



# How To WPF アプリケーション Part4 By 中博俊

## WPFとは

- いちおうWindows Vista向けに出荷された.NET3.0で、初搭載されたWindowsクライアントプログラムの一形態
- 現在の最新バージョンは3.5
- いちおうWindows Formsの置き換え
- クライアントの表現を豊かにしてくれるツールとしてこれから期待しています。
- XAMLというものでUIを記述することができます。
- SilverlightでもXAMLを使います。

## おさらい

- 第1回はButtonとTextBoxを使った簡単なアプリケーションの作り方でした。
- WPFアプリケーションはDocument-Viewとして完全にUIとドキュメントを分けて考えましょう。
- そしてINotifyPropertyChangedのインターフェイスを究めようという内容でした。
- NotifyPropertyChangedBaseは今回も出てきますのでおさらい

## NotifyPropertyChangedBase

```
public class NotifyPropertyChangedBase :  
    INotifyPropertyChanged {  
    public event PropertyChangedEventHandler  
        PropertyChanged;  
    protected void FirePropertyChanged(  
        string PropertyName) {  
        if (this.PropertyChanged != null) {  
            this.PropertyChanged(this,  
                new PropertyChangedEventArgs(PropertyName));  
        }  
    }  
}
```



## おさらい

- 第2回はコンバータと、ラジオボタンの制御についてでした。
- コンバータはIValueConverterを使い、ConvertとConvertBackを用意するところを解説しました。
- 第3回はコンボボックスを使いAncestorを使いVisual Treeをちょっとだけ操作してみました。

## 今回のアジェンダ

- ListView

- 一覧を表示しよう

- ソートしよう

- 編集しよう

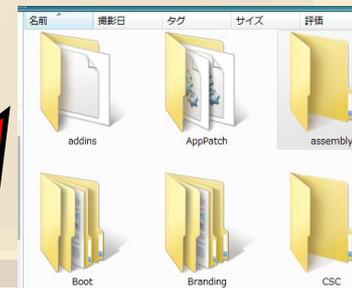
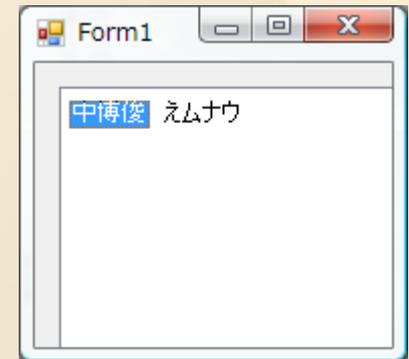
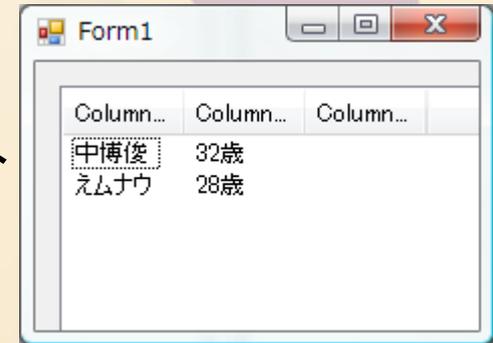
- レコード追加しよう

