

通过设计器和模拟器快速了解 **Fire Workflow**

作者：非也 QQ: 20674450 Email: nychen2000@163.com

目 录

| | |
|-----------------------------------|----|
| 1. Fire Workflow 的构成..... | 2 |
| 2. 设计器的安装..... | 2 |
| 1) NetBeans 设计器插件安装..... | 2 |
| 2) Eclipse 设计器插件安装..... | 4 |
| 3. 设计器各部分介绍..... | 4 |
| 1) 设计器界面..... | 5 |
| 2) 模拟器界面..... | 6 |
| 3) Xml 源代码界面..... | 7 |
| 4. 设计并模拟运行一个最简单的流程..... | 7 |
| 1) 创建一个流程定义文件..... | 8 |
| 2) 给新流程增加相关的 Activity 和 Task..... | 9 |
| 3) 模拟流程的执行..... | 13 |
| a. 创建新的流程实例..... | 13 |
| b. 签收工单..... | 15 |
| c. 完成工单..... | 15 |
| d. 签收并完成审批环节相关的工单，流程实例终止..... | 16 |

1. Fire Workflow 的构成

Fire Workflow 由模型、引擎、设计器（包含模拟器）三部分组成。

模型部分规定了流程定义文件的各种元素及其相互关系，例如流程 (WorkflowProcess)、活动 (Activity)、转移 (Transition)、开始节点 (StartNode)、结束节点 (EndNode)、同步器 (Synchronizer)。模型部分的实现在 org-fireflow-model.jar 中。

引擎读取流程定义文件并解释执行。引擎提供一组对象和相关的 API 供外部系统调用，如流程实例 (ProcessInstance)、任务实例 (TaskInstance)、工单 (WorkItem)、事件等等。引擎部分的实现在 org-fireflow-engine.jar 中。

设计器编辑并输出流程定义文件。Fire Workflow 的设计器附带了强大的模拟器，可以在设计时模拟流程的执行，从而检查流程定义的正确性。

此处附带解释一下我的一个观点：我认为，流程定义文件和 java 文件一样，是应用系统源代码的一部分。因此，流程设计器做成了当前流行的 IDE 的插件，便于开发人员进行流程开发。而且每个流程单独一个定义文件，就像每个 java 类在通常情况单独一个文件一样。

2. 设计器的安装

1) NetBeans 设计器插件安装

注：截止 2009-01-26 日，NetBeans 尚有不少 bug 没有修正。:(

1、首先将 FireflowDesigner_Plugin_for_Netbeans.zip 解压缩，到一个临时目录。然后打开 netbeans 的插件管理器(Tools->Plugins)，选择"Downloaded"Tab 页面，如下图 2.1-1：

