メニュー

メニューバーの作成

メニューバーは、ユーザーとの対話に必要不可欠な機能の一つです
アプリケーションは、どの機能を実行させるかをユーザーに選択させる最良の手段として
ウィンドウの上部にメニューバーをつけて対話することができるのです

Win32 API でプログラムからメニューを作成すると、ものすごく面倒でした
メニューを作成するために、構造体を設定して関数でメニューオブジェクトを作りました
しかし .NET はオブジェクト指向なので、より美しい構造にまとめられています

まず、全てのメニューは **System.Windows.Forms.Menu** を基底とします

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

public abstract class Menu : Component
```

見てのとおり、このクラスは抽象クラスです
メニューバーやメミュー項目は、すべてこのクラスを継承しているのです
最も重要なのは、このクラスの **Menu.MenuItems** プロパティです

public Menu.MenuItemCollection MenuItems {get;}

このプロパティが返す MenuItemCollection 型は Menu クラスの内部クラスです
System.Windows.Forms.Menu.MenuItemCollection クラスは次のようになっています

```
public class Menu.MenuItemCollection :
 IList , ICollection , IEnumerable
```

このクラスは、メニュー項目を管理する非常に重要な役割を持っています
Menu クラスを継承するクラスは、MenuItems プロパティからこのクラスにアクセスすることができます
このクラスの細かい機能については、すぐ後で詳しく説明します

では、実際にインスタンスを作ってメニューバーを作成するにはどうすればよいでしょうか
メニューバーは **System.Windows.Forms.MainMenu** で表されます

```
System.Object
   System.MarshalByRefObject
      System.ComponentModel.Component
         System.Windows.Forms.Menu
            System.Windows.Forms.MainMenu

public class MainMenu : Menu
```

このクラスのコンストラクタは、次のように定義されています

public MainMenu();
public MainMenu(MenuItem[] items);

items には、新しく作成されたメニューに追加する項目を指定します
MenuItem 型については、すぐ後で詳しく説明します

インスタンスを生成できれば、これで MenuItemCollection にアクセスできます
作成したメニューに新しく項目を追加するには MenuItems プロパティから
MenuItemCollection.Add() メソッドにアクセスして追加します

```
public virtual int Add(MenuItem item);
public virtual MenuItem Add(string caption);
public virtual int Add(int index , MenuItem item);
public virtual MenuItem Add(string caption , EventHandler onClick);
public virtual MenuItem Add(string caption , MenuItem[] items);
```

item には追加する項目を示す MenuItem を
caption には、メニュー項目として追加する文字列を指定します
index は新しく追加するメニュー項目の位置を示すインデックス番号を指定します
これを指定しなければ、新しい項目はメニューの末尾に追加されます

onClick はメニューをユーザーが選択した時に発生するイベントを指定します
items は、このメニュー項目に含まれる新しい MenuItem オブジェクトの配列を指定します

戻り値が int 型のメソッドは、項目が追加されたインデックスを返します
MenuItem を返すメソッドは、コレクションに追加されているメニュー項目を返します

とりあえず、今は文字列を使ってメニュー項目を追加してみましょう
作成した MainMenu オブジェクトをウィンドウに設定するには
Form.Menu プロパティを使って取得したり設定することができます

public MainMenu Menu {get; set;}

では、まず簡単なメニューバーを作ってみましょう

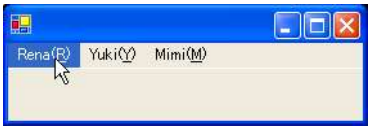
```
using System.Windows.Forms;

class WinMain : Form {
```

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

  public WinMain() {
   MainMenu mm = new MainMenu();
   mm.MenuItems.Add("Rena(&R)");
   mm.MenuItems.Add("Yuki(&Y)");
   mm.MenuItems.Add("Mimi(&M)");

   Menu = mm;
  }
 }
```



このプログラムを実行すると、3つの項目があるメニューバー付きのウィンドウが表示されます
因みに、メニューは Alt キー + 文字キーで操作 することができます
これは、デフォルトで Alt キー + アイテムの先頭文字 でアイテムが選択されます
しかし、メニューアイテムの名前に & を指定 すると直後の文字に下線がつき
Alt キーとその直後の文字（下線付き文字）でアイテムを選択できるようになります

メニュー項目は、文字列で指定するほかに
System.Windows.Forms.MenuItem クラスを用いて表現することができます
こちらを使ったほうが、オブジェクト指向プログラミングとして美しい形でしょう
このクラスはメニュー項目とイベント、ショートカットなどがカプセルかされているからです

```
System.Object
   System.MarshalByRefObject
      System.ComponentModel.Component
         System.Windows.Forms.Menu
            System.Windows.Forms.MenuItem

public class MenuItem : Menu
```

