Windows Explorer-Like Program: extends ListView

```
using System;
using System.Diagnostics; // For Process.Start
using System.Drawing;
using System.IO;
using System.Windows.Forms;

class FileListView : ListView {
string strDirectory;

public FileListView() {
View = View.Details;

ImageList imglst = new ImageList();
imglst.Images.Add(new Bitmap(GetType(), “DOC.BMP”));
imglst.Images.Add(new Bitmap(GetType(), “EXE.BMP”));

SmallImageList = imglst;
LargeImageList = imglst;

Columns.Add(“Name”, 100, HorizontalAlignment.Left);
Columns.Add(“Size”, 100, HorizontalAlignment.Right);
Columns.Add(“Modified”, 100, HorizontalAlignment.Left);
Columns.Add(“Attribute”, 100, HorizontalAlignment.Left);
}
public void ShowFiles(string strDirectory) {
this.strDirectory = strDirectory;

Items.Clear();
DirectoryInfo dirinfo = new DirectoryInfo(strDirectory);
FileInfo[] afileinfo;

try {
afileinfo = dirinfo.GetFiles();
} catch {
```

```
return;
}

foreach (FileInfo fi in afileinfo) {
ListViewItem lvi = new ListViewItem(fi.Name);

if (Path.GetExtension(fi.Name).ToUpper() == “.EXE”)
lvi.ImageIndex = 1;
else
lvi.ImageIndex = 0;

lvi.SubItems.Add(fi.Length.ToString(“N0”));
lvi.SubItems.Add(fi.LastWriteTime.ToString());

string strAttr = “”;

if ((fi.Attributes & FileAttributes.Archive) != 0)
strAttr += “A”;

if ((fi.Attributes & FileAttributes.Hidden) != 0)
strAttr += “H”;

if ((fi.Attributes & FileAttributes.ReadOnly) != 0)
strAttr += “R”;

if ((fi.Attributes & FileAttributes.System) != 0)
strAttr += “S”;

lvi.SubItems.Add(strAttr);

Items.Add(lvi);
}
}
protected override void OnItemActivate(EventArgs ea) {
base.OnItemActivate(ea);

foreach (ListViewItem lvi in SelectedItems) {
try {
Process.Start(Path.Combine(strDirectory, lvi.Text));
} catch {
```

```
continue;
}
}
}
}

class ExplorerLike : Form {
FileListView filelist;
MenuItemView mivChecked;

public static void Main() {
Application.Run(new ExplorerLike());
}
public ExplorerLike() {
BackColor = SystemColors.Window;
ForeColor = SystemColors.WindowText;

filelist = new FileListView();
filelist.Parent = this;
filelist.Dock = DockStyle.Fill;

Splitter split = new Splitter();
split.Parent = this;
split.Dock = DockStyle.Left;
split.BackColor = SystemColors.Control;

Menu = new MainMenu();
Menu.MenuItems.Add("&View");

string[] astrView = { "Lar&ge Icons", "S&mall Icons",
"&List", "&Details" };
View[] aview = { View.LargeIcon, View.SmallIcon,
View.List, View.Details };
EventHandler eh = new EventHandler(MenuOnView);

for (int i = 0; i < 4; i++) { MenuItemView miv = new
MenuItemView(); miv.Text = astrView[i]; miv.View = aview[i];
miv.RadioCheck = true; miv.Click += eh; if (i == 3) // Default
== View.Details { mivChecked = miv; mivChecked.Checked = true;
```

```
filelist.View = mivChecked.View; } Menu.MenuItems[0].MenuItems.Add(miv); } Menu.MenuItems[0].MenuItems.Add("-"); MenuItem mi = new MenuItem("&Refresh", new EventHandler(MenuOnRefresh), Shortcut.F5); Menu.MenuItems[0].MenuItems.Add(mi); } void DirectoryTreeViewOnAfterSelect(object obj, TreeViewEventArgs tvea) { filelist.ShowFiles(tvea.Node.FullPath); } void MenuOnView(object obj, EventArgs ea) { mivChecked.Checked = false; mivChecked = (MenuItemView)obj; mivChecked.Checked = true; filelist.View = mivChecked.View; } void MenuOnRefresh(object obj, EventArgs ea) { } } class MenuItemView : MenuItem { public View View; } [/csharp]
```