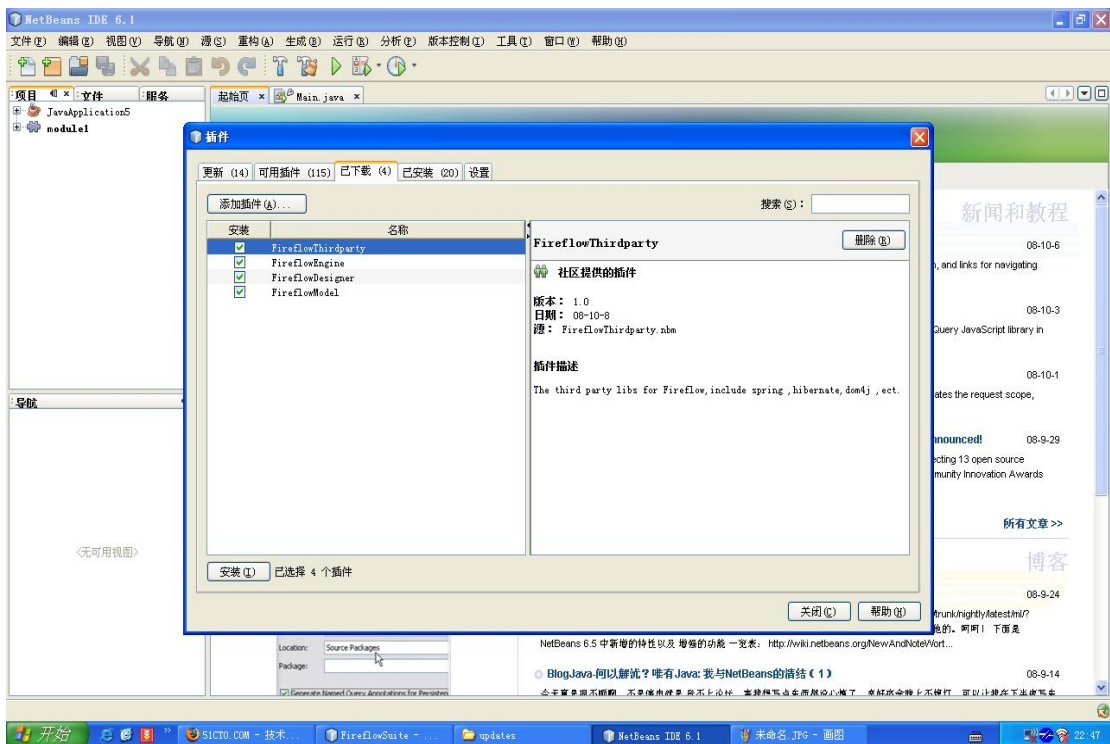


图 2.1-1

2、在该页面中，通过按钮 “Add Plugins...”（添加插件按钮）将上一步解压的.nbm 文件 load 进来，如上图 2.1-1。

3、在上图 2.1-1 界面中点击 “install”（安装按钮），将 3 个 nbm 安装到 netbeans 中。

4、安装完毕后，重新启动 netbeans，可以在已安装的插件列表中看到 Fireflow 插件。

如图 2.1-2

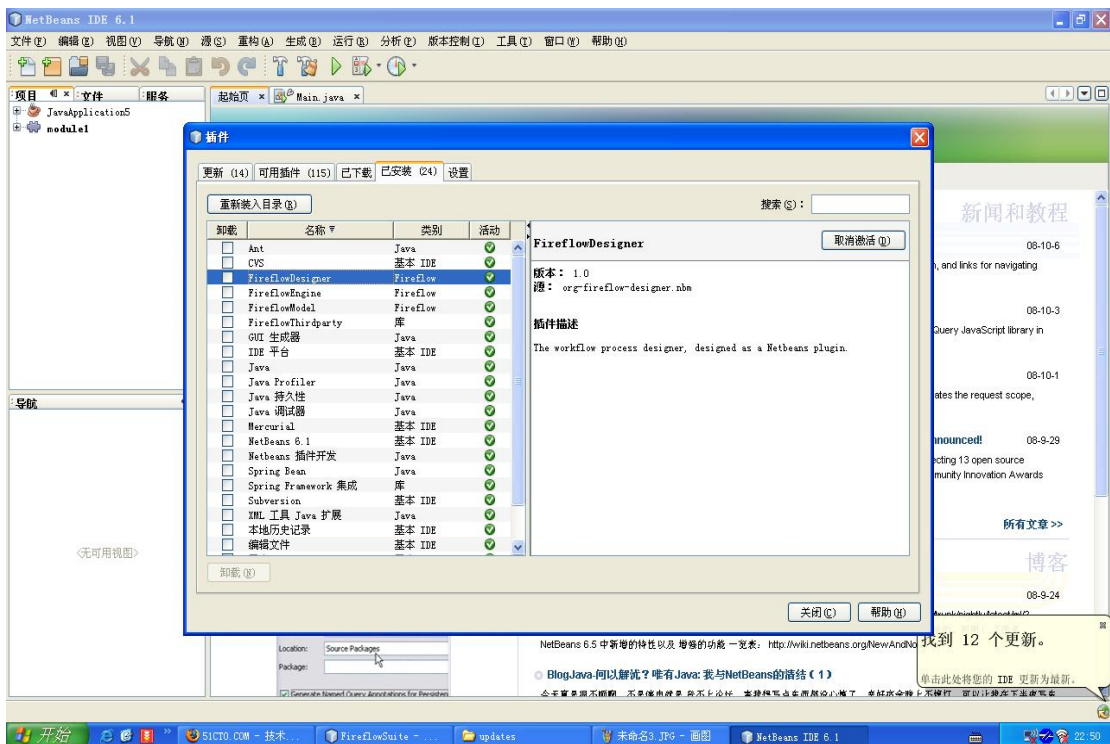
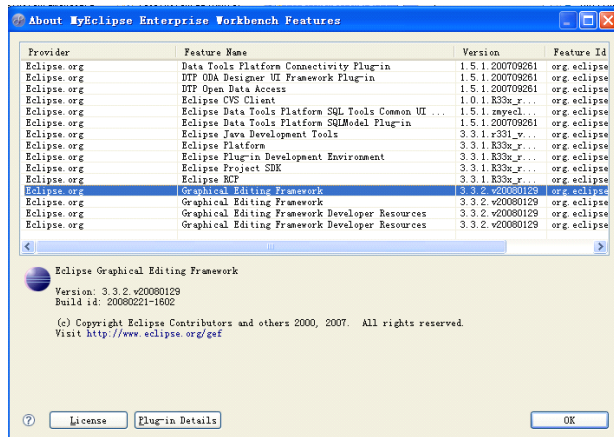


图 2.1-2

2) Eclipse 设计器插件安装

1、Eclipse 设计器基于 GEF (Graphic Edite Framework)，因此确保你的 eclipse IDE 已经安装了 GEF 3.3.2 或者以上版本。



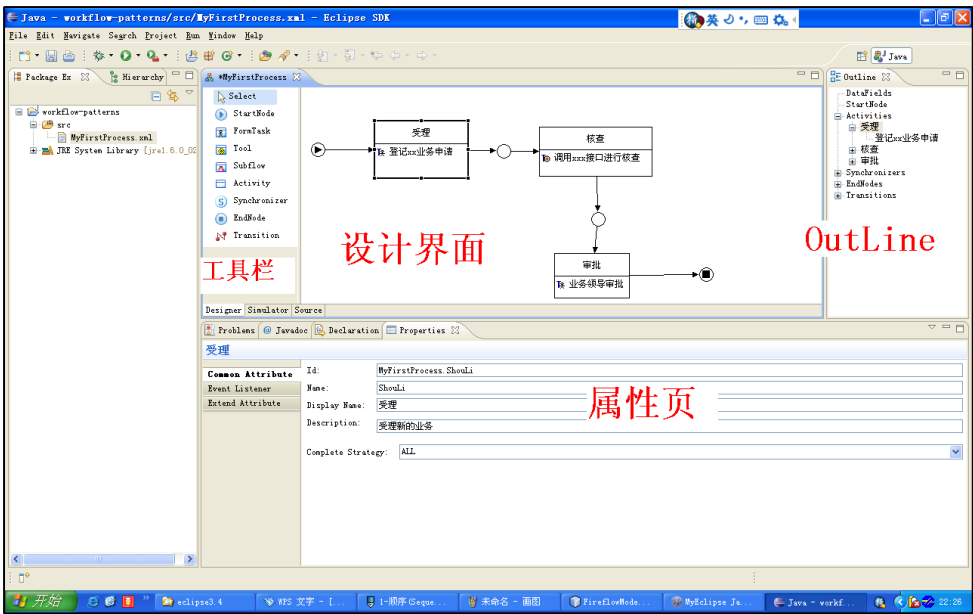
2、将流程设计器插件包 FireflowDesigner4Eclipse_x.x.x.jar 拷贝到 Eclipse 的 Plugins 目录下。

