**插件开发学习笔记**

-----蔡观洋

-----start:2013.06.18

**目 录**

[1 Eclipse平台扩展点介绍 1](#_Toc359507998)

[1.1 透视图扩展点 1](#_Toc359507999)

[1.1.1 创建自定义透视图 1](#_Toc359508000)

[1.1.2 扩展现有的透视图 1](#_Toc359508001)

[1.2 视图扩展点 2](#_Toc359508002)

[1.2.1 创建自定义视图 2](#_Toc359508003)

[1.2.2 视图间的通信 3](#_Toc359508004)

[1.3 编辑器扩展点 4](#_Toc359508005)

[1.3.1 单页面编辑器 4](#_Toc359508006)

[1.3.2 多页面编辑器 6](#_Toc359508007)

[1.4 扩展当前平台功能 7](#_Toc359508008)

[1.4.1 首选项修改 7](#_Toc359508009)

[1.4.2 平台视图中上下文菜单，菜单项的修改 7](#_Toc359508010)

[2 JFace使用 7](#_Toc359508011)

[2.1 向导对话框 7](#_Toc359508012)

[附 录 7](#_Toc359508013)

[1. Eclipse平台内部视图ID 7](#_Toc359508014)

[2. 参考网址 7](#_Toc359508015)

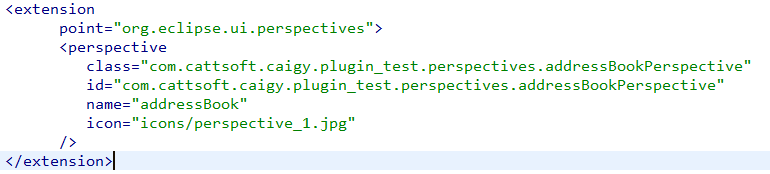
1 Eclipse平台扩展点介绍

1.1 透视图扩展点

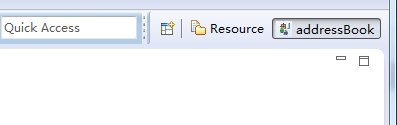
1.1.1 创建自定义透视图

扩展点：org.eclipse.ui.perspectives

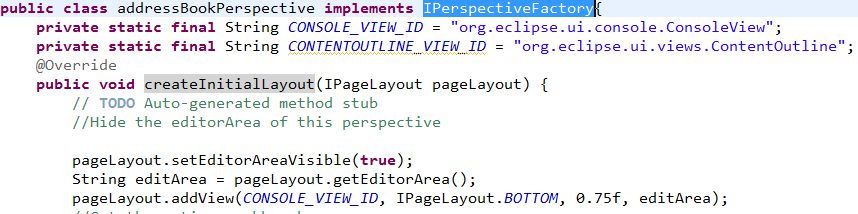
在plugin.xml.中配置如下：



运行效果如下：



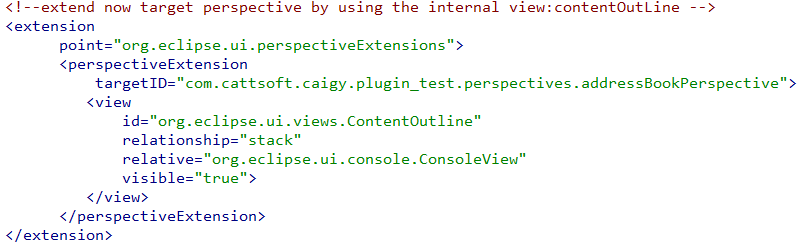
开发过程中用户需要编写一个实现接口IPerspectiveFactory的类，设置此透视图的页面布局。如下：



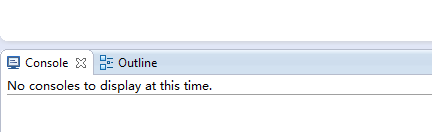
上述代码调用了IPageLayout接口的方法向透视图中添加了Eclipse平台内部的控制台视图，也可以添加用户自定义的视图，

1.1.2 扩展现有的透视图

Plugin.xml配置信息如下：



上述配置将大纲视图放在了Console视图的右边，更改的是上节刚创建的透视图。效果如下：



注意：在透视图中添加视图的方式有两种，一种如1.1.1所示，在透视图初始化类中添加代码；另一种如1.1.2节所描述的通过扩展点添加视图。

另外也可以在window—show view 添加视图的快捷方式，参考Eclipse插件开发学习笔记中的P347.