- データグリッドなくてもいいよね？

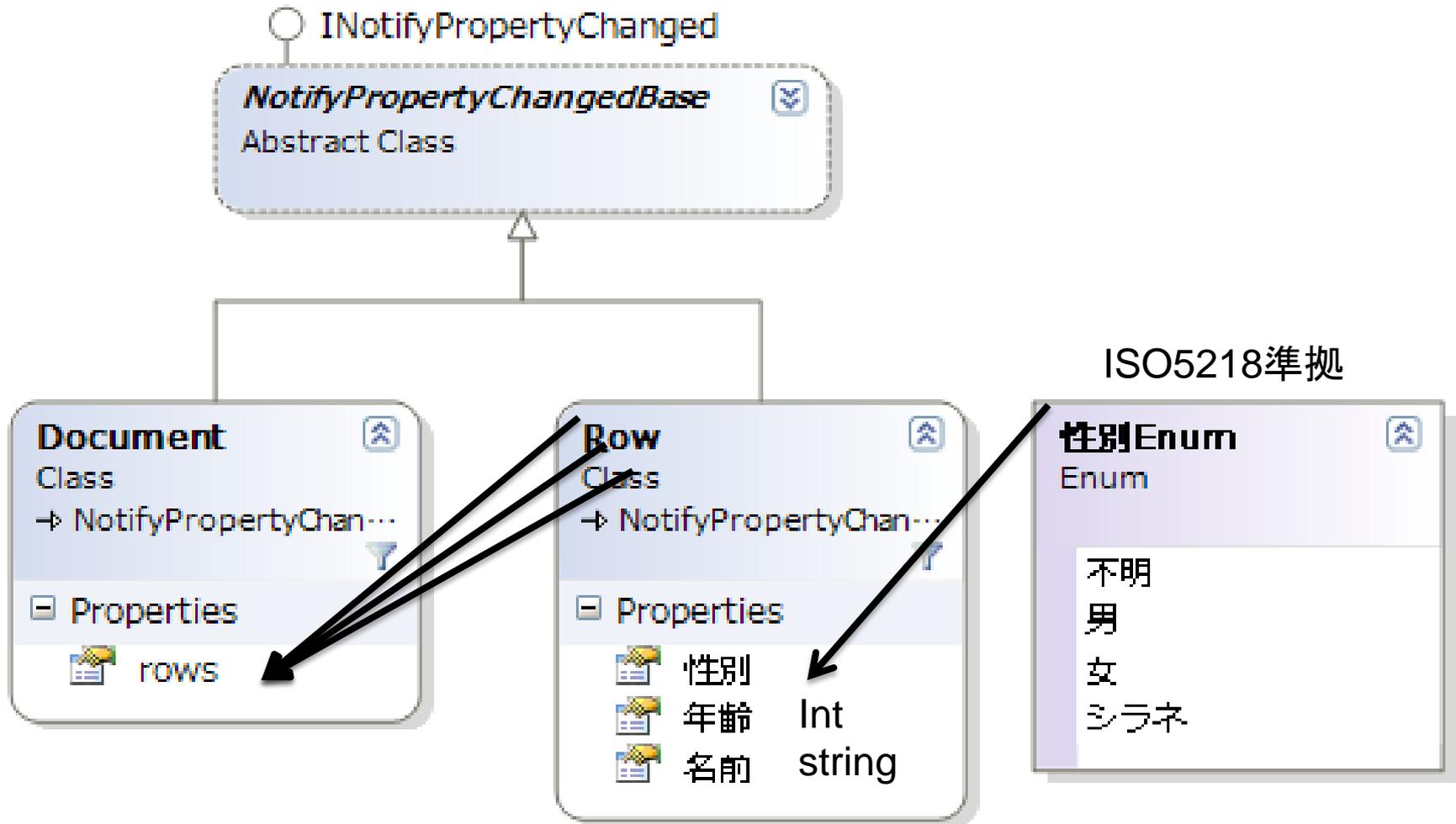
## Listviewとは

- 一覧を表示する
- アイコンモード、詳細モードなどがある
- エクスプローラのアレ
- でもWPFのListviewにはラージアイコンなどはまだ？実装されていない。

## Windows Forms



# DEMOのデータたち(前回と同じ)



## DEMO2のデータたち

```
public class Document : NotifyPropertyChangedBase {  
    public Row[] rows {  
        get { return _rows; }  
        set { _rows = value; }  
    }  
  
    private Row[] _rows = new Row[]  
    {  
        new Row(){名前="なか",年齢 = 32, 性別=性別Enum.男},  
        new Row(){名前="えムナウ",年齢 = 21, 性別=性別Enum.男},  
        new Row(){名前="R田中",年齢 = 18, 性別=性別Enum.シラネ}  
    };  
}
```

# DEMO1

## Listview

- 幅を変更するのもできます。
- セルの順番が勝手に入れ替えられる可能性を考慮してください。  
(というか無関心でいられるかがポイント)

名前	年齢	性別
なか	32	男
えむ	21	男
R田	18	シラネ

クリック!

年齢	性別	名前
32	男	なか
21	男	えむナウ
18	シラネ	R田中

クリック!

ソートくらいつけましょ

DEMO2



わんくま同盟 大阪勉強会 #18

- WPFはコンテンツが置けるところ何でも差し替え可能です。
- しかもこのGridViewにはテンプレート機能が...

# DEMO3

WATCH 1

Name	Value
- [0] _rows	{WpfApplication1.Row[3]}
- [0]	{WpfApplication1.Row}
+ base	{WpfApplication1.Row}
_名前	"なが2"
_年齢	32
_性別	男
名前	"なが2"
年齢	32
性別	男
+ [1]	{WpfApplication1.Row}

## ObservableCollection<T>

- いままでデータソースにはRow[]を使ってました。

```
private Row[] _rows = new[]  
{  
    new Row {名前="なか",年齢 = 32, 性別=性別Enum.男},  
    new Row {名前="えムナウ",年齢 = 21, 性別=性別Enum.男},  
    new Row {名前="R田中",年齢 = 18, 性別=性別Enum.シラネ}  
};
```

でもこれお勧めできません。

ObservableCollection<T>を使いましょう

## ObservableCollection<T>

- ObservableCollection<T>はINotifyCollectionChangedを実装しています。
- INotifyCollectionChangedはINotifyPropertyChangedのコレクション版です。
- データが増えたりすると、通知をあげてくれます。

# DEMO4

- ListView カッコいいでしょ？
- これさえあればDataGridもGridViewもイラネ
- と思っていたのですが、夏ごろに追加されるかもだそうです。
- まだまだWPF関連から目が離せませんね。

Enjoy WPF