

分布式利器Zookeeper (一)



张丰哲 (/u/cb569cce501b) ✓已关注

2017.05.04 22:13* 字数 1655 阅读 1292 评论 0 喜欢 33 赞赏 3

(/u/cb569cce501b)

Zookeeper不论是在实际项目中，还是在各种分布式开源项目都得到了广泛应用，从本篇博客开始，将为大家带来我对Zookeeper的认识。这个系列将会涵盖Zookeeper的介绍、环境搭建、配置说明、Java操作Zookeeper（原生API方式）、zkclient操作Zookeeper方式、Zookeeper的典型应用场景分析以及Curator框架等。

Hello,Zookeeper

Zookeeper是一个高可用的分布式管理与协调框架，基于ZAB算法（原子消息广播协议）实现。ZK能够很好的保证分布式环境下的数据一致性，这一特性使得ZK是处理分布式一致性问题利器！这个利器有哪些特性呢，看看下面你就了解了。

可靠性：一旦Zookeeper成功的应用了一个事务，并完成对client的响应，那么Zookeeper内部集群的所有服务器的状态都会是一致的保留下来。

单一视图：由于上面可靠性的保证，使得无论client连接的是ZK集群中的哪个服务器，所看到的数据都是一致的。

顺序一致性：从一个client发起的请求，最终会严格的按照发起的顺序被应用到Zookeeper中去。【实质上，ZK会对每一个client的每一个请求，进行编号，说白了，就是分配一个全局唯一的递增编号，这个编号反映了所有事务操作的先后顺序。】

实时性：通常意义下的实时性是指一旦事务被成功应用，那么client会立刻从服务器端获取到变更后的新数据。ZK仅仅能够保证在一定时间内，client最终一定能从服务器上获取到最新的数据。

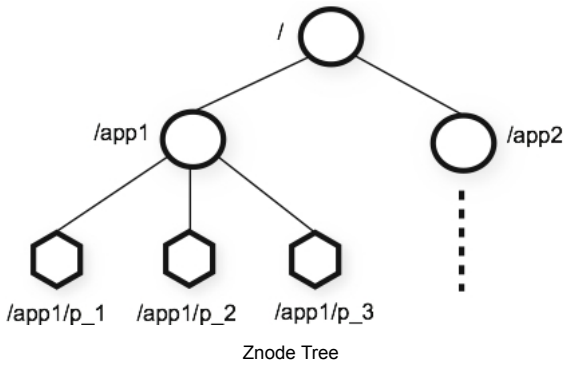
高可用：在ZK集群内部，会有一个Leader，多个Follower。一旦Leader挂掉，那么ZK会通过Paxos算法选举出新的Leader，只要ZK集群内部的服务器有一半以上正常工作，那么ZK就能对外正常提供服务！

简单的数据结构：类似于Linux文件系统的树形结构，简单，实用！（树形层次空间）

高性能：性能有多高呢，举个栗子，比如我们经常通过创建临时节点来处理分布式锁，要知道临时节点是存储在内存中的，在读场景压力测试下，QPS高达10W+！也就是说ZK在读场景下，性能非常突出！