添加现有透视图时需要知道平台内部视图的ID，可以参考附录中提供的ID号。

1.2 视图扩展点

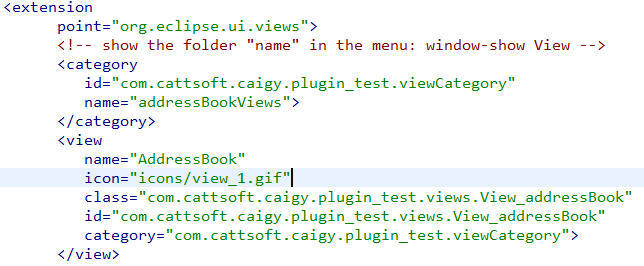
1.2.1 创建自定义视图

Eclipse中的Console，工程浏览等都是一个视图，

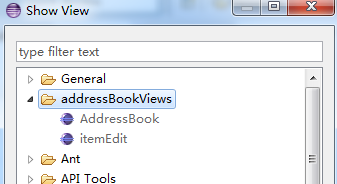
在window—> show view中可以看到平台提供的视图

扩展点为：org.eclipse.ui.views

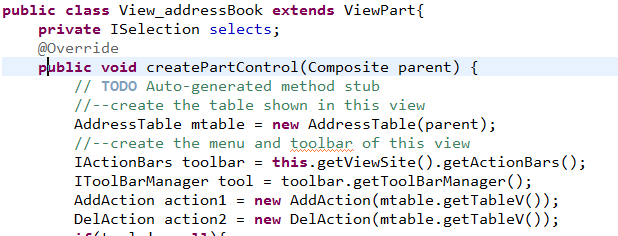
在Plugin.xml的配置为：



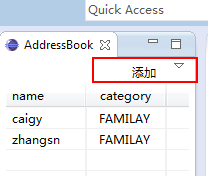
其中category用以生成在“show view”窗口中显示的视图类别目录：



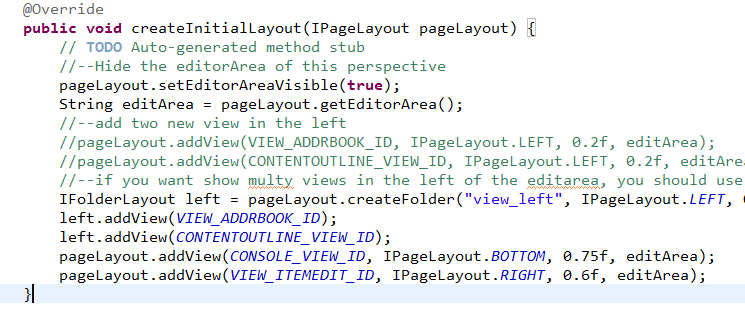
节点view声明一个视图，category标示其所在的目录，class标示此视图对应的类。此类需要实现IViewPart接口，开发中大多直接继承ViewPart类即可，实现两个方法：



方法createPartControl用以初始化视图界面的控件，由于视图一般用作导航，展示功能，大多时候使用JFace中的TableViewer，ListViewer，TreeViewer即可。这些控件基于MVC思想，使用方便。效果如下：



