リストボックス

項目の列挙

ファイルや、何らかの項目を列挙するにはリストボックスを使うと便利です
リストボックスは System.Windows.Forms.ListBox で表されます

```
System.Object
   System.MarshalByRefObject
      System.ComponentModel.Component
         System.Windows.Forms.Control
            System.Windows.Forms.ListControl
               System.Windows.Forms.ListBox

public class ListBox : ListControl
```

このクラスのコンストラクタは、デフォルトコンストラクタしか定義されていません
ListControl は、リストボックスのような項目列挙コントロールの基本クラスとなるものですが
内部設計上の問題で、公開されているメンバに注目するものはほとんどありません

ListBox は、追加される浮動の項目を管理するために、内部に
System.Windows.Forms.ListBox.ObjectCollection クラスを持ちます

```
public class ListBox.ObjectCollection :
 IList , ICollection , IEnumerable
```

このクラスは、これまでのコレクション用の内部クラス同様に
追加や削除、インデックスの管理などを行ってくれます
この内部クラスにアクセスするには ListBox.Items プロパティを使います

public ListBox.ObjectCollection Items {get;}

リストボックスでは、ここから ObjectCollection にアクセスして
何らかのオブジェクトをリストボックスに追加し、管理させることができます
リストボックスは、Object.ToString() メソッドが返す文字列をコントロールに表示します

オブジェクトの追加は ObjectCollection.Add() メソッド
または ObjectCollection.AddRange() メソッドで行います

public int Add(object item);

public void AddRange(object[] items);
public void AddRange(ListBox.ObjectCollection value);

item には、追加するオブジェクトを、items には追加するオブジェクトの配列を指定します
value は、追加するオブジェクトを格納している ObjectCollection を指定します
複数の項目を一度に追加するには AddRange() メソッドを使うと便利です

項目には ObjectCollection.Item インデクサからアクセスすることができます
項目の総数は ObjectCollection.Count プロパティから取得できます

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

コレクションが保有する項目を消すには ObjectCollection.Remove()
または ObjectCollection.RemoveAt() メソッドを使います
項目の全てを削除したい場合は ObjectCollection.Clear を使います

public void Remove(object value);
public void RemoveAt(int index);
public virtual void Clear();

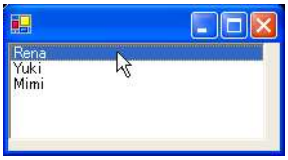
value には削除したい項目のオブジェクトを、index はインデックスを指定します

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

class WinMain : Form {
 public static void Main(string[] args) {
  Application.Run(new WinMain());
 }

 public WinMain() {
  string[] kitty = {"Rena" , "Yuki" , "Mimi"};
  ListBox lb = new ListBox();
  lb.Size = new Size(200 , 80);
  lb.Items.AddRange(kitty);

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



このプログラムは、リストボックスを作成し、これに文字列オブジェクトを追加しています
その結果、図のようにリストボックスは項目を表示します

デフォルトでは、常に1つの項目しか選択できないように設定されていますが
ListBox.SelectionMode プロパティを使って複数選択可能にすることもできます

public virtual SelectionMode SelectionMode {get; set;}

このプロパティには、セレクトモードを示す SelectionMode 列挙型メンバを指定します
この型は System.Windows.Forms.SelectionMode 列挙型です

```
[Serializable]
[ComVisible(true)]
public enum SelectionMode
```

この列挙型は、次のような意味を持つメンバを定義しています

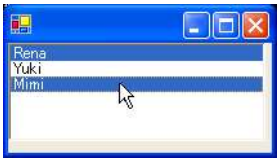
メンバ	解説
MultiExtended	複数の項目を選択できます また、Shift キー、Ctrl キー、および方向キーを使用して項目を選択できます
MultiSimple	複数の項目を選択できます
None	選択できる項目はありません
One	1 つの項目だけ選択できます

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

class WinMain : Form {
 public static void Main(string[] args) {
  Application.Run(new WinMain());
 }

 public WinMain() {
  string[] kitty = {"Rena" , "Yuki" , "Mimi"};
  ListBox lb = new ListBox();
  lb.Size = new Size(200 , 80);
  lb.Items.AddRange(kitty);
  lb.SelectionMode = SelectionMode.MultiExtended;

