Zookeeper通知机制

<http://www.atguigu.com/jsfx/6322.html>

# 1. 通知机制简介

客户端注册监听它关心的目录节点，当目录节点发生变化（数据改变、被删除、子目录节点增加删除）时，zookeeper会通知客户端。

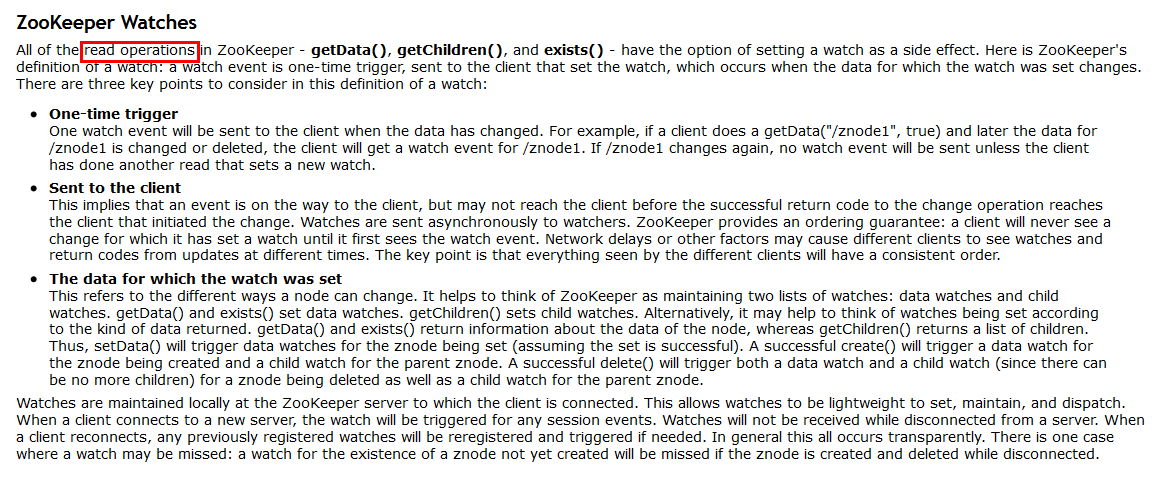
## 1.1  是什么

ZooKeeper 支持watch(观察)的概念。客户端可以在每个znode结点上设置一个观察。如果被观察服务端的znode结点有变更，那么watch就会被触发，这个watch所属的客户端将接收到一个通知包被告知结点已经发生变化，把相应的事件通知给设置过Watcher的Client端。

Zookeeper里的所有读取操作：getData(),getChildren()和exists()都有设置watch的选项。

一句话：异步回调的触发机制

# 2. Watch事件



## 2.1 一次触发

当数据有了变化时zkserver向客户端发送一个watch,它是一次性的动作，即触发一次就不再有效，类似一次性纸杯。

只监控一次。如果想继续Watch的话，需要客户端重新设置Watcher。因此如果你得到一个watch事件且想在将来的变化得到通知，必须新设置另一个watch。

## 2.2 发往客户端

Watches是异步发往客户端的，Zookeeper提供一个顺序保证：在看到watch事件之前绝不会看到变化，这样不同客户端看到的是一致性的顺序。

在（导致观察事件被触发的）修改操作的成功返回码到达客户端之前，事件可能在去往客户端的路上，但是可能不会到达客户端。观察事件是异步地发送给观察者（客户端）的。ZooKeeper会保证次序：在收到观察事件之前，客户端不会看到已经为之设置观察的节点的改动。网络延迟或者其他因素可能会让不同的客户端在不同的时间收到观察事件和更新操作的返回码。这里的要点是：不同客户端看到的事情都有一致的次序。

## 2.3 为数据设置watch

节点有不同的改动方式。可以认为ZooKeeper维护两个观察列表：数据观察和子节点观察。getData()和exists()设置数据观察。getChildren()设置子节点观察。此外，还可以认为不同的返回数据有不同的观察。getData()和exists()返回节点的数据，而getChildren()返回子节点列表。所以，setData()将为znode触发数据观察。成功的create()将为新创建的节点触发数据观察，为其父节点触发子节点观察。成功的delete()将会为被删除的节点触发数据观察以及子节点观察（因为节点不能再有子节点了），为其父节点触发子节点观察。

观察维护在客户端连接到的ZooKeeper服 务器中。这让观察的设置、维护和分发是轻量级的。客户端连接到新的服务器时，所有会话事件将被触发。同服务器断开连接期间不会收到观察。客户端重新连接 时，如果需要，先前已经注册的观察将被重新注册和触发。通常这都是透明的。有一种情况下观察事件将丢失：对还没有创建的节点设置存在观察，而在断开连接期 间创建节点，然后删除。

## 2.4 时序性和一致性

Watches是在client连接到Zookeeper服务端的本地维护，这可让watches成为轻量的，可维护的和派发的。当一个client连接到新server，watch将会触发任何session事件，断开连接后不能接收到。当客户端重连，先前注册的watches将会被重新注册并触发。

关于watches，Zookeeper维护这些保证：

（1）Watches和其他事件、watches和异步恢复都是有序的。Zookeeper客户端保证每件事都是有序派发

（2）客户端在看到新数据之前先看到watch事件

（3）对应更新顺序的watches事件顺序由Zookeeper服务所见

# 3. 程序代码

## 3.1 一次性