视图出现的位置则需要用户指定，可以在透视图类中加以说明。



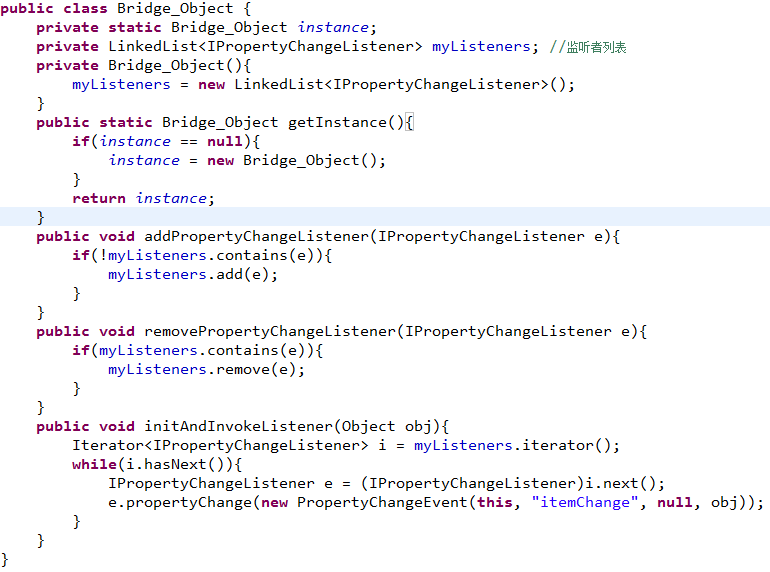
可以使用Jface中的菜单管理器为视图添加自己的上下文菜单，工具栏菜单等。如上图红色框圈到的区域。

1.2.2 视图间的通信

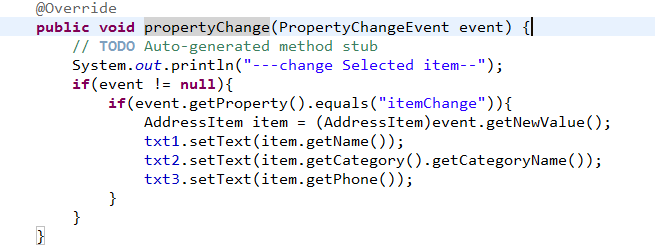
我们可以使用SWT或JFace来定制视图的界面，但是如何让两个或多个视图之间传递数据？

1. 属性改变监听器模式：它主要通过使用JFace中一个重要的接口org.eclipse.jface.util.IPropertyChangeListener来实现，通过注册该监听器，以及使用IPropertyChangeEvent，可以达到事件传递的目的。采用“观察者模式”来实现此方法。

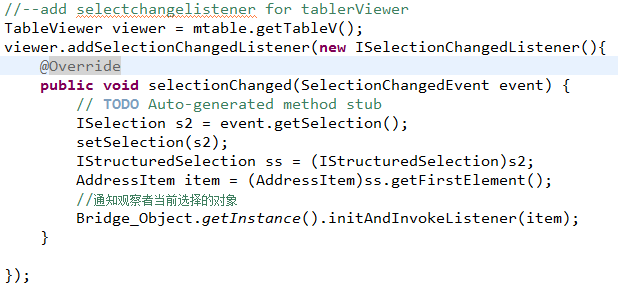
1）首先，创建了一个类Bridge\_Object来作为实体，它提供了基本的添加删除观察者，通知观察者方法，由于它需要保存观察者，且需要再多个视图类中使用，采用“单例模式”控制实例的个数。如下：



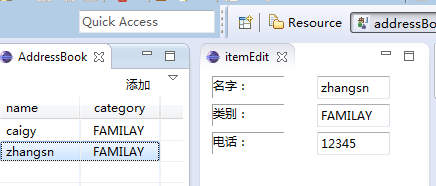
2）其次定义了一个视图B，它实现了接口IPropertyChangeListener，作为观察者，监听视图A的变化情况。需要实现方法propertyChange，实体调用此方法来更新视图B的界面。



3）假设当我们选中视图A中表格的行是，将选中的行传递到视图B中，此时我们可以在视图A中的表格添加一个监听器，而后调用实体类Bridge\_Object中的initAndInvoke方法通知观察者做出变化。



运行效果如下图：



1. 选择提供器和选择监听器模式，有点复杂，参考网址[2] ---后期补充

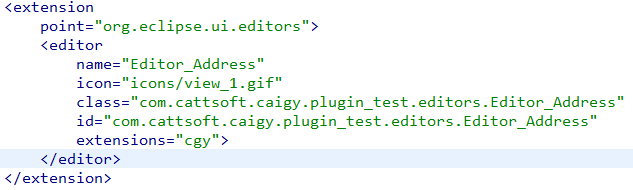
1.3 编辑器扩展点

在Eclipse中，如在资源视图中，双击.java类型的文件，则会调用系统内部绑定好的Java源文件编辑器，用户使用它编辑文件。我们可以创建自定义编辑器， Eclipse平台可以根据文件后缀名自动打开相应的编辑器。

