# 第五章：Core Engine API

# 5.1、概述

# 5.2、KieBase

1、一个基础库可以包含所有在回话中需要执行的流程定义

2、一个基础库可以新建一个会话(KieSession)

KieHelper kieHelper = new KieHelper();

KieBase kieBase = kieHelper.addResource(ResourceFactory.newClassPathResource("MyProcess.bpmn")).build();

# 5.3、KieSession

1、一个会话可以通过流程ID开启一个流程，从而创建一个流程实例

KieSession ksession = kbase.newKieSession();

ProcessInstance processInstance = ksession.startProcess("com.sample.MyProcess");

5.3.1 ProcessRuntime (见ProcessRuntime接口)

1）为了与所有的流程关联起来ProcessRuntime接口中定义了所有会话的方法

5.3.2 Event Listeners (见ProcessEventListener接口)

1）会话提供注册和移除监听，一个流程事件监听(ProcessEventListener)可以用于相关的流程事件，比如开启和完成一个流程，进入和离开一个流程等等。

2）一个事件对象提供相关信息的接口，比如流程实例和节点实例关联到事件

3）JBPM现成提供一个现成的监听用于创建一个审核日志

import org.kie.api.KieServices;

import org.kie.api.logger.KieRuntimeLogger;

KieRuntimeLogger logger = KieServices.Factory.get().getLoggers().newFileLogger(ksession, "test");

// add invocations to the process engine here,

// e.g. ksession.startProcess(processId);

...

logger.close();

# 5.4、RuntimeManager 运行管理器

5.4.1 概述

1）RuntimeManager简化和增强库API的运用

2）运行管理器的首要任务就是管理和提供运行引擎给来访者

3）两个重要的JBPM引擎元素：KieSession、TaskService

# 5.5 Configuration

1. 主要设置java虚拟机的参数

# 第六章 Process

# 6.1 BPMN 2.0

1. BPMN是一种规范，定义了怎样图形化呈现一个业务流程的标准，还包括执行元素定义的语意和怎样存储一个流程定义的XML格式。
2. 三种主要类型的节点：
3. Events：事件节点，模拟特定事件的发生，开启、结束事件。
4. Activity：在执行流程过程中，活动定义了不同的action去执行
5. Gateways：在流程中，网关可以定义多个不同的路径
6. 使用BPMN 2.0格式，通过以下几种形式去创建你自己的流程：
7. the JBPM Designer 是一个基于web的开源编辑器，它支持BPMN 2.0 格式。
8. A New BPMN2 Eclipse plugin 是一个eclipse的组件去创建流程
9. 直接通过手动的创建一个BPMN2.0 格式的XML文件

# 6.2 Process

1）一个业务流程是一张图， 它描述了一系列需要被执行步骤的顺序。

2）通过以下三种方法创建一个流程：

1. 采用图形化流程编辑器，比如jbpm web设计器和eclipse BPMN2 Modeler
2. 使用XML定义流程，在BPMN2规范中根据XML流程格式定义流程
3. 直接使用流程API创建一个流程

# 6.3 Activitys

1）script task 脚本任务

1、有一个传入连接和传出连接

2、这个关联的Action规定什么可以被执行，这个方言用于action编码和这个action实际的代码。

这个代码可以访问任何变量和全局变量

3、这里有一个预定义变量kcontent关联ProcessContext对象，该对象可以访问流程实例ProcessInstance或者节点实例NodeInstance,可以去获得和设置变量

可以通过kcontent.getKieRuntime()访问ksession.