このクラスのコンストラクタは次のように定義されています

```
public MenuItem();
public MenuItem(string text);
public MenuItem(string text , EventHandler onClick);
public MenuItem(string text , MenuItem[] items);

public MenuItem(
 string text , EventHandler onClick , Shortcut shortcut
);
public MenuItem(
 MenuMerge mergeType , int mergeOrder ,
 Shortcut shortcut , string text ,
 EventHandler onClick , EventHandler onPopup ,
 EventHandler onSelect , MenuItem[] items
);
```

text には、メニュー項目の文字列を指定します
onClick は、メニュー選択時に発生するイベントを指定します
items は、このメニュー項目のサブメニューを示すオブジェクトの配列を指定します

shortcut には、メニュー項目で使用できるショートカットキーを指定します
mergeType は別のメニューの項目にマージされたときの MenuItem の動作を、
mergeOrder はそのメニュー項目がマージ後のメニューで占める位置を相対的に示す値を指定します
onPopup は Popup イベント、onSelect は Select イベントを処理するハンドラを指定します
これらのイベントについて、詳細は後期します

メニュー項目には、ショートカットキーを割り当てることができます
これは System.Windows.Forms.Shortcut 列挙型で指定できます

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

この列挙型は、次のようなメンバを定義しています

メンバ	ショートカット条件	メンバ	ショートカット条件
Alt0	Alt + 0	Alt1	Alt + 1
Alt2	Alt + 2	Alt3	Alt + 3
Alt4	Alt + 4	Alt5	Alt + 5
Alt6	Alt + 6	Alt7	Alt + 7
Alt8	Alt + 8	Alt9	Alt + 9
AltBksp	Alt + BackSpace	AltF1	Alt + F1
AltF10	Alt + F10	AltF11	Alt + F11
AltF12	Alt + F12	AltF2	Alt + F2
AltF3	Alt + F3	AltF4	Alt + F4
AltF5	Alt + F5	AltF6	Alt + F6
AltF7	Alt + F7	AltF8	Alt + F8

AltF9	Alt + F9	Ctrl0	Ctrl + 0
Ctrl1	Ctrl + 1	Ctrl2	Ctrl + 2
Ctrl3	Ctrl + 3	Ctrl4	Ctrl + 4
Ctrl5	Ctrl + 5	Ctrl6	Ctrl + 6
Ctrl7	Ctrl + 7	Ctrl8	Ctrl + 8
Ctrl9	Ctrl + 9	CtrlA	Ctrl + A
CtrlB	Ctrl + B	CtrlC	Ctrl + C
CtrlD	Ctrl + D	CtrlDel	Ctrl + Delete
CtrlE	Ctrl + E	CtrlF	Ctrl + F
CtrlF1	Ctrl + F1	CtrlF10	Ctrl + F10
CtrlF11	Ctrl + F11	CtrlF12	Ctrl + F12
CtrlF2	Ctrl + F2	CtrlF3	Ctrl + F3
CtrlF4	Ctrl + F4	CtrlF5	Ctrl + F5
CtrlF6	Ctrl + F6	CtrlF7	Ctrl + F7
CtrlF8	Ctrl + F8	CtrlF9	Ctrl + F9
CtrlG	Ctrl + G	CtrlH	Ctrl + H
CtrlI	Ctrl + I	CtrlIns	Ctrl + Insert
CtrlJ	Ctrl + J	CtrlK	Ctrl + K
CtrlL	Ctrl + L	CtrlM	Ctrl + M
CtrlN	Ctrl + N	CtrlO	Ctrl + O
CtrlP	Ctrl + P	CtrlQ	Ctrl + Q
CtrlR	Ctrl + R	CtrlS	Ctrl + S
CtrlShift0	Ctrl + Shift + 0	CtrlShift1	Ctrl + Shift + 1
CtrlShift2	Ctrl + Shift + 2	CtrlShift3	Ctrl + Shift + 3
CtrlShift4	Ctrl + Shift + 4	CtrlShift5	Ctrl + Shift + 5
CtrlShift6	Ctrl + Shift + 6	CtrlShift7	Ctrl + Shift + 7
CtrlShift8	Ctrl + Shift + 8	CtrlShift9	Ctrl + Shift + 9
CtrlShiftA	Ctrl + Shift + A	CtrlShiftB	Ctrl + Shift + B
CtrlShiftC	Ctrl + Shift + C	CtrlShiftD	Ctrl + Shift + D
CtrlShiftE	Ctrl + Shift + E	CtrlShiftF	Ctrl + Shift + F
CtrlShiftF1	Ctrl + Shift + F1	CtrlShiftF10	Ctrl + Shift + F10
CtrlShiftF11	Ctrl + Shift + F11	CtrlShiftF12	Ctrl + Shift + F12
CtrlShiftF2	Ctrl + Shift + F2	CtrlShiftF3	Ctrl + Shift + F3
CtrlShiftF4	Ctrl + Shift + F4	CtrlShiftF5	Ctrl + Shift + F5
CtrlShiftF6	Ctrl + Shift + F6	CtrlShiftF7	Ctrl + Shift + F7
CtrlShiftF8	Ctrl + Shift + F8	CtrlShiftF9	Ctrl + Shift + F9
CtrlShiftG	Ctrl + Shift + G	CtrlShiftH	Ctrl + Shift + H
CtrlShiftI	Ctrl + Shift + I	CtrlShiftJ	Ctrl + Shift + J
CtrlShiftK	Ctrl + Shift + K	CtrlShiftL	Ctrl + Shift + L
CtrlShiftM	Ctrl + Shift + M	CtrlShiftN	Ctrl + Shift + N
CtrlShiftO	Ctrl + Shift + O	CtrlShiftP	Ctrl + Shift + P
CtrlShiftQ	Ctrl + Shift + Q	CtrlShiftR	Ctrl + Shift + R
CtrlShiftS	Ctrl + Shift + S	CtrlShiftT	Ctrl + Shift + T
CtrlShiftU	Ctrl + Shift + U	CtrlShiftV	Ctrl + Shift + V
CtrlShiftW	Ctrl + Shift + W	CtrlShiftX	Ctrl + Shift + X
CtrlShiftY	Ctrl + Shift + Y	CtrlShiftZ	Ctrl + Shift + Z
CtrlT	Ctrl + T	CtrlU	Ctrl + U
CtrlV	Ctrl + V	CtrlW	Ctrl + W
CtrlX	Ctrl + X	CtrlY	Ctrl + Y
CtrlZ	Ctrl + Z	Del	Delete
F1	F1	F10	F10
F11	F11	F12	F12
F2	F2	F3	F3
F4	F4	F5	F5
F6	F6	F7	F7
F8	F8	F9	F9
Ins	Insert	None	**ショートカット キーなし**
ShiftDel	Shift + Delete	ShiftF1	Shift + F1
ShiftF10	Shift + F10	ShiftF11	Shift + F11
ShiftF12	Shift + F12	ShiftF2	Shift + F2
ShiftF3	Shift + F3	ShiftF4	Shift + F4
ShiftF5	Shift + F5	ShiftF6	Shift + F6
ShiftF7	Shift + F7	ShiftF8	Shift + F8

