

第48章 もうちょっとまともなイベントの例



この章では、表題のごとくもうちょっとまともなイベントのプログラムを作ります。

その前に、ConsoleクラスのKeyAvailableプロパティについて解説します。

このプロパティは、C#2.0から導入されたものです。

```
public static bool KeyAvailable { get; }
```

staticなプロパティなので、Console.KeyAvailableの形で使えます。

入力されたキーがストリームに存在するときはtrue、それ以外の時はfalseを返します。getアクセスしかないなので、自分で値を設定することはできません(当たり前ですが…)。

また、Consoleクラスには、ReadKeyメソッドがあります。

```
public static ConsoleKeyInfo ReadKey (
    bool intercept
)
public static ConsoleKeyInfo ReadKey ()
```

オーバーロードされた2つのバージョンがあります。

これは、ユーザーの押したキーを取得します。

引数なしバージョンの方は、押されたキーをコンソールに表示します。

引数付きバージョンでは、これがfalseなら画面に表示し、trueなら表示しません。(MSDNの日本語版の説明は逆になっているので注意)

戻り値はConsoleKeyInfo構造体型となっています。構造体については後の章で解説しますが、取り扱いはクラスとほとんど同じです。

この構造体はC#2.0から追加されました。

この構造体には3つのpublicなプロパティが存在します。それは、Key,KeyChar,Modifiers です。

```
public ConsoleKey Key { get; }
```

押されたConsoleKeyの値を返します。これは、ConsoleKeyは列挙体で、System名前空間で定義されています。F1とかEnterキーなども定義されています。列挙体についてはすでに第19章でやっていますね。

```
public char KeyChar { get; }
```

ユーザーが押したキーのUNICODE文字を返します。

```
public ConsoleModifiers Modifiers { get; }
```

ConsoleModifiersも列挙体です。メンバは、Alt,Control,Shiftの3つです。

Modifiersプロパティを調べると、Altキー、コントロールキー、シフトキーの押下状態がわかります。

さて、これで押されたキーがわかりますね。

この章のサンプルプログラムでは、無限ループ内でキーが押されたかどうかを調べ、キーが押されたなら、イベントを送信します。イベントが送信されたら、押されたキーを画面に表示します。xキーが押されたらプログラムを終了します。シフトキーやコントロールキーやAltキーを同時に押されたかどうかを検出します。

では、サンプルのプログラムを見てみましょう。

```
// event02.cs
```

```
using System;
```

```
delegate void EventHandler(ConsoleKeyInfo c);
```

```
class Key
{
    public event EventHandler key;

    public void OnKey(ConsoleKeyInfo k)
    {
        if (key != null)
        {
            key(k);
        }
    }
}
```

```
class event02
```

```

{
    static void Hit(ConsoleKeyInfo ki)
    {
        string strMod = "";

        if ((ki.Modifiers & ConsoleModifiers.Alt) != 0)
            strMod += "Alt+";
        if ((ki.Modifiers & ConsoleModifiers.Shift) != 0)
            strMod += "Shift+";
        if ((ki.Modifiers & ConsoleModifiers.Control) != 0)
            strMod += "Control+";

        Console.WriteLine("押されたキーは[ {0}{1} ]です", strMod, ki.Key);
    }

    public static void Main()
    {
        ConsoleKeyInfo cki;

        Key k = new Key();

        k.key += new EventHandler(Hit);

        Console.WriteLine("xで終了します");

        while (true)
        {
            if (Console.KeyAvailable)
            {
                cki = Console.ReadKey(true);
                if (cki.KeyChar == 'x')
                    break;
                k.OnKey(cki);
            }
        }
    }
}

```

前章と異なり、イベントにより実行されるメソッドがMainメソッドのあるクラスに存在します。

Mainメソッドのwhileループ内で、Console.KeyAvailableがtrueになったら、Console.ReadKeyメソッドを読んでConsoleKeyInfoを取得します。

もし、押されたキーがxならbreak文でループを抜けてプログラムは終了です。

それ以外の時は、OnKeyでイベントを発生させます。この時、引数ckiを持っていることに注意してください。

イベントハンドラのデリゲート宣言でも、ConsoleKeyInfo型引数を持っています。

イベント送信クラスであるKeyクラスもよく見てください。

public event EventHandler key;のイベントオブジェクトの宣言は前章と同じです。

次のOnKeyメソッドは、ConsoleKeyInfo型の引数kを持っています。

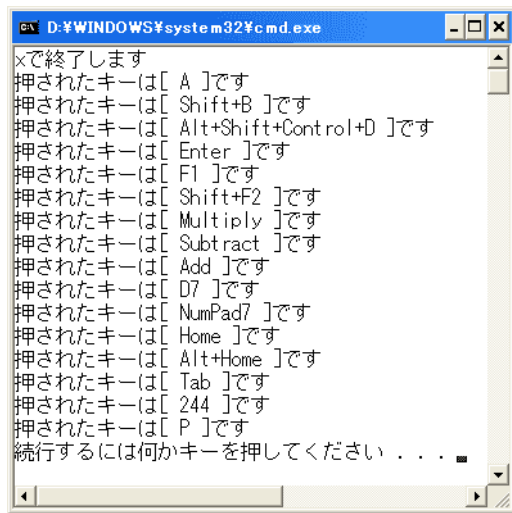
この引数は、key(k);というようにイベントオブジェクトに渡されます。

イベントが発生したら、Hitメソッドが実行されます。これは、Mainメソッドのあるクラスのメンバです。

Hitメソッドでは、渡されたConsoleKeyInfoのModifiersプロパティを調べ、Alt,Shift,Controlキーが押下されていたら、文字列strModに押下状態を追加していきます。

そして、押されたキーを表示します。

では、実行結果を見てみましょう。



「押されたキーは[244]です」とあるのは「半角/全角」キーを押したときです。

[\[C# Index\]](#) [\[総合Index\]](#) [\[Previous Chapter\]](#) [\[Next Chapter\]](#)

Update 23/Sep/2006 By Y.Kumei

当ホーム・ページの一部または全部を無断で複写、複製、転載あるいはコンピュータ等のファイルに保存することを禁じます。