3. 设计器各部分介绍

此处以 Eclipse 插件为例，介绍设计器的组成部分，NetBeans 插件的界面布局大同小异，不赘述。

Fire Workflow 设计器是一个多页面编辑器，包含了设计器页面、模拟器页面和源代码页面。


1) 计器界面



如上图，设计器主要由如下几部分构成：1、图形化的设计界面，2、工具栏，3、Outline 页面，4、属性页。

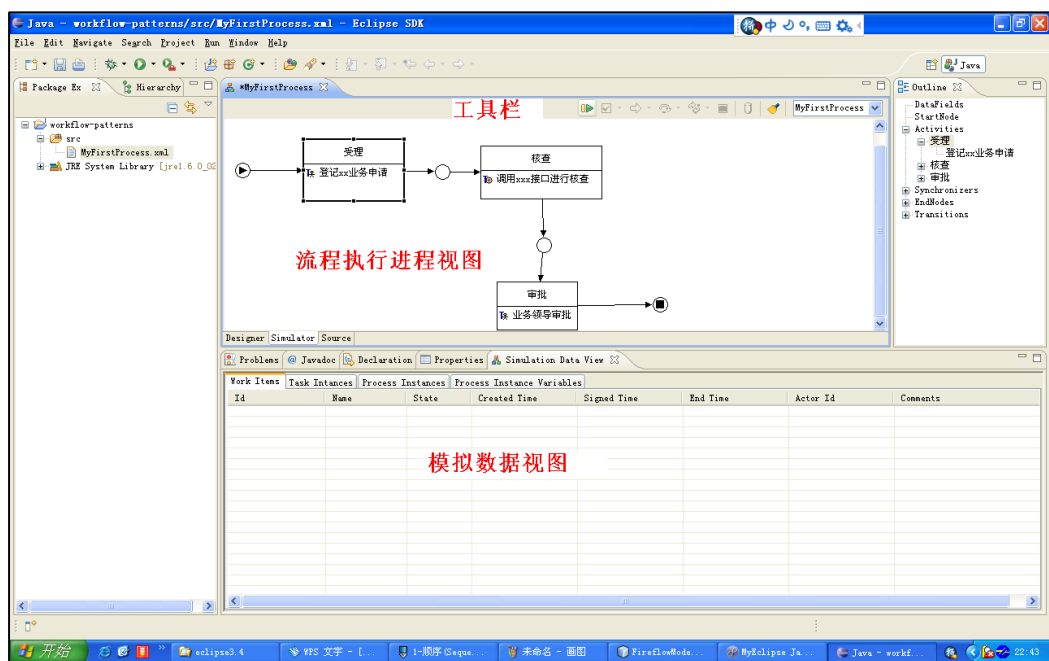
工具栏上各 工具按钮的作用如下：

| 工具栏按钮 | 含义 |
|--|--|
|  Select | 使得设计界面的鼠标处于“select”状态 |
|  StartNode | 创建一个开始节点，每个流程有且仅有一个开始节点 |
|  FormTask | 创建一个新的环节，该环节自动包含一个表单类型的任务 (Form Task) |
|  Tool | 创建一个新的环节，该环节自动包含一个 Tool 类型任务。Tool 类型的任务用于调用 java 程序。 |
|  Subflow | 创建一个新的环节，该环节自动包含一个 Subflow 类型的任务。 |
|  Activity | 创建一个空的环节，该环节不包含任何任务。当然，可以在任何环节中追加任意数量的任务。 |
|  Synchronizer | 创建一个同步器。Fire workflow 是基于 Petri Net 的一个工作流系统，同步器节点相当于 Petri Net 中的 Place，任何两个环节直接都必须有一个同步器节点。 开始节点和结束节点是两类特殊的同步器。 |
|  EndNode | 创建一个结束节点，一个流程可以有任意多个结束节点。 |

| | |
|--|-----------------|
|  Transition | 创建节点间的“转移”，即连接线 |
|--|-----------------|

2) 模拟器界面

点击编辑区的“Simulator”Tab 页，切换到模拟器界面。模拟器包含三部分：1、位于上方的工具栏，2、图形化的流程执行进程视图，3、模拟数据视图，如果该视图没有打开，则通过“window-->show view-->other...-->Fire workflow-->Simulation Data View”打开。如下图



模拟数据视图由 4 个 Tab 页面组成，分别是：




Work Items：展示当前的工单信息，该表格展示的信息正是工单表 T_FF_RT_WORKITEM 中的数据。







Task Instances：展示当前的任务实例信息，该表格展示的信息正是任务实例表 T_FF_RT_TASKINSTANCE 中的数据。

Process Instances：展示当前的流程实例信息，该表格展示的信息正是流程实例表 T_FF_RT_PROCESSINSTANCE 中的数据。

Process Instance Variables：展示当前的流程实例变量信息，该表格展示的信息正是实例变量表 T_FF_RT_PROCINST_VAR 中的数据。

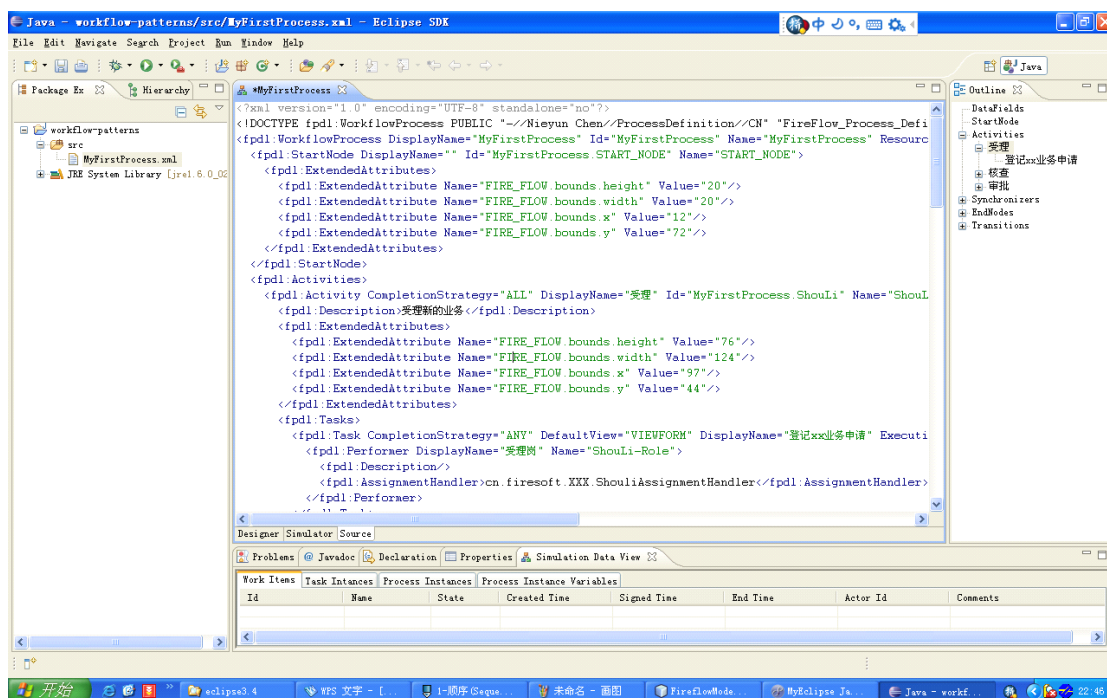
模拟器工具栏含义如下表

| | |
|---|----------------------------|
| 按钮图标 | 含义 |
|  | 创建一个新的流程实例 |
|  | 签收当前工单 |
|  | 完成当前工单，并且按照预定义的流程逻辑启动下一个环节 |

| | |
|---|--|
| | 及其任务 |
|  | 完成当前工单，并且跳转到指定的环节及其任务，即“自由流程”。 |
|  | 完成当前工单，并且跳转到已经完成的某个环节及任务，重新启动之，即“循环”。 |
|  | 终止当前流程实例 |
|  | 设置当前流程实例的流程变量。在 Fire workflow 中，流程变量并不需要预先定义 DataField，可以在任何时候设置任意数量的流程变量。 |
|  | 清空所有的模拟数据。 |
|  | 流程执行进程视图中当前流程的名称，在有子流程的情况下，该下拉列表用于在视图中切换不同的流程。 |