ShiftF9	Shift + F9	ShiftIns	Shift + Insert

メニュー項目にショートカットを割り当てる場合に、これらの列挙型メンバを指定します
頻繁に使われると想定されるメニュー項目には、ショートカットを割り当ててください
特に、熟練の利用者はマウスではなくキーボードを中心に入力する傾向があります

メニュー項目が別のメニューの項目に組み合わされた時の
この MenuItem の動作を指定するには
System.Windows.Forms.MenuMerge を使います

```
[Serializable]
public enum MenuMerge
```

この列挙型メンバは、次のように定義されています

メンバ	解説
Add	MenuItem は、マージ後のメニュー内にある 既存の MenuItem オブジェクトのコレクションに追加されます
MergeItems	この MenuItem のすべてのサブメニュー項目は マージ後のメニューで同じ位置にある 既存の MenuItem オブジェクトのサブメニュー項目にマージされます
Remove	MenuItem は、マージ後のメニューには含まれません
Replace	MenuItem は、マージ後のメニューで 同じ位置にある既存の MenuItem と置換されます

ショートカットキーは **MenuItem.Shortcut** プロパティで
マージタイプについては **MenuItem.MergeType** プロパティで
マージ後の位置は **MenuItem.MeregeOrder** プロパティでも設定できます

public Shortcut Shortcut {get; set;}
public MenuMerge MergeType {get; set;}
public int MergeOrder {get; set;}

同様に、メニューの文字列も **MenuItem.Text** プロパティを使って
取得したり、新しく設定することができます

public string Text {get; set;}

これらのプロパティを使えば、柔軟にメニュー項目の設定を変更できます
さっそく、MenuItem クラスを使ったメニューを作ってみましょう

```
using System.Windows.Forms;

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

 public WinMain() {
  MainMenu mm = new MainMenu();
  MenuItem rena = new MenuItem("Rena");
  MenuItem yuki = new MenuItem("Yuki");
  MenuItem mimi = new MenuItem("Mimi");

  mm.MenuItems.Add(rena);
  mm.MenuItems.Add(yuki);
  mm.MenuItems.Add(mimi);

  Menu = mm;
 }
}
```

このプログラムを実行すると、先ほどのように
メニューバーがついているウィンドウが表示されます
基本的には同じですが、MenuItem を使っているというところで違います

前のページへ　　戻る　　次のページへ