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



このプログラムは、SelectionMode プロパティを使って複数選択可能なリストボックスを作成します
Shift キーや Ctrl キーを使って、図のように複数の項目を選択することができます

現在選択されている項目を得るには ListBox.SelectedItem プロパティ
または ListBox.SelectedItems プロパティを使います

public object SelectedItem {get; set;}
public ListBox.SelectedObjectCollection SelectedItems {get;}

SelectedItems プロパティは、複数選択可能リストボックスでのみ使用します
この場合は、項目ではなく選択されている項目のコレクションを返します
SelectedItem プロパティは、項目が選択されていない場合は null を示します

System.Windows.Forms.ListBox.SelectedObjectCollection クラスは
選択されているオブジェクトのコレクションを管理するクラスです

```
public class ListBox.SelectedObjectCollection :
 IList , ICollection , IEnumerable
```

選択されている項目数は SelectedObjectCollection.Count プロパティから
選択されている項目は SelectedObjectCollection.Item インデクサから得られます

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

選択されている項目は、インデックスで取得することも可能です
この場合は ListBox.SelectedIndex プロパティ
または ListBox.SelectedIndices プロパティを使います

public override int SelectedIndex {get; set;}
public ListBox.SelectedIndexCollection SelectedIndices {get;}

SelectedIndex は選択されている項目のインデックスを返します
項目が選択されていない場合は -1 を示します
SelectedIndices プロパティは、選択されている項目の
インデックスを格納しているコレクションを返します

System.Windows.Forms.ListBox.SelectedIndexCollection クラスは
選択されている項目のインデックスのコレクションを管理するクラスです

```
public class ListBox.SelectedIndexCollection :
 IList , ICollection , IEnumerable
```

選択されている項目数は **SelectedIndexCollection.Count** プロパティから
選択されている項目は **SelectedIndexCollection.Item** インデクサから得られます

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

仕組みは、他のコレクション管理クラスと同じなので理解しやすいでしょう
これらを使えば、複数選択可能リストから正確に選択されている項目を得られます

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

class WinMain : Form {
 Image img;
 public static void Main(string[] args) {
  Application.Run(new WinMain(args));
 }

 public WinMain(string[] file) {
  Image[] img = new Image[file.Length];
  for(int i = 0 ; i < file.Length ; i++) img[i] = new Bitmap(file[i]);

  ListBox lb = new ListBox();
  lb.Size = new Size(200 , 80);
  lb.Items.AddRange(img);
  lb.Click += new System.EventHandler(_Select);
  Controls.Add(lb);
 }

 public void _Select(object sender , System.EventArgs e) {
  ListBox lb = (ListBox)sender;
  img = (Image)lb.SelectedItem;
  Invalidate();
 }
 override protected void OnPaint(PaintEventArgs e) {
  if (img != null) e.Graphics.DrawImage(img , 0 , 80 , 200 , 200);
 }
}
```

このプログラムは、まずコマンドライン引数から静止画のファイル名を入力します
入力する静止画の数は任意なので、複数個入力してください
すると、ファイル名を元に Image オブジェクトの配列を作成し、これをリストに列挙します

リストをクリックするとイベントが発生し、現在選択されているオブジェクトを抽出します
そして、選択されているイメージオブジェクトをクライアント領域に描画しているのです

チェックリスト

リストボックスの項目にチェックボックスのような機能をつけたいと思うかもしれません
例えば、ダウンロード支援ソフトのインターフェイスを考えて見ましょう
HTML ファイルを読み込み、ダウンロードが可能な URI を列挙します
このうちダウンロードするべきファイルをユーザーに選択してもらうとすれば
複数選択可能リストボックスより、チェックボックス型リストボックスのほうが便利です

System.Windows.Forms.CheckedListBox はリストボックスを拡張し
項目に文字列だけではなく、チェックボックスを表示できるようにしたクラスです

これを使えば、リストボックスの項目をチェックする機能を利用することができます

```
System.Object
   System.MarshalByRefObject
      System.ComponentModel.Component
         System.Windows.Forms.Control
            System.Windows.Forms.ListControl
               System.Windows.Forms.ListBox
                  System.Windows.Forms.CheckedListBox

public class CheckedListBox : ListBox
```

このクラスのコンストラクタは、デフォルトコンストラクタしか定義されていません

チェックリストの項目を管理するのは、ListBox.ObjectCollection を継承した
System.Windows.Forms.CheckedListBox.ObjectCollection です

