリケーション］を選択して、「名前」にはTestTryと入力します。フォームが開いたらButtonコントロールをひとつ、TextBoxコントロールを3つ配置し、表1のようにプロパティを設定します（図2）。

配置と設定が終わったら、Buttonコントロールをダブルクリックします。するとコードエディタが開きますので、このButtonコントロール（Button1）のClickイベントハンドラをリスト2のように記述します。コードの内容につい

本稿で前提となるもの

OS	Windows XP Professional（SP1）
開発環境	Visual Basic.NET

初級 中級 上級

て、とくに説明は必要ないですね。単純な除算を行なうプログラムです。

このプログラムをデバッグ実行してみてください。そして「被除数」テキストボックスに6、「除数」テキストボックスに2を入力して［実行］ボタンをクリックすると、「商」テキストボックスに3と表示します。あたり前ですね。

では、次に「被除数」には6、「除数」には0を入力して実行してみましょう。すると、図3のようなメッセージボックスが現われます。メッセージボックスの［中断］ボタンをクリックするとコードエディタに移り、

```
Kazu3 = Kazu1 / Kazu2
```

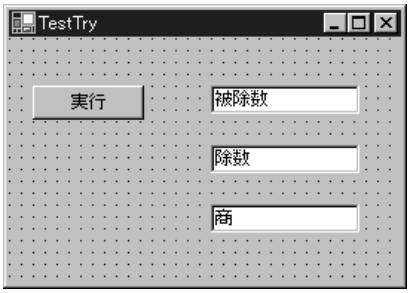
という計算式の部分でエラーが出ていることがわかります。このように、システムがエラーを出してプログラムが止まってしまうようなことをしてはいけません。

「On Error XXXでエラートラップを

表1：各コントロールの設定

オブジェクト名	Textプロパティ
Form1	TestTry
Button1	実行
TextBox1	被除数
TextBox2	除数
TextBox3	商

図2：フォーム「TestTry」のデザイン



リスト1：DataForm1にある［読み込み］ボタンのClickイベントハンドラ

```
Private Sub btnLoad_Click(ByVal sender As System.Object, _
 ByVal e As System.EventArgs) Handles btnLoad.Click
  Try
    ' データセットを読み込みます。
    Me.LoadDataSet()
  Catch eLoad As System.Exception
    ' エラー処理コードをここに追加します。
    ' エラー メッセージがある場合は表示します。
    System.Windows.Forms.MessageBox.Show(eLoad.Message)
  End Try
  Me.objdstTest1_PositionChanged()

End Sub
```

リスト2：［実行］ボタンのClickイベントハンドラ

```
Private Sub Button1_Click(ByVal sender As System.Object, _
 ByVal e As System.EventArgs) Handles Button1.Click

  Dim Kazu1 As Integer
  Dim Kazu2 As Integer
  Dim Kazu3 As Integer

  Kazu1 = TextBox1.Text
  Kazu2 = TextBox2.Text

  Kazu3 = Kazu1 / Kazu2
  TextBox3.Text = Kazu3

End Sub
```

図1：Try…Catch…Finallyステートメントの書式

```
Try
    エラーが発生する可能性のあるステートメントの並び
Catch [エラーを指定する変数名 As 例外オブジェクト型] [When条件式]
    発生したエラーを処理するステートメント
[Exit Try]
[Finally
    エラー処理がすべて行なわれた後に実行されるステートメント]
End Try
```

※［］部は省略可能

図3：0による除算を行なったときのエラー