3) Xml 源代码界面

点击编辑区的“Source”Tab 按钮，打开源代码编辑页，如下图。源代码编辑页只能查看 xml 格式的流程定义文件，不能编辑。



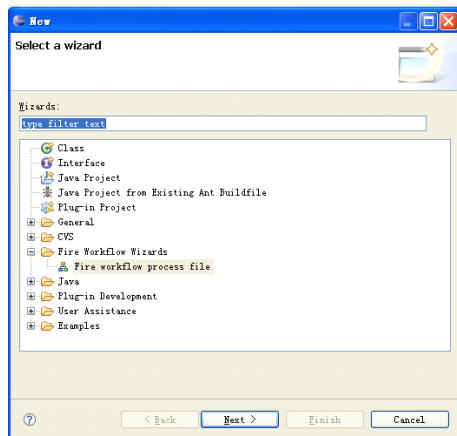
4. 设计并模拟运行一个最简单的流程

此处以 Eclipse 插件为例进行介绍。

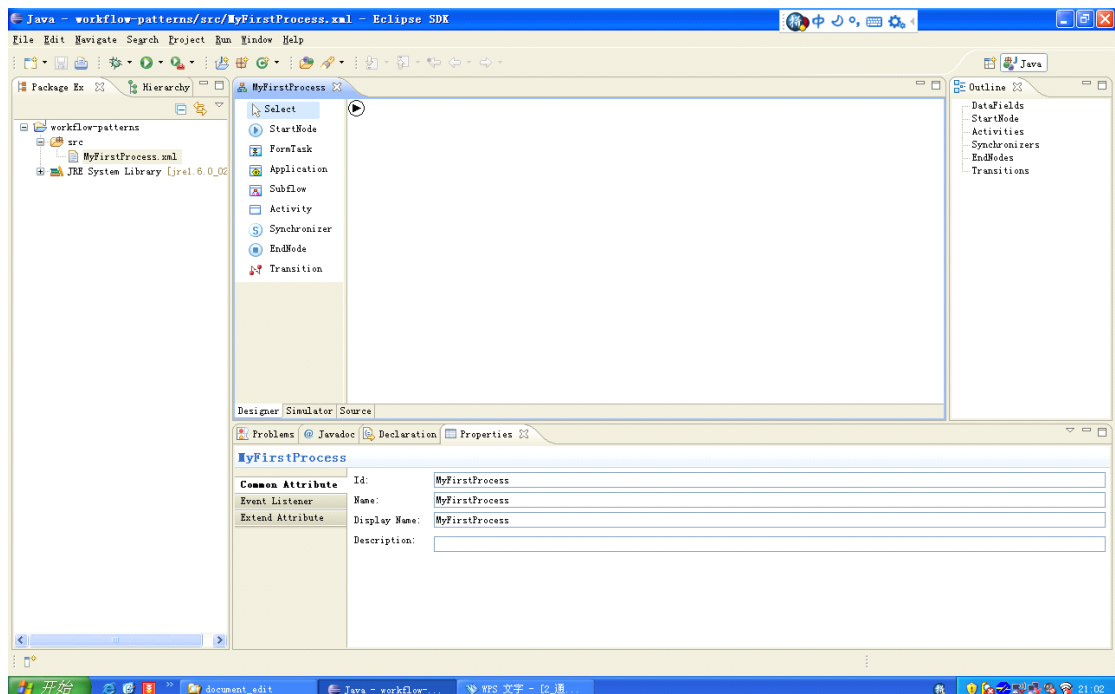
我们虚拟一个业务，该业务的流程如下：受理->核查->审批->结束。其中“核查”是一个后台自动执行的程序，该程序调用外部系统的接口检查新受理的业务合法性。

1) 创建一个流程定义文件

首先需要创建一个新的流程定义文件。从 File->New->Other... 打开对话框，在“Fire Workflow Wizards”目录中选择“Fire workflow process file”，点击 Next，如下图：




在下一 Wizard 页面中输入新的流程定义文件的名称“MyFirstProcess.xml”，系统默认将“MyFirstProcess”作为新流程的流程名称。系统为新创建的流程定义文件打开流程编辑界面，如下图：

