アップダウン

数値の入力

多くのプログラムでは、ユーザーに何らかの数値情報の入力を要求することがあります
それは時間や年齢、サイズや長さなど、意味のある数値情報です
しかし、このような数値情報は必ず何らかの範囲が定められているでしょう

年齢を 200 と入力している場合、それは明らかに間違っているでしょうし
29時80分-24秒という時間も存在するわけがありません

このような数値情報をユーザーに入力してもらう場合
プログラムはテキストを数値に変換し、それが正しい値かを調べる必要があります
これは、以外に多くの場面で使う割りに極めて面倒な作業です

そこで、数値の入力はテキストボックスではなく**アップダウンコントロール**を使います
これは、従来スピンコントロールと呼ばれていたコントロールです

アップダウンコントロールは、**System.Windows.Forms.UpDownBase** から派生します
UpDownBase は上下2つのボタンが付いているテキストボックスを抽象化したクラスです

```
System.Object
   System.MarshalByRefObject
      System.ComponentModel.Component
         System.Windows.Forms.Control
            System.Windows.Forms.ScrollableControl
               System.Windows.Forms.ContainerControl
                  System.Windows.Forms.UpDownBase

public abstract class UpDownBase : ContainerControl
```

基本的に、このクラスは上下ボタンつきのテキストボックスを抽象化したもので
このユーザーインターフェイスをどのように用いるかという挙動は派生クラスに委ねられます

数値の入力は **System.Windows.Forms.NumericUpDown** クラスで実装されます

```
System.Object
   System.MarshalByRefObject
      System.ComponentModel.Component
         System.Windows.Forms.Control
            System.Windows.Forms.ScrollableControl
               System.Windows.Forms.ContainerControl
                  System.Windows.Forms.UpDownBase
                     System.Windows.Forms.NumericUpDown

public class NumericUpDown : UpDownBase, ISupportInitialize
```

このクラスのコンストラクタも、他のコントロール実装クラス同様に
引数を受け取らないデフォルトコンストラクタのみです

NumericUpDown では、内部で数値を decimal 型として扱っています
128ビット型の高精度な管理を行うため、大きい数字を扱うこともできます
このコントロールが管理する数値には **NumericUpDown.Value** でアクセスできます

public decimal Value {get; set;}

デフォルトで、Value は 0 に設定されます
入力可能な数値は、最大値が **NumericUpDown.Maximum** プロパティ
最小値が **NumericUpDown.Minimum** プロパティでアクセスできます

public decimal Maximum {get; set;}
public decimal Minimum {get; set;}

Maximum はデフォルトで 100、Minimum は 0 が設定されます
ユーザーは、テキストも含めて自由にコントロールに入力することができますが
プログラムが Value プロパティから値を受け取る時は適切な値に変換されます
Maximum より多い場合は Maximum に、Minimum より低い場合は Minimum になります
ただし、プログラムが Value を通して範囲外の値を代入すると例外が発生します

アップダウンコントロールについている上下ボタンを押すと
コントロールの数値がインクリメントされたり、デクリメントされたりします
この範囲は **NumericUpDown.Increment** で設定できます

public decimal Increment {get; set;}

デフォルトでは、ボタンを押すと 1 ずつ上下するようになっていますが
Increment プロパティを用いて 5 を設定すれば、5 ずつ上下するようになります

```
using System.Drawing;
using System.Windows.Forms;

public class WinMain : Form {
 NumericUpDown numericUpDown1 = new NumericUpDown();
 public static void Main() {
  Application.Run(new WinMain());
 }

 public WinMain() {
```

```
  numericUpDown1.Bounds = new Rectangle(10 , 10 , 200 , 20);
  numericUpDown1.Maximum = 999;
  numericUpDown1.Increment = 5;

  Controls.Add(numericUpDown1);
 }
}
```

図を見ると、通常のテキストボックスとは異なり
コントロールの右端に上下ボタンがあることが確認できます

Value **の値が変更されると** NumericUpDown.ValueChanged **イベントが発生します**
または、NumericUpDown.OnValueChanged() **メソッドが呼び出されます**

public event EventHandler ValueChanged;
protected virtual void OnValueChanged(EventArgs e);

ユーザーやプログラムが値を変更するとこのイベントが発生します
コントロールを継承している場合は OnValueChanged() **メソッドをオーバーライドできます**
このイベントを監視すれば、ユーザーがアップダウンボタンを押したタイミングを捉えられます

```
using System;
using System.Drawing;
using System.Windows.Forms;

public class WinMain : Form {
 NumericUpDown numericUpDown1 = new NumericUpDown();
 public static void Main() {
  Application.Run(new WinMain());
 }

 public WinMain() {
  BackColor = Color.Black;
  numericUpDown1.Bounds = new Rectangle(10 , 10 , 200 , 20);
  numericUpDown1.Maximum = 255;
  numericUpDown1.ValueChanged += new EventHandler(_ValueChanged);

  Controls.Add(numericUpDown1);
 }

 private void _ValueChanged(object sender , EventArgs e) {
  int color = (int)numericUpDown1.Value;
  BackColor = Color.FromArgb(color , color , color);
 }
}
```

このプログラムは、コントロールの値を元に背景色を変更します
背景色を変更するタイミングは、ValueChanged **イベントの発生時です**

前のページへ　戻る　次のページへ