Javaユーザーのための C#入門

乗り換えをもくろむあなたに

第3回 Windowsフォーム

矢嶋 聡 *YAJIMA, Satoshi*
NRIラーニングネットワーク株式会社
MSDN Regional Director

はじめに

前回は、コンソールアプリケーションを使いながら、基本的な構文や構造、またJavaにはない特徴のひとつである属性宣言について説明しました。これらのサンプルは、テキストベースのプログラムでしたが、3回目の今回は、GUI環境のアプリケーションらしいWindowsフォームについて説明します。Windowsフォームは、文字通りフォーム（対話型の画面）からなるアプリケーションです。ここではJavaにおけるGUI環境のアプリケーションとの比較も触れながら説明します。

また、C#もJavaもGUI環境のアプリケーションでは、イベント駆動型プログラミングが使われています。C#では、これを実現するためにJavaとは異なる「デリゲート」と呼ばれるメカニズムを使っています。デリゲートの特徴についても、後半で説明します。

前回でも触れたように、本連載では、C#の文法そのものの解説ではなく、「何が作れるのか」「どう作るのか」に重点を置いているので、Visual Studio.NETを使いながら、どんなアプリケーションができるのか、どう作るのかを見てゆきます。誌面の都合もあるので、C#のすべてを取り上げられるわけではありませんが、今後C#を修得する上での予備知識となれば幸いです。

また、本記事は対象者としてJavaプログラマの初心者を想定してはいますが、C#の入門的な内容を扱うのが主なテーマであるので、Javaプログラマに限らず役立つ内容にしました。Javaプログラマを含む、これからC#をはじめようかと検討している初級者レベルの開発者全般を対象にしていると考えてください。

本稿で前提となるもの

OS	Windows 2000 Professional（SP3）以降
開発環境	Visual Studio.NET Visual C# .NET Compiler 7.00.9466 .NET Framework 1.0 Ver 1.0.3705（SP2） Internet Explorer 6.0

初級 ― 中級 ― 上級

この記事で解説しているサンプルプログラムは、付録CD-ROMの¥DMAG¥¥CSHARPフォルダ以下に収録しています。

¥TIMECALC：本稿で解説したサンプルプロジェクト

今回のサンプル

今回作成するサンプルプログラムは、起動すると、図1のような画面が表示されます。これは、時差がある2つの地域について、一方の地域に対応する、他方の時間を簡単な一覧で表示するプ

ログラムです。たとえば、テキストボックスの中に東京から見たヨーロッパの時差である「−8」を入力して、[算出]ボタンをクリックすると、図2のように左列に東京の時間、右列に対応するヨーロッパ時間が表示されます（これを見れば東京で10:00〜18:00の間に働くとき、ヨーロッパでは02:00〜10:00なので、向こうと連携しながら仕事をする際、いつ打ち合わせをすると効率がよいかなどを判断するに役立ちます）。

また、フォームにはGUIアプリケーションでありがちなメニューバーも付

図1：サンプルプログラム起動時

図2：入力した時差を元に時間の一覧が表示される

リスト1：ファイル Form1.cs の Form1 クラスの部分の抜粋

```
public class Form1 : System.Windows.Forms.Form
{
 // 時差を記録しておくフィールド
 private int myGap = Int32.MinValue;
 // 独自の描画に利用するリソース
 private Font  myFont = new Font("Arial", 9, FontStyle.Regular);
 private Brush myBrush = Brushes.Black;                          ①
          （略）
 // 描画
 // 今回は描画を簡単にするため座標はハードコーディング
 private void Draw(Graphics g)
 {
  const int INCR = 2;  // 定数INCRの定義
  const int LINE = 16; // 定数LINEの定義
  string s1, s2;
  // i<24を満たす限りループする
  for(int i=0, posY=10;                                          ②
   i<24; i+=INCR, posY+=LINE)
  {
   s1 = String.Format("{0:00}:00", i);
   s2 = String.Format("{0:00}:00", (i+myGap+24) % 24);
   g.DrawString(s1, myFont, myBrush, 10, posY); // 文字列描画
   g.DrawString(s2, myFont, myBrush, 80, posY); // 文字列描画

  }

 // PictureBoxのPaintイベントハンドラ
 // 描画が必要なときに呼び出される
 private void pictHours_Paint(object sender,
                    System.Windows.Forms.PaintEventArgs e)
 {                                                               ③
  // 時差が指定されている場合は描画
  if(myGap != Int32.MinValue)
   Draw(e.Graphics);
 }

// ボタンのClickイベントハンドラ
private void btnCalc_Click(object sender, System.EventArgs e)
{
 try
 {
  int n = Int32.Parse(txtGap.Text
  if(n > -24 && n < 24) myGap = n;                               ④
  else myGap = Int32.MinValue;
 }
 catch
 {
  myGap = Int32.MinValue;
 }
 pictHours.Invalidate();
}

// 終了メニュー
private void mnuQuit_Click(object sender, System.EventArgs e)
{
 // 終了                                                         ⑤
 Close();
}

// 閉じる前の判断用のイベント
private void Form1_Closing(object sender,
                  System.ComponentModel.CancelEventArgs e)
{
 DialogResult res;
 res = MessageBox.Show(this, "終了してよろしいですか。",          ⑥
  "TimeCalc", MessageBoxButtons.OKCancel);
 if(res == DialogResult.Cancel)
  e.Cancel = true;
}
}
```

注）Visual Studio.NETが自動生成したコードの部分は一部省略しています。