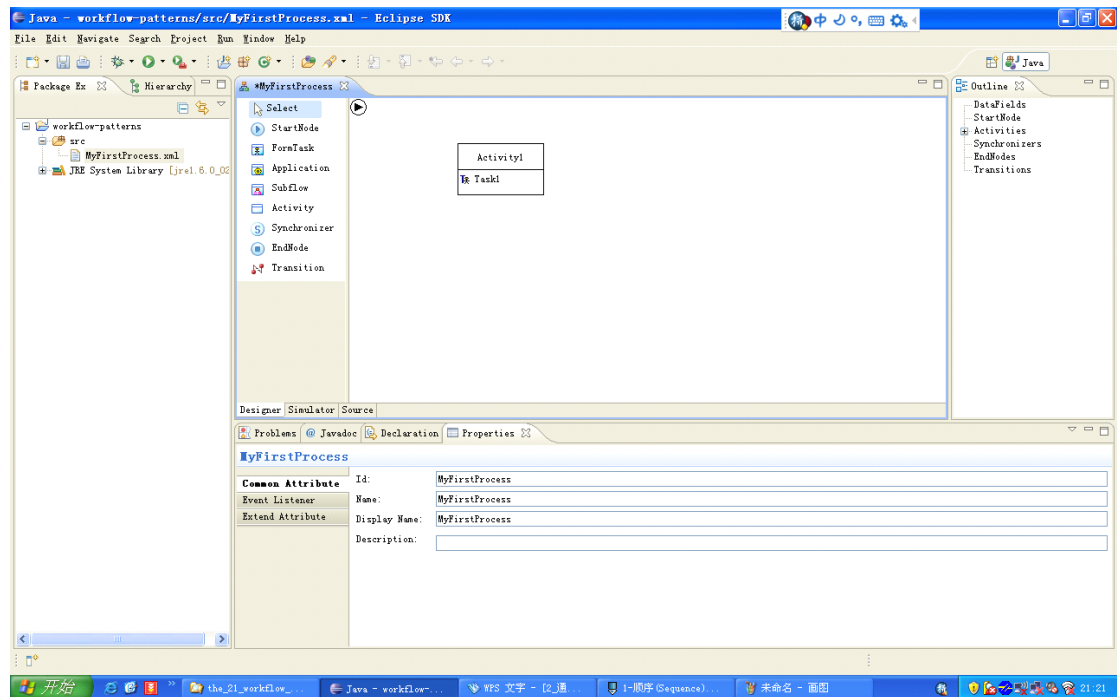


2) 给新流程增加相关的 Activity 和 Task

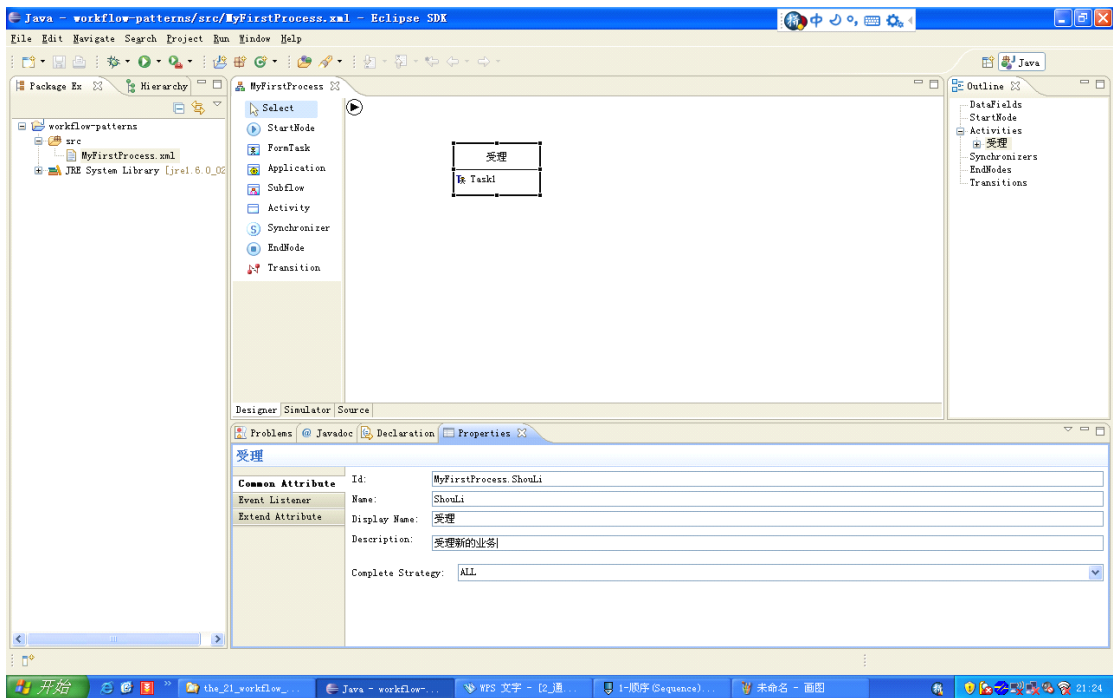
A) 受理环节

根据虚拟的业务场景，我们首先增加受理环节。在流程设计器工具栏上选择

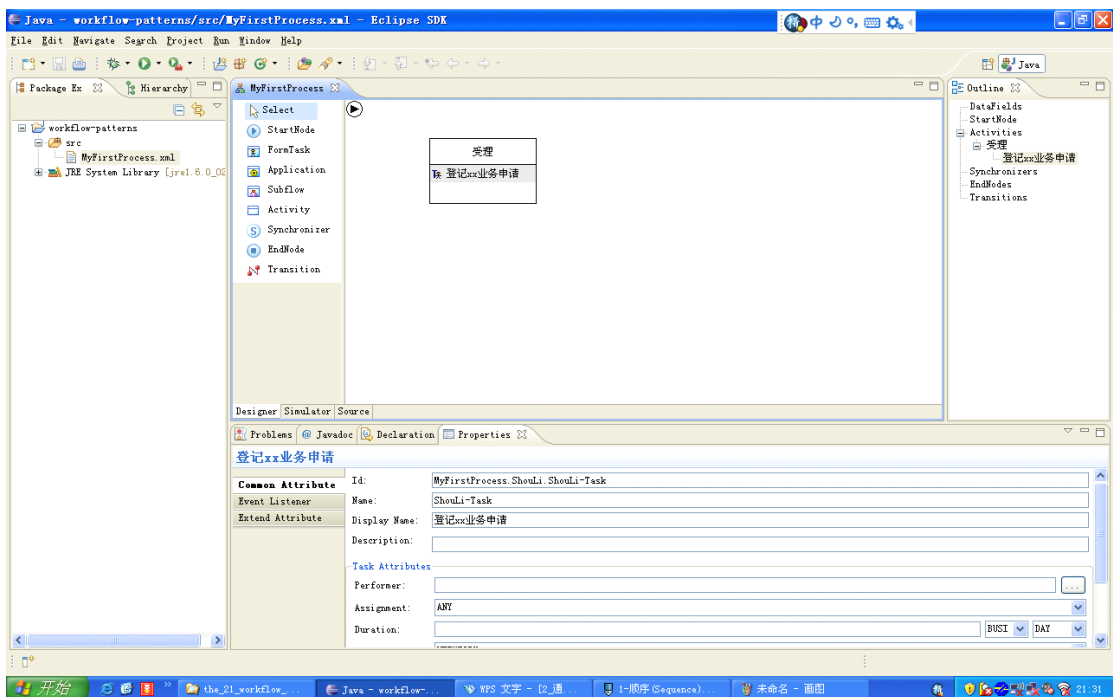
 **FormTask**，然后在设计界面点击鼠标左键。流程中新增了一个名称为“Activity1”的环节，同时里面自动增加了一个名称为 Task1 的 Form 类型的任务。如下图：



选中 Activity1，在属性界面中将其 Name 属性的值改成“ShouLi”，将其 DisplayName 属性的值改成“受理”。如下图

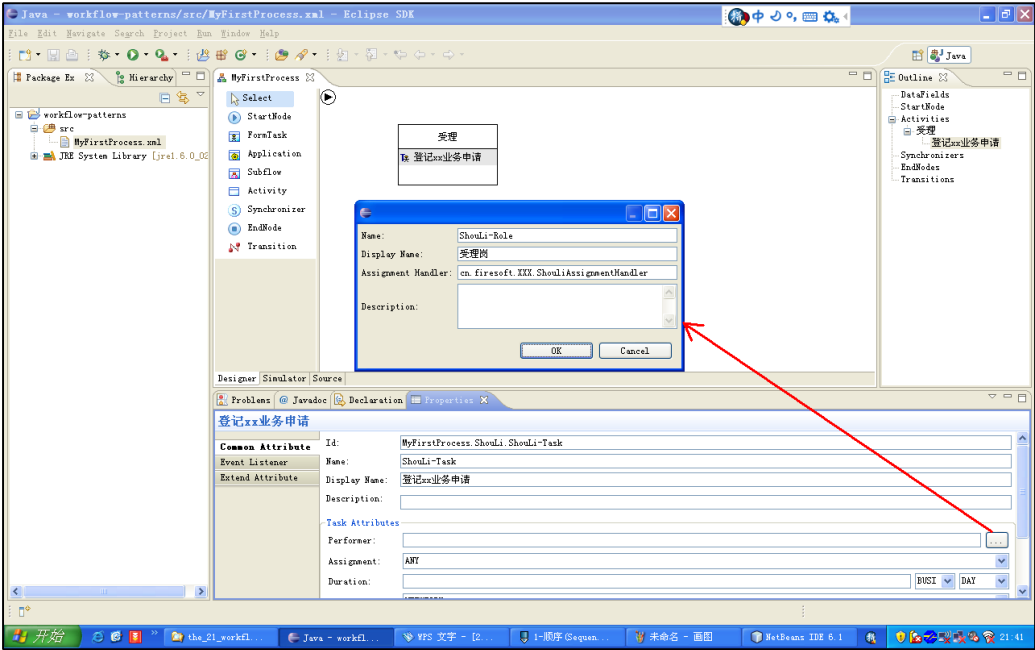


选中 Task1，在属性界面中将其 Name 属性的值改成“ShouLi-Task”，将其 DiplayName 的值也改成“登记 xx 业务申请”。修改后如下图




