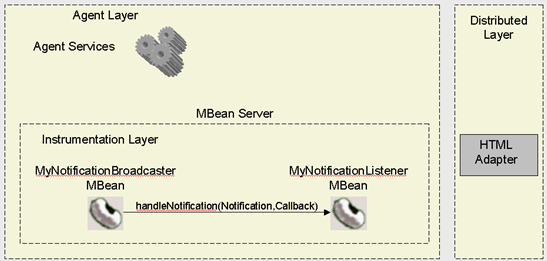
# JMX与Notification通知

## 1、简介

JMX中的Notification也就是“通知”、“提示”或“报告”的意思，将外部对Mbean数据的更新通知给相关程序，一个总的图示如下。



## 2、MBean接口

本次的测试接口也比较简单，但注意，Mbean接口只能有一个实现，多个实现时会出现异常。

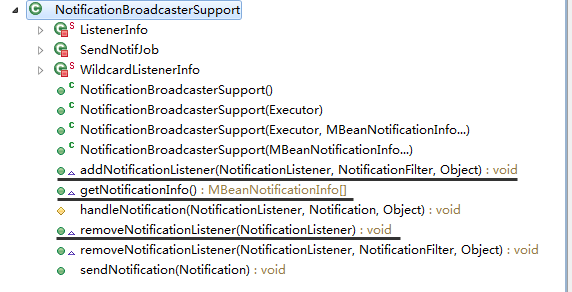
|  |
| --- |
| **package** org.xman.jmx.notification;  **public** **interface** MyNotificationMBean {  **public** **void** **setGreeting**(String greeting);  **public** String **getGreeting**();  **public** **void** **printGreeting**();  } |

## 3、被观察者Mbean实现

显然，MBean就是被观察被通报的对象，它的数据被更新时它需要向上作出反应。在JMX中，它需要直接实现一个叫NotificationBroadcaster的接口，或者继承一个已实现该接口的类。NotificationBroadcaster接口代码如下所示：

|  |
| --- |
| **public** **interface** NotificationBroadcaster {  **public** **void** **addNotificationListener**(  NotificationListener listener,  NotificationFilter filter,  Object handback)  **throws** IllegalArgumentException;  **public** MBeanNotificationInfo[] **getNotificationInfo**();  **public** **void** **removeNotificationListener**(NotificationListener listener)  **throws** ListenerNotFoundException;  } |

在很多时候，我们继承一个叫NotificationBroadcasterSupport的类，该类已经实现了NotificationBroadcaster接口。



我们的测试类代码如下。

|  |
| --- |
| **package** org.xman.jmx.notification;  **import** javax.management.Notification;  **import** javax.management.NotificationBroadcasterSupport;  **public** **class** MyNotification **extends** NotificationBroadcasterSupport **implements** MyNotificationMBean {  **public** **MyNotification**() {  **this**.greeting = "Hello, My Notification!";  }  **public** **MyNotification**(String greeting) {  **this**.greeting = greeting;  }  **public** **void** **setGreeting**(String greeting) {  **this**.greeting = greeting;  Notification notification = **new** Notification("org.xman.jmx.notification", **this**, -1, System.*currentTimeMillis*(),  greeting);  sendNotification(notification);  }  **public** String **getGreeting**() {  **return** greeting;  }  **public** **void** **printGreeting**() {  System.out.println(greeting);  }  **private** String greeting;  } |

## 4、观察者

我们观察者通常也就是由Agent来扮演，它只需要实现NotificationListener接口即可。

|  |
| --- |
| **package** org.xman.jmx.notification;  **import** java.lang.management.ManagementFactory;  **import** javax.management.MBeanServer;  **import** javax.management.MBeanServerFactory;  **import** javax.management.Notification;  **import** javax.management.NotificationListener;  **import** javax.management.ObjectName;  **import** com.sun.jdmk.comm.HtmlAdaptorServer;  **public** **class** MyNotificationAgent **implements** NotificationListener {  **private** MBeanServer mbs = **null**;  **public** **MyNotificationAgent**() {  // 使用该MBeanServer可以通过jconsole来查看MBean情况  mbs = ManagementFactory.*getPlatformMBeanServer*();  // 使用该MBeanServer需要通过访问页面http://localhost:9092来查看页面  mbs = MBeanServerFactory.*createMBeanServer*("HelloSpy");  HtmlAdaptorServer adapter = **new** HtmlAdaptorServer();  adapter.setPort(9092);  MyNotification hw = **new** MyNotification();  ObjectName adapterName = **null**;  ObjectName helloWorldName = **null**;  **try** {  helloWorldName = **new** ObjectName("HelloSpy:name=helloJmx");  mbs.registerMBean(hw, helloWorldName);  adapterName = **new** ObjectName("HelloSpy:name=htmladapter,port=9092");  mbs.registerMBean(adapter, adapterName);  hw.addNotificationListener(**this**, **null**, **null**);  adapter.start();  } **catch** (Exception e) {  e.printStackTrace();  }  }  **public** **void** **handleNotification**(Notification notif, Object handback) {  System.out.println("Receiving notification...");  System.out.println(notif.getType());  System.out.println(notif.getMessage());  }  @SuppressWarnings("unused")  **public** **static** **void** **main**(String args[]) {  MyNotificationAgent agent = **new** MyNotificationAgent();  }  } |

## 5、小结

如果需要实现通知功能，只需要在Mbean实现类继承NotificationBroadcasterSupport类，且Agent端实现NotificationLister接口即可。