1.3.1 单页面编辑器

1）Plugin.xml配置编辑器扩展点：

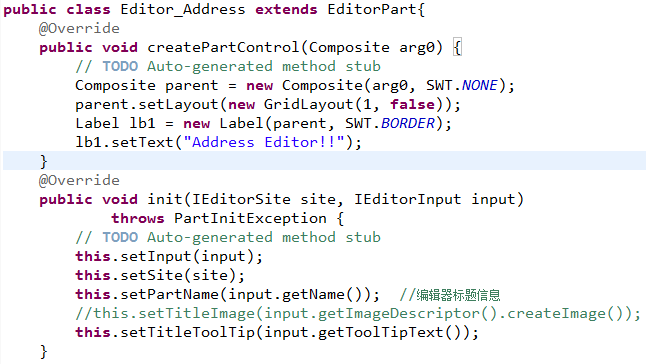
扩展点为：org.eclipse.ui.editors



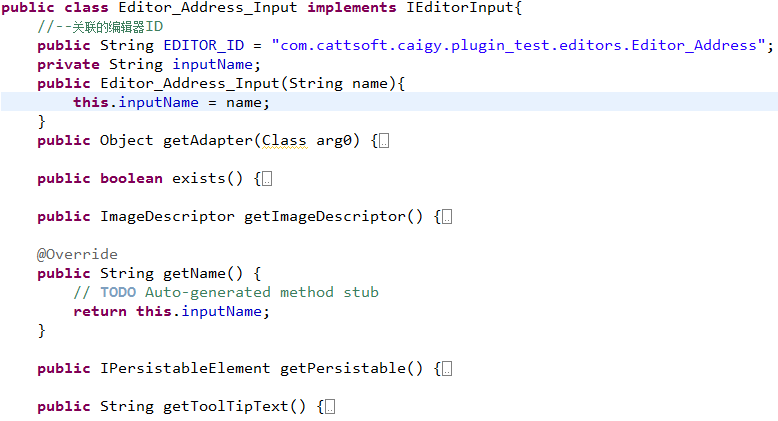
说明：其中extensions指定了此编辑器可以接受的文件类型，多个类型间用逗号隔开，注意后缀名字前不带点。

2）创建编辑器类

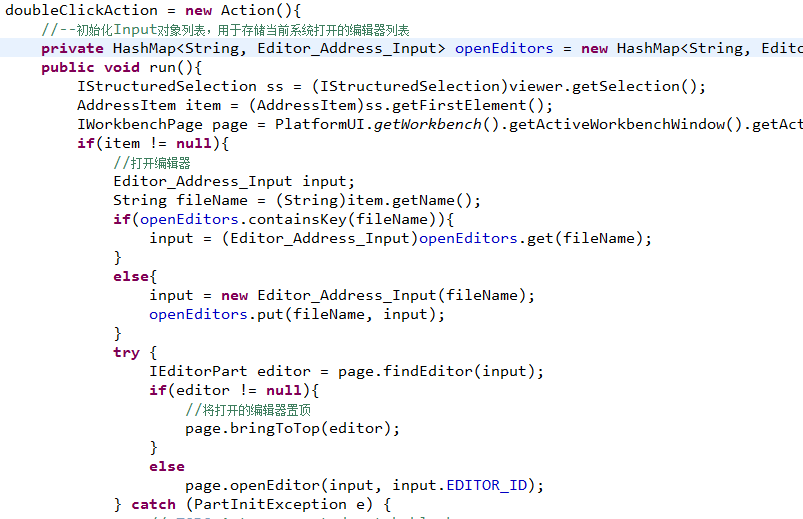
A. EditorPart类，设计编辑器的页面，



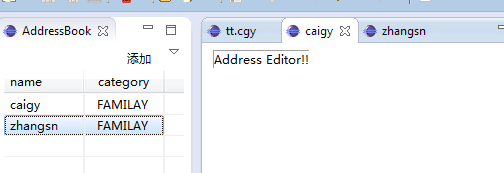
B．EditorInput类，设置编辑器的标题，图标，提示等信息，因为一个编辑器可以对应多个EditorInput实例。