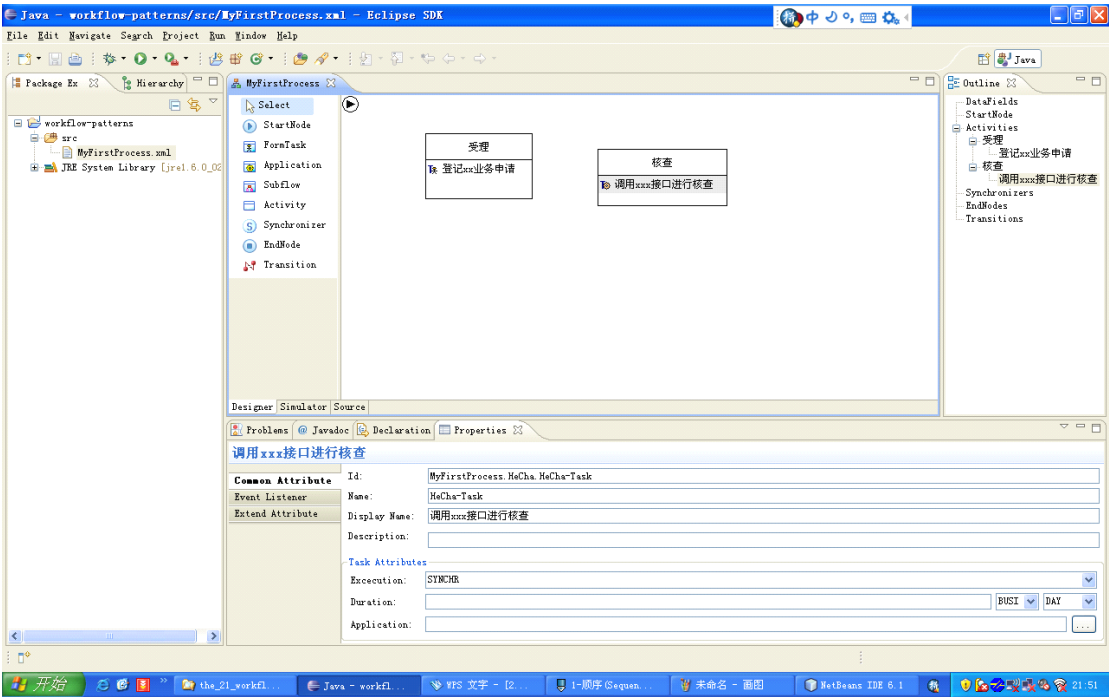
对于 Form 类型的 Task，一定要指定其操作者(Performer) 属性。在属性页的 Performer 属性项中点击其按钮，打开 Performer 编辑界面，如下图。我们将 Performer 的 Name 设置为“ShouLi-Role”，DisplayName 设置为“受理岗”。Assignment Handler 是一个实现了 org.fireflow.engine.ou.IAssignmentHandler 接口的任务分配处理类，任务分配的实际工作由该类完成。我们假设在该虚拟的项目中实现了一个名称为

cn.firesoft.XXX.ShouldAssignmentHandler 的类，完成受理岗的任务分配工作。操作者属性设置完成后，界面如下

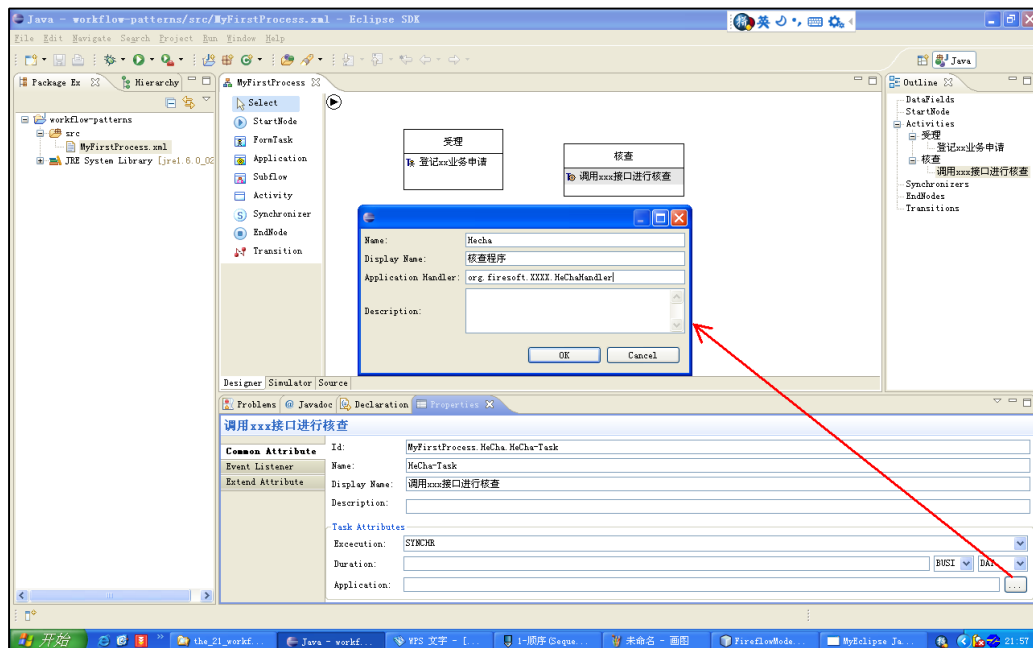


B) 核查环节

在设计器工具栏，点击  Tool 增加一个新的环节，该环节自动生成一个 Tool 类型的 Task。将此 Activity 的 Name 和 DisplayName 改成“HeCha”和“核查”；将 Task 的 Name 和 DisplayName 改成“HeCha-Task”和“调用 xxx 接口进行核查”。如下图



对于 Tool 类型的 Task，必须指定其 Application 属性。点击属性页 Application 后面的按钮，打开对话框；输入 Application 的 Name、DisplayName、Handler 如下：



