**TreeView**

**계층도 (p. 441)**

Control

ItemsControl

HeaderedItemsContorl

MenuItem

ToolBar

**TreeViewItem**

MenuBase (abstract)

ContextMenu

Menu

StatusBar

**TreeView**

* ItemControl은 Selector 클래스의 부모 클래스이다.
* Selector클래스는 ListBox와 ComboBox의 부모 클래스이다.
* ItemControl은 Items라는 중요한 프로퍼티를 갖는다. 이 프로퍼티는 컨트롤에 목록으로 보이는 항목들의 컬렉션이다.
* HeaderedItemsContorl은 ItemControl에 Header라는 이름의 프로퍼티를 추가한 것으로, Header프로퍼티는 Object 타입이지만 주로 텍스트가 사용된다.
* Menu는 탑 레벨 MenuItem 객체 컬렉션이기 때문에 Header 프로퍼티가 없다.
* **TreeView도 탑 레벨 TreeViewItem 객체 컬렉션이기 때문에 Header 프로퍼티가 없다.**

***예제01) ManuallyPopulateTreeView (p.442)***



TreeViewItem itemAnimal = new TreeViewItem();

itemAnimal.Header = "Animal";

tree.Items.Add(itemAnimal);

TreeViewItem itemCat = new TreeViewItem();

itemCat.Header = "Cat";

itemCat.Items.Add("Calico");

TreeViewItem item = new TreeViewItem();

item.Header = "Alley Cat";

itemCat.Items.Add(item);

Button btn = new Button();

btn.Content = "Noodles";

itemCat.Items.Add(btn);

itemCat.Items.Add("Siamese");

itemAnimal.Items.Add(itemCat);

TreeViewItem itemDog = new TreeViewItem();

itemDog.Header = "Dog";

itemDog.Items.Add("Poodle");

itemDog.Items.Add("Irish Setter");

itemDog.Items.Add("German Shepherd");

itemAnimal.Items.Add(itemDog);

***예제02) RecurseDirectorieslnefficiently (p.444)***

**예제02는 TreeView에 디렉토리 트리를 연결하는 잘못된 방법이다.**

* **이유 1 – 디스크 탐색 시간이 오래 걸린다.**
* **이유 2 – 필요 이상으로 많은 것을 만들어낸다.**

**GetSubdirectories(subitem);**

**해결방법 : TreeView에서 필요한 하위 디렉토리만을 트리에 추가**

**lazy loading(lazy binding)기법 사용 : 성능을 위해 객체를 선언만하고 실제로 객체를 생성하거나 로딩하는 것은 필요한 시점으로 미루는 것을 의미**

**TreeViewItem 이벤트**

**Expanded : 사용자가 자식 항목을 보려고 + 기호를 클릭했을떄도 발생하지만,**

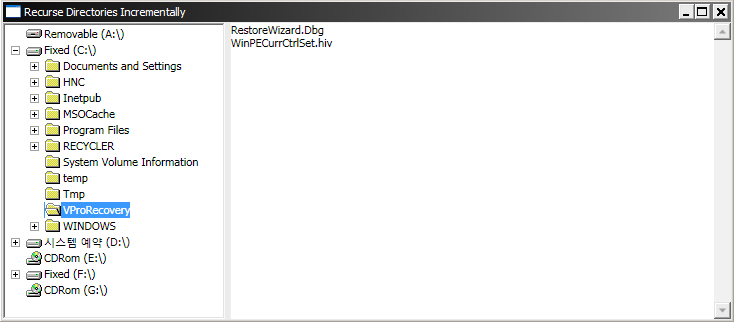
**프로그램에서 임의로 IsExpanded 프로퍼티를 true로 설정했을 때도 발생**

**Collapse : 반대로 자식 항목이 안보이게 될 때 발생**

**Selected & Unselected : 사용자가 항목을 선택했을 때나 IsSelected 프로퍼티가**

**변경됐을 때 발생**

***예제03) RecurseDirectoriesIncrementally (p.447)***

****

**ScrollViewer**

DirectoryTreeViewItem.cs (p.450)

폴더 항목들을 값과 연동시켜주는 메소드

public void Populate()

{

Items.Clear();

DirectoryInfo[] dirs;

try

{

dirs = dir.GetDirectories();

}

catch

{

return;

}

foreach (DirectoryInfo dirChild in dirs)

Items.Add(new DirectoryTreeViewItem(dirChild));

}

DirectoryTreeView.cs (p.452)

드라이브를 설정하는 메소드

public void RefreshTree()

{

BeginInit();

Items.Clear();

// Obtain the disk drives.