4、当某个流程到达任务脚本时，它将会执行action，然后继续运行下一个节点。

5、注意事项：

\* 当创建一个高级业务流程，应该被业务人员理解

\* 脚本应该是即时的

\* 应该避免去连接外部系统

\* 脚本不应该抛出异常

2）Service task

1、所有的工作执行在流程引擎的外面

2、里面预定义了不同的服务，比如：发送一个邮件、记录一个信息等等。使用一个唯一的名称和通过定义传入参数和传出结果（结果与工作类型关联起来），用户可以定义特定域或者工作项目

3、当流程执行到服务任务时，相关联的工作内容将会被执行。

4、服务任务应该有一个传入连接和一个传出连接。

3）User task

1、流程可以参与一个需要人工扮演来执行的任务。

2、用户任务有一个传入连接和一个传出连接

3、用户任务可以结合泳道swimlanes，泳道可以指定多个人工任务给相同的扮演者。

4）Reusable sub-process (Call Activity)

1、在当前流程中调用另外一个流程。

2、当流程执行到一个复用的子流程，引擎将会开启指定ID的流程，

5）Business rule task

1、代表一系列需要计算的规则，当流程执行到该节点将会计算规则

2、规则定义在不同的文件中，文件的定义通过the Drools rule format

6）Embedded sub-process

1、不仅一部分流程嵌入在该节点里，而且在这个节点容器里定义的额外的变量在所有节点是可以使用的

* + 1. 会有一个开启节点和一个结束节点

# 6.4 Events

1）Start event

流程的开启 ，一个流程应该有一个确切的流程开启节点

2）End event

流程的结束，一个流程可以有一个或者多个结束节点

3）Throwing error event

1、一个错误事件在流程中用于预示一个异常情况

2、流程将会寻找适当的错误处理程序，该程序有处理这种错误类型。如果没有找到错误处理程序，流程实例将会中止。

4）Intermediate event

中间体事件有两种：一种捕获定时器事件，另一种是捕获信号事件

4.1）Catching timer event

1、在指定的一段时间内，一个定时器将会触发一次或多次

2、这个定时器指定延迟多久这个定时器将会第一次触发

4.2）Catching signal event

1、该节点用于在流程执行期间去响应内部或者外部的事件

2、该节点指定预期事件的类型

3、一个流程事件可以暗示一个特定事件发生使用

ksession.signalEvent(eventType, data, processInstanceId)

4、你可以生成一个信号，这个信号来自内部的一个流程实例。可以使用一个脚本

kcontext.getKieRuntime().signalEvent(eventType, data, kcontext.getProcessInstance().getId());

# 6.5 Gateways

1）Diverging gateway

1、你可以在流程中创建一个分支，一个分流网关节点有一个传入连接、多个传出结果连接。

2、目前支持的三种类型的网关节点

A、and or parallel 和或者并行网关，意味着这个节点将会继续所有的传出连接节点

B、XOR or exclusive 异或或者独有的，意味着将会选择一个确切的传出连接，通过计算每一个传出连接的约束。你应该确保每次都有一个确切的约束会被执行。

C、OR or inclusive 或者或者包含，意味着所有计算为true的条件约束传出连接都会被执行

2）Converging gateway

1、允许你同步多个分支

2、一个会聚节点有两个或者多个传入连接和一个传出连接

3、目前支持三种类型的网关节点：

A、and or parallel 和或者并行网关，意味着会等待所有传入连接节点都完成后才会继续下一个节点

B、XOR or exclusive 异或或者独有的，意味着一旦完成一个传入连接将会继续下一个节点

C、OR or inclusive 或者或者包含，（没看懂）

# 6.6 others

1）Variables

1、

# 第七章 Human Tasks

7.1 介绍

该节点需要人工执行该流程。

7.2 在流程中使用用户任务

该节点包含哪些属性

7.3 数据映射

人工任务经常需要展现于任务相关的数据。

7.4 任务的生命周期

7.5 任务的批准

7.6 任务服务和流程引擎

7.7 人工任务与任务服务的相互作用

# 第八章 持久化和事务

8.1 流程实例状态

持久化流程自身的定义和历史相关的信息。

8.2 审计日志

1）审计数据模型

2）在一个数据库中保存流程事件

3）为了进一步处理，在jms队列中存储流程事件

8.3 事务

1）容器管理事务

8.4 配置

1）添加依赖

2）人工配置该引擎到持久化中

3）引擎配置只为了持久性测试使用JBPMHelper