C) 审批环节

审批环节也是一个通过人机交互的环节，其任务类型是 Form 类型。该环节相关的属性值如下：

Activity Name: ShenPi

Activity DisplayName : 审批

Task Name : ShenPi-Task



Task DisplayName : 业务领导审批

Performer Name: ShenPi-Role

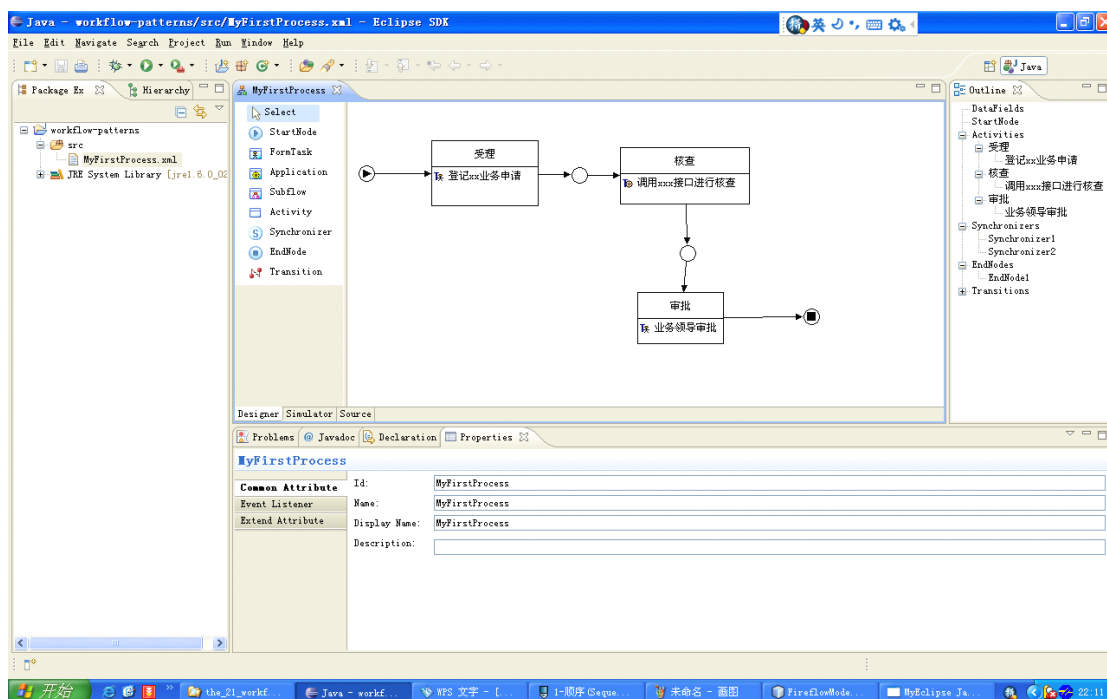
Performer DisplayName: 审批岗

Performer Assignment Handler: cn.firesoft.XXXX.ShenPiAssignmentHandler

D) 增加结束节点和转移


点击  EndNode 给流程增加一个结束节点；点击  Transition，顺序连接各环节。

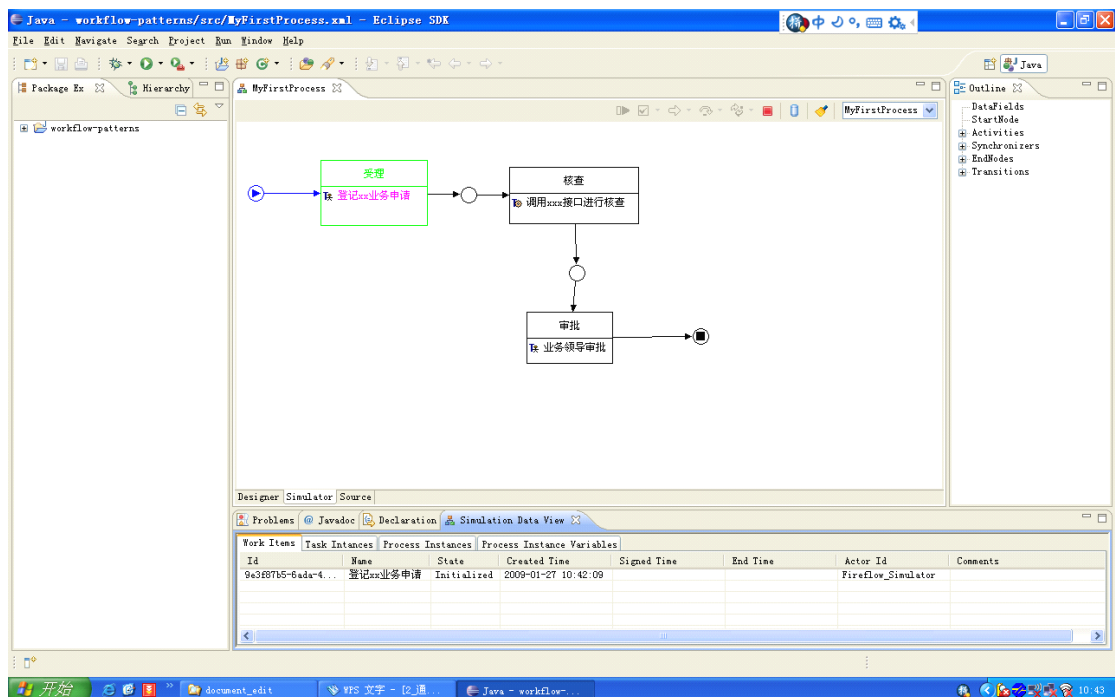
设计器会自动在两个环节直接增加同步器节点。流程设计完成后，如下图



3) 模拟流程的执行

a. 创建新的流程实例

点击模拟器工具栏的，创建新的流程实例。新的流程实例创建完毕后，模拟器自动启动第一个环节及其任务。“流程执行进程视图”用不同的颜色表示各节点的运行状态，蓝色表示已经执行完毕，绿色表示正在运行中，粉红色表示已经初始化等待签收的任务。在下图中，“受理”环节正在运行中，“登记 xx 业务申请”已经初始化，等待签收。



打开模拟数据视图，在“Work Items”页面中我们可以看到，产生了一个新的工单，其状态是 **Initialized**，其 ActorId 是 **Fireflow_Simulator**（模拟器默认的操作者）。如下图

| Work Items | | | | | | | | |
|--------------------|------|-------------------|--------------|----------------------------|----------|----------|--------------------|--|
| Task Instances | | Process Instances | | Process Instance Variables | | | | |
| Id | Name | State | Created Time | Signed Time | End Time | Actor Id | Comments | |
| 9e3f87b5-6ada-4... | | 登记xx业务申请 | Initialized | 2009-01-27 10:42:09 | | | Fireflow_Simulator | |

切换到模拟数据视图的“Task Instances”页面，可以看到系统产生了一个新的任务实例，其状态也是 **Initialized**。如下图

| | | | | | | | | | | | |
|--|----------|-------------|---------------------|------------|--------------|----------|-----------|---------|-------------------|-------------------|--|
| Problems @ Javadoc Declaration Simulation Data View | | | | | | | | | | | |
| Work Items Task Instances Process Instances Process Instance Variables | | | | | | | | | | | |
| Id | Name | State | Create Time | Start Time | Expired Time | End Time | Task Type | Assi... | Task Id | Activity Id | |
| 899f9c33-1573-4... | 登记xx业务申请 | Initialized | 2009-01-27 10:42:09 | | | | FORM | ANY | MyFirstProcess... | MyFirstProcess... | |

切换到模拟数据视图的“Process Instances”页面，可以看到系统产生了一个新的流程实例，其状态是 **Started**。如下图

| | | | | | | | | | | | |
|--|--|----------------|---------|---------------------|---------------------|--------------|----------|----------------|---------|-------------------|-------------|
| Problems @ Javadoc Declaration Simulation Data View | | | | | | | | | | | |
| Work Items Task Instances Process Instances Process Instance Variables | | | | | | | | | | | |
| Id | | Name | State | Create Time | Start Time | Expired Time | End Time | Process Id | Version | Parent Process... | Parent Task |
| cfl9dk2e-ce3e-4... | | MyFirstProcess | Started | 2009-01-27 10:42:09 | 2009-01-27 10:42:09 | | | MyFirstProcess | 1 | | |

在模拟数据视图的“Process Instance Variables”页面，当前没有任何流程变量。


模拟器创建流程实例的方法与你在你的项目中创建流程实例的方法完全相同，都是调用 **Fire Workflow Engine** 的相关 API 实现。唯一的不同在于，模拟器将后台数据库中数据的变化情况及状态通过图形的方式展示出来。创建流程实例并执行该流程实例的代码如下：