```
System.Object
   System.Windows.Forms.ListBox.ObjectCollection
      System.Windows.Forms.CheckedListBox.ObjectCollection

public class CheckedListBox.ObjectCollection : ListBox.ObjectCollection
```

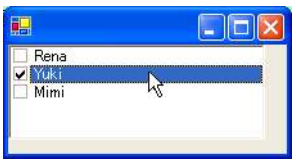
基本的に、機能は ListBox.ObjectCollection と変わりません
やはり、Items プロパティを使ってこれにアクセスします

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

class WinMain : Form {
 public static void Main(string[] args) {
  Application.Run(new WinMain());
 }

 public WinMain() {
  string[] kitty = {"Rena" , "Yuki" , "Mimi"};
  CheckedListBox lb = new CheckedListBox();
  lb.Size = new Size(200 , 80);
  lb.Items.AddRange(kitty);

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



リストボックスのプログラムを改良して、チェックリストを追加するように変更しました
図のように、項目にチェックボックスがついたリストを表示します

デフォルトでは、チェックボックスにチェックを入れたりはずしたりするには
一度項目を選択して、その後再びクリックしなければなりません
すぐにチェックしたい場合は CheckedListBox.CheckOnClick を設定します

public bool CheckOnClick {get; set;}

このプロパティはデフォルトで false ですが
true を指定すれば、項目の選択と同時にチェックされます

チェックされている項目を知るには CheckedListBox.CheckedIndices を
項目を得るには CheckedListBox.CheckedItems プロパティを使います

public CheckedListBox.CheckedIndexCollection CheckedIndices {get;}
public CheckedListBox.CheckedItemCollection CheckedItems {get;}

CheckedIndices は、チェック済みの項目のインデックスをコレクションしている
System.Windows.Forms.CheckedListBox.CheckedIndexCollection を返します

```
public class CheckedListBox.CheckedIndexCollection :
 IList , ICollection , IEnumerable
```

このクラスのコンストラクタは隠蔽されているので、インスタンスは作れません
インデックスを得るために、このクラスはインデクサとプロパティを提供しています
チェック項目の数は CheckedIndexCollection.Count プロパティで
インデックスを得るには CheckedIndexCollection.Item インデクサを使います

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

同様に CheckedItemCollection プロパティは、チェック済み項目のコレクション
System.Windows.Forms.CheckedListBox.CheckedItemCollection を返します

```
public class CheckedListBox.CheckedItemCollection :
 IList , ICollection , IEnumerable
```

このクラスもまた、コンストラクタを呼び出すことはできません
チェック項目の数は CheckedItemCollection.Countプロパティで

チェック項目は CheckedItemCollection.Item インデクサで得られます

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

これらの機能を使えば、正しくチェック済みの項目を取得し
適切な処理を施すことができるようになるでしょう

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

class WinMain : Form {
 CheckedListBox.CheckedItemCollection cll;
 public static void Main(string[] args) {
  Application.Run(new WinMain());
 }

 public WinMain() {
  string[] kitty = {"Rena" , "Yuki" , "Mimi"};
  CheckedListBox lb = new CheckedListBox();
  lb.Size = new Size(200 , 80);
  lb.Items.AddRange(kitty);
  lb.CheckOnClick = true;
  lb.Click += new System.EventHandler(OnClick);

  Controls.Add(lb);
 }

 override protected void OnPaint(PaintEventArgs e) {
  Graphics g = e.Graphics;
  if (cll == null) return;

  for(int i = 0 ; i < cll.Count ; i++)
   g.DrawString("Checked = " + cll[i] ,
    Font , Brushes.Black , 200 , Font.Height * i);
 }

 public void OnClick(object sender , System.EventArgs e) {
  cll = ((CheckedListBox)sender).CheckedItems;
  Invalidate();
 }
}
```

このプログラムは、チェック済み項目をリストの横に表示します
クリックイベントが発生すると、まずコレクションをフィールドに保存して再描画します
OnPaint() メソッドでは、このコレクションから項目の情報を得て表示しています

前のページへ　　戻る　　次のページへ