イメージリスト

イメージのコレクションを管理

高度なグラフィックス処理アプリケーションを建築しようと考えた場合
あるいは、浮動的にイメージを管理するコントロールを作成しようと思った場合
一般的に考えられるのは、イメージを配列で扱うという方法です

オブジェクト指向であり、インデクサがサポートされる C# ではこの方法で十分ですが
Windows には、プログラマに代わってイメージのコレクションを管理してくれる機能があります
それは、標準コントロールのイメージリストコントロールです

.NET では **System.Windows.Forms.ImageList** クラスで表されます
Image クラスとは直接の関係はなく、GDI+ ではなく比較的コントロール近い存在です

```
System.Object
   System.MarshalByRefObject
      System.ComponentModel.Component
         System.Windows.Forms.ImageList

public sealed class ImageList : Component
```

このクラスのコンストラクタには、次のようなものがあります

public ImageList();
public ImageList(IContainer container);

container にはイメージリストと関連付けるコンテナを指定します
コンテナとは、他のコンポーネントを含むことができるオブジェクトを意味します
この **System.ComponentModel.IContainer** インターフェイスの実装は
Component.Container プロパティから取得することができます

public IContainer Container {get;}

Component クラスといえば Control クラスの基底クラスなので
全てのコントロールは Container プロパティからコンテナにアクセスできるのです

すでに、メニューやコントロールの設計を見てきたので想像できると思いますが
やはりイメージリストも、インスタンスにイメージを追加する形でコレクションを管理します
コレクションを管理するのは、やはり内部クラスです

System.Windows.Forms.ImageList.ImageCollection クラスは
ImageList クラスの内部で定義されている、イメージ管理用の内部クラスです

```
public sealed class ImageList.ImageCollection :
 IList , ICollection , IEnumerable
```

このクラスを利用することで、イメージリストにイメージを追加し
インデックス番号でこれを取得するようなプログラムが簡単にできるようになります
ImageCollection には **ImageList.Images** プロパティからアクセスできます

public ImageList.ImageCollection Images {get;}

イメージを追加するには **ImageCollection.Add()** を使います

public void Add(Icon value);
public void Add(Image value);
public int Add(Image value, Color transparentColor);

value には、追加する Icon 型か Image 型のイメージを指定します
transperentColor にはイメージをマスクする色を指定します

イメージリスト内のイメージのサイズは常に同じサイズにスケーリングされます
このサイズは **ImageList.ImageSize** プロパティで設定できます

public Size ImageSize {get; set;}

デフォルトで、イメージのサイズは 16 × 16 ピクセルに設定されています
このプロパティを使って、イメージリストのサイズを目的のサイズに変更してください

追加したイメージは、**ImageCollection.Item()** で取得することができます
インデックスは、やはりイメージが追加された順番に割り当てられます
現在のイメージの数は **ImageCollection.Count** プロパティで得られます

public Image this[int index] {get; set;}
public int Count {get;}

index には取得、または設定するイメージリストのイメージのインデックスを指定します
リスト内のイメージを取得して描画する時は、このようにインデクサから得てもよいのですが
ImageList.Draw() メソッドによって、簡単に描画することができます

```
public void Draw(
 Graphics g , Point pt , int index
);
public void Draw(
```

```
 Graphics g , int x , int y , int index
);
public void Draw(
 Graphics g ,
 int x , int y , int width , int height , int index
);
```

g には、描画先のデバイスコンテキストを示す Graphics オブジェクトを
x には描画する X 座標、y には Y 座標、width には幅、height には高さを指定します
index は、イメージリスト内の描画するイメージのインデックスを指定します

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

class WinMain : Form {
        int index = 0;
        ImageList img;
        public static void Main(string[] args) {
                Application.Run(new WinMain(args));
        }

        public WinMain(string[] imgFiles) {
                img = new ImageList();
                img.ImageSize = new Size(100 , 100);
                img.Images.Add(new Bitmap(imgFiles[0]));
                img.Images.Add(new Bitmap(imgFiles[1]));
        }
        override protected void OnPaint(PaintEventArgs e) {
                img.Draw(e.Graphics , 0 , 0 , index);
        }

        override protected void OnMouseEnter(System.EventArgs e)
                index = 1;
                Invalidate();
        }
        override protected void OnMouseLeave(System.EventArgs e)
                index = 0;
                Invalidate();
        }

}
```

このプログラムは、コマンドライン引数で指定した2つのイメージから
イメージリストを作成して、それをクライアント領域に描画しています
マウスがクライアント領域に入ると、インデックスを変更して表示する絵を変更します

前のページへ　戻る　次のページへ