C#中实现文件拖放打开的方法

```
  private void Form1_DragEnter(object sender, System.Windows.Forms.DragEventArgs e)
    {
    if (e.Data.GetDataPresent(DataFormats.FileDrop))
    e.Effect = DragDropEffects.Link;
    else e.Effect = DragDropEffects.None;
    }

    private void Form1_DragDrop(object sender, System.Windows.Forms.DragEventArgs e)
    {
    //其中 label1.Text 显示的就是拖进文件的文件名;
    label1.Text = ((System.Array)e.Data.GetData(DataFormats.FileDrop)).GetValue(0).ToString();
    }
```

注：窗体的 AllowDrop=True;

必须处理好三种事件："ItemDrag"、"DragEnter"、"DragDrop"。其中只有第一种事件是在源组件中触发的，另外二种事件是在目标组件中触发的。其中当用户拖动组件触发"ItemDrag"事件；当拖动数据进入目标组件区域触发"DragEnter"事件；当用户在目标组件区域放置拖动的数据触发"DragDrop"事件。下面就根据拖放操作的操作顺序来详细介绍：

（1）.开始"拖"（Drag）操作：
通过"DoDragDrop"方法拉开了拖放操作的第一步。"DoDragDrop"方法的语法为： DoDragDrop (object data , DragDropEffects allowedEffects) ;

其中第二个参数来是说明此次拖放操作最后所要实现的效果，因为拖放操作有时实现的效果是把源组件中的内容"拖"到目标组件中，这种效果就是 "Move"；有时拖放的效果是在目标组件中加入拖动的数据，对源组件的内容是没有什么影响的，这种效果就是"Copy"。当然无论是"Move"还是 "Copy"，这都要通过具体的编程来实现，设定这些效果只是告诉操作系统，你进行拖放操作的类型，从而为拖放操作设定特定的图标。此例中实现开始"拖放 "操作的具体实现代码如下：

```
private void treeView1_ItemDrag ( object sender , ItemDragEventArgs e )
{
string strItem = e.Item.ToString ( ) ;
//开始进行"Drag"操作
DoDragDrop ( strItem , DragDropEffects.Copy | DragDropEffects.Move ) ;
}
```

在上面代码中，我们定义的拖放数据类型是字符串，其实拖放的数据类型可以是很多种的，你可以通过修改"DoDragDrop"方法的第一个参数来设定你所要拖放数据类型，譬如：位图或者其他什么。

（2）.目标组件允许进行拖放操作：

既然你已经开始进行拖放操作，你还必须告诉你要拖放到的目标组件，要接受你所拖放的数据，"DragEnter"事件正好可以处理。在下列的代码中，我们是通过判断拖放数据类型来确定是否接受拖放，如果是字符串，则可以，否则，则不行。具体代码如下：

```
private void listView1_DragEnter ( object sender , DragEventArgs e )
{
//判断是否目前拖动的数据是字符串，如果是，则拖动符串对目的组件进行拷贝
```

```
if ( e.Data.GetDataPresent ( DataFormats.Text ) )
e.Effect = DragDropEffects.Move ;
else
e.Effect = DragDropEffects.None ;
}
```

（3）.获得拖放的字符串，在目标组件中加入相应的内容：

此步的处理过程是十分明确的，要分成二步来进行，首先要得到拖放的字符串，其次是在目标组件中加入以此字符串为标题的项目。当然还要在相应的位置了。下面就是实现这二步操作的具体代码：

```
    private void listView1_DragDrop ( object sender , DragEventArgs e )
{
string dummy = "temp" ;
//获得进行"Drag"操作中拖动的字符串
string s = ( string ) e.Data.GetData ( dummy.GetType ( ) ) ;
s = s.Substring ( s.IndexOf ( ":" ) + 1 ).Trim ( ) ;
Position.X = e.X ;
Position.Y = e.Y ;
Position = listView1.PointToClient ( Position ) ;
//在目标组件中加入以此字符串为标题的项目
listView1.Items.Add ( new ListViewItem ( s , 0 ) ) ;
}
```

此致通过对这三个事件的编程，已经完成了由 TreeView 组件到 ListView 组件的拖放操作。