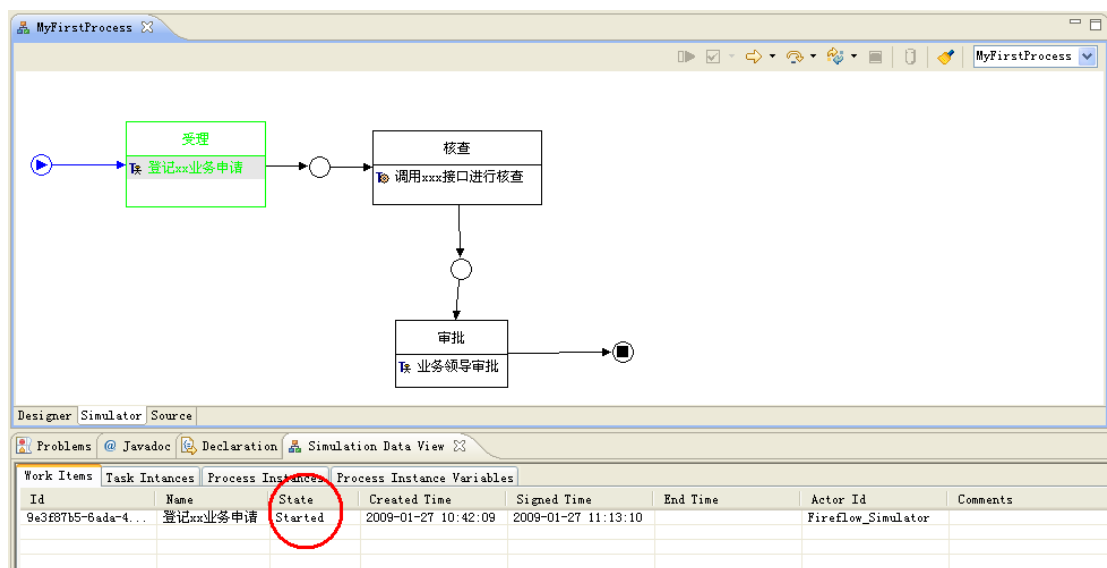
```
//获得 WorkflowSession 对象
IWorkflowSession fireflowSession = ctx.getWorkflowSession();
//根据流程名称创建流程实例
IProcessInstance processInstance = fireflowSession
```

```
.createProcessInstance(workflowProcess.getName());  
//调用流程实例的 run()方法启动流程实例,  
processInstance.run();
```

b. 签收工单


一般情况下，工单（Work Item）都需要被签收后才允许执行相关的表单操作。签收在实际业务中表现为上下工作岗位之间的工作交接，在 Fire Workflow 工作流系统中表现为对工单的“认领”。通常情况下 Fire Workflow 会将工单分配给具有操作权限的所有用户，如果其中某个用户签收了该工单，那么该工单就会在其他用户的任务列表中删除（会签情况除外）。

在“流程执行进程视图”中，选中粉红色的任务，然后点击 。则对应的工单被签收。相关的任务颜色由粉红色变成绿色，WorkItem 的状态由 Initialized 变成 Started。如下图所示

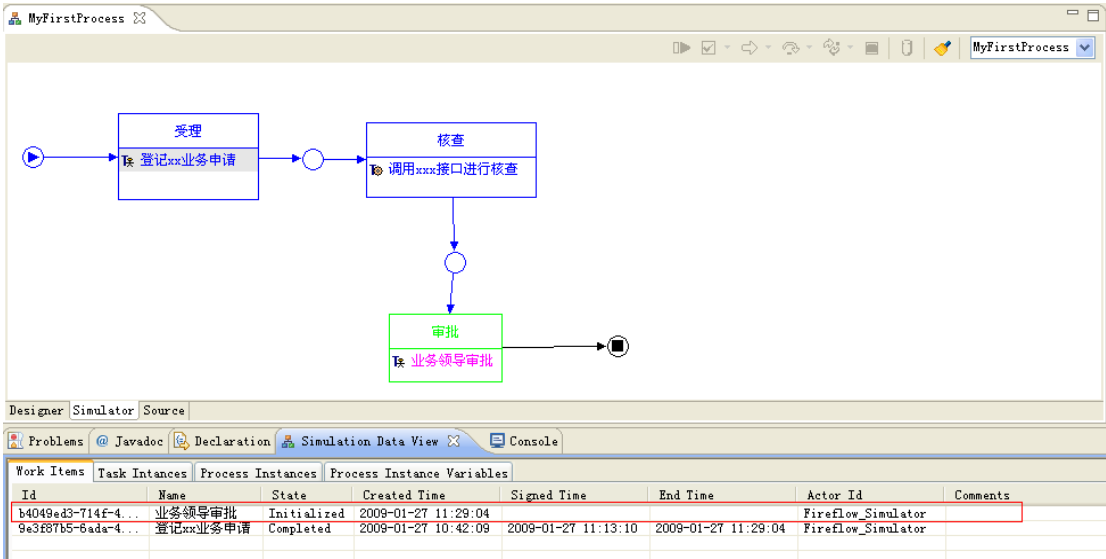


签收工单按钮是一个 DropDown 类型的工具栏按钮，默认情况下是以 Fireflow_Simulator 用户登录工作流系统进行操作，也可以以其他 ActorId 登录工作流系统进行操作，您可以在《3_各种工作流模式的实现》文档中关于会签的章节看到其用法。

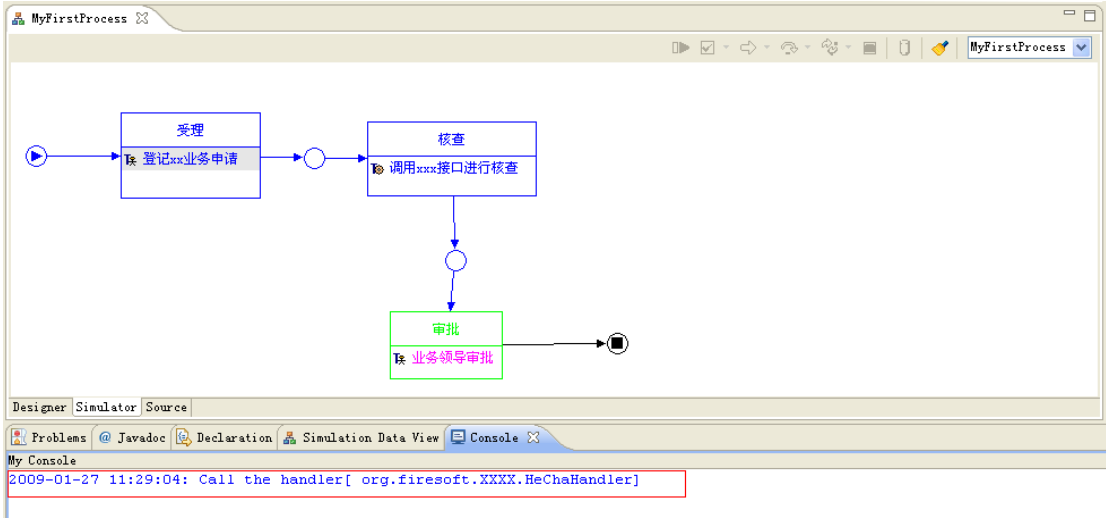
c. 完工单

在实际业务中，工单对应的业务表单录入完毕后需要调用 WorkItem.complete()方法完成该工单，并启动下一个环节。在模拟器中通过工具栏按钮  实现对该方法的调用。工单完成后，对应的 Task 以及 Activity 的颜色变成蓝色，同时下一个环节被启动。

由于在本示例中，“受理”的下一个环节是 Tool 类型的 Task，工作流引擎会自动执行并结束这个 Task，然后启动“审批”Task。如下图



模拟器并不会像在真实系统中一样执行 Tool Task 的 application handler，仅仅在控制台上打印一行提示信息，表示已经发现了相关的 application handler，如下图。



d. 签收并完成审批环节相关的工单，流程实例终止

与上述操作类似，可以签收并完成“审批”环节的相关工单，此时工作流引擎发现流程实例已经没有活动的 Token，于是结束整个流程实例，其状态变成“Completed”，如下图