DriveInfo[] drives = DriveInfo.GetDrives();

foreach (DriveInfo drive in drives)

{

char chDrive = drive.Name.ToUpper()[0];

DirectoryTreeViewItem item =

new DirectoryTreeViewItem(drive.RootDirectory);

**… 드라이브 상태 및 드라이브에 맞는 비트맵 설정 코드 …**

}

EndInit();

}

***예제04) TemplateTheTree (p.457)***

****

**ItemTemplate 프로퍼티**

ItemsControl에 의해 정의된 **ItemTemplate** 프로퍼티는 TreeView를 상속 받는다.

더욱이 **HierarchicalDataTemplate** 클래스는 DataTemplate을 상속받았기 때문에

TreeView가 만들어야 하는 계층적 데이터를 표현하는 가장 이상적인 클래스이다. **( p.456)**

// Create TreeView and set as content of window.

TreeView treevue = new TreeView();

Content = treevue;

// Create HierarchicalDataTemplate based on DiskDirectory.

HierarchicalDataTemplate template =

new HierarchicalDataTemplate(typeof(DiskDirectory));

// Set Subdirectories property as ItemsSource.

template.ItemsSource = new Binding("Subdirectories");

// Create FrameworkElementFactory for TextBlock.

FrameworkElementFactory factoryTextBlock =

new FrameworkElementFactory(typeof(TextBlock));

// Bind Text property with Name property from DiskDirectory.

factoryTextBlock.SetBinding(TextBlock.TextProperty, new Binding("Name"));

// Set this Textblock as the VisualTree of the template.

template.VisualTree = factoryTextBlock;

// Create a DiskDirectory object for the system drive.

DiskDirectory dir = new DiskDirectory(new DirectoryInfo(

Path.GetPathRoot(Environment.SystemDirectory)));

// Create a root TreeViewItem and set its properties.

TreeViewItem item = new TreeViewItem();

item.Header = dir.Name;

item.ItemsSource = dir.Subdirectories;

item.ItemTemplate = template;

// Add TreeViewItem to TreeView.

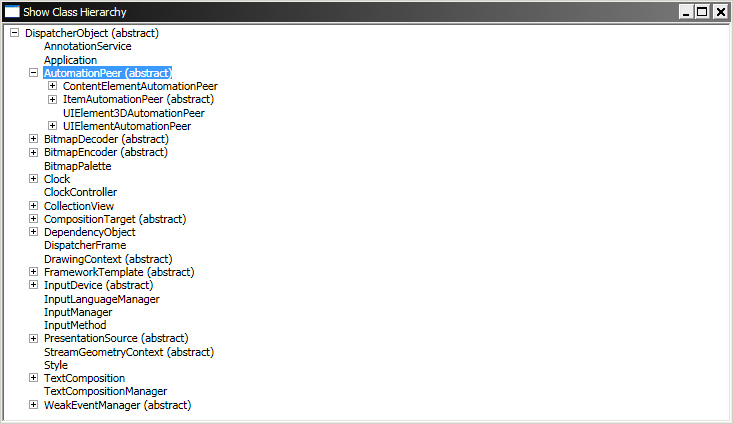
treevue.Items.Add(item);

item.IsExpanded = true;

**(p.459)**

**TemplateTheTree.cs**

***예제05) ShowClassHierarchy (p.461)***

******

**DispatcherObject를 상속받은 WPF의 모든 public 클래스를 트리로 보여주는 예제**