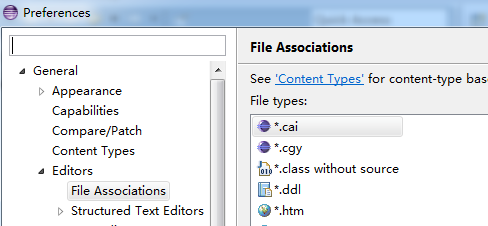
C. 关联编辑器和编辑器源，我们可以在视图中添加一个双击事件，选中一项时就打开一个视图；



运行效果如下图：

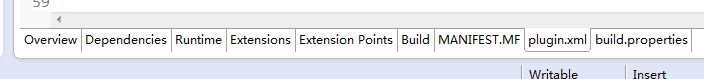


另外我们自定义的编辑器，支持后缀为.cgy的文件，可以在window->Preferences中看到



1.3.2 多页面编辑器（后期补充）

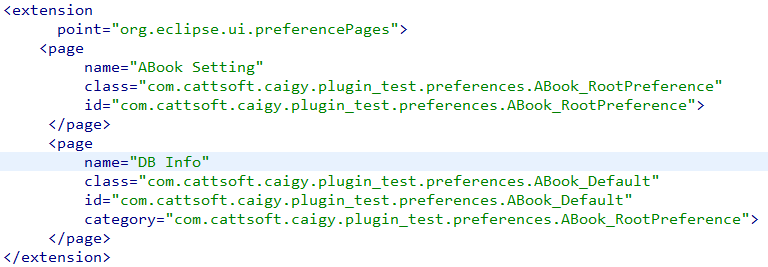
Eclipse提供的支持多页面的类有MultiEditor和MultiPageEditorPart，FormEditor，具体用法参考《Eclipse插件开发学习笔记》。实现效果如双击插件工程中的Plugin.xml文件打开的编辑器一样（因为在扩展点属性中filenames设置了固定的文件，所以没有打开内置的xml编辑器）。



1.4 扩展当前平台功能

1.4.1 首选项修改

1）plugin.xml中添加扩展点：



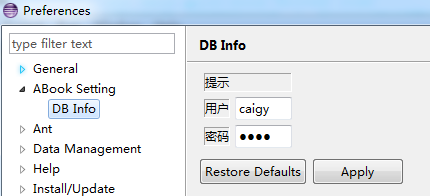
说明：category用以标示首选项页面在页面树中的父节点

2）编辑首选项页面类，他需要继承PreferencePage类创建页面的组件，实现接口IWorkbenchPreferencePage进行初始化。

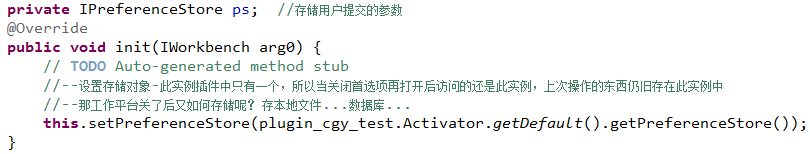


其中用户可以覆盖父类中的按钮方法（恢复，确定，应用，取消..），如performDefaults和performApply等自定义行为。

运行效果如下：



另外用户可以采用JFace中的接口IPreferenceStore来存储用户提交的变量。不过它只能在平台此次运行中起作用，重启Eclipse后，此数据丢失，如果需要长久保存的话，可以存储到配置文件，启动插件时重新读取此文件即可。



1.4.2 平台视图中上下文菜单，菜单项的修改

2 JFace使用

2.1 向导对话框

2.2 首选项

2.3 表格，列表，树（可动态编辑内容）

3 SWT使用

3.1 布局管理器

附 录

1. Eclipse平台内部视图ID

Console: org.eclipse.ui.console.ConsoleView

大纲视图ContentOutline: org.eclipse.ui.views.ContentOutline

属性视图：org.eclipse.ui.views.PropertySheet

任务视图： org.eclipse.ui.views.BookmarkNavigator

书签视图：org.eclipse.ui.views.TaskList

2. 参考网址

[1] <http://www.eclipse.org/articles/viewArticle/ViewArticle2.html> --(视图创建帮助文档)

[2] <http://www.cnblogs.com/zephyr/archive/2008/05/30/1210960.html> --(视图间的通信)