初步认识Zookeeper的数据模型



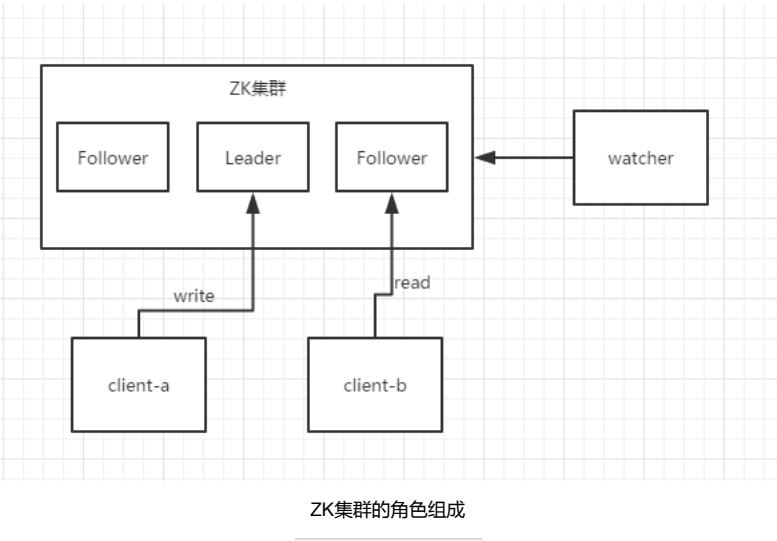


每一个节点被称为znode，znode可以有子节点目录，并且每个znode可以存储数据（特别需要注意的是临时节点不可以有子节点）

znode如果是临时节点，意味着创建这个znode的client一旦与ZK集群失去联系，这个临时的znode将被自动删除。（事实上，client与ZK通信是采用长连接方式，并通过心跳的方式保持连接，这种状态就是session，一旦session失效，就是连接断开，临时节点会被删除掉）

znode是可以被监控的，不论是znode本身的数据变化，还是该znode下的子节点的变化，都可以进行监控，这也是ZK的核心特性。（很多应用场景就是基于这个特性，后续进行详细介绍）

初步认识Zookeeper的角色组成



这里，我们先了解下ZK Server的身份特性：

Leader：负责客户端的write类型的请求

Follower：负责客户端的read类型请求，并可以参与Leader的选举

watcher：后文介绍。

^

+

🔖

🔗

install Zookeeper

这里以zookeeper-3.4.5版本为例，搭建一个ZK集群（至少要求3个节点，并且各个ZK SERVER之间系统时间保持一致）。使用的机器列表是：192.168.99.121、192.168.99.122、192.168.99.123。

以192.168.99.121为例进行说明：

```
tar -xvf zookeeper-3.4.5.tar.gz
```

解压

```
export JAVA_HOME=/software/jdk1.7
export CLASSPATH=.
export ZOOKEEPER_HOME=/software/zookeeper-3.4.5
export PATH=$JAVA_HOME/bin:$ZOOKEEPER_HOME/bin:$PATH
```

/etc/profile中配置ZK环境变量（注意source下）

```
# The number of milliseconds of each tick
tickTime=2000
# The number of ticks that the initial
# synchronization phase can take
initLimit=10
# The number of ticks that can pass between
# sending a request and getting an acknowledgement
syncLimit=5
# the directory where the snapshot is stored.
# do not use /tmp for storage, /tmp here is just
# example sake.
dataDir=/software/zookeeper-3.4.5/data
# the port at which the clients will connect
clientPort=2181
#
# Be sure to read the maintenance section of the
# administrator guide before turning on autopurge.
# http://zookeeper.apache.org/doc/current/zookeeperAdmin.html#sc_maintenance
#
# The number of snapshots to retain in dataDir
#autopurge.snapRetainCount=3
# Purge task interval in hours
# Set to "0" to disable auto purge feature
#autopurge.purgeInterval=1
server.0=192.168.99.121:2888:3888
server.1=192.168.99.122:2888:3888
server.2=192.168.99.123:2888:3888
```

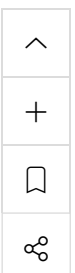
zoo.cfg

```
[root@mydream121 zookeeper-3.4.5]# mkdir data
[root@mydream121 zookeeper-3.4.5]#
```

创建ZK数据目录

```
[root@mydream121 data]# vim myid
[root@mydream121 data]# cat myid
0
[root@mydream121 data]#
```

myid



配置说明：

tickTime：ZK集群与客户端、ZK集群内部SERVER之间的心跳间隔，默认2S。

initLimit：在客户端连接ZK集群时，可以忍受多少个心跳次数。上述的配置表明，如果client初始连接ZK集群，在10*2S=20S内ZK集群没有返回连接成功，即意味着连接失败。

syncLimit：表示ZK集群内部Leader和Follower之间请求应答可以忍受多少个心跳次数。

dataDir：ZK保存数据以及日志的目录。

clientPort：ZK集群对外暴露的接口，即client访问ZK集群的端口（2181）。

server.x = IP:port-a:port-b

X表示是第几号服务器，从0开始编号，并和dataDir下的myid文件对应。

port-a表示Leader和Follower进行信息通信的端口（2888）。

port-b表示Follower进行选举的端口（3888）。

```
[root@mydream121 bin]# zkServer.sh start
JMX enabled by default
Using config: /software/zookeeper-3.4.5/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
```

分别启动ZK

```
[root@mydream123 ~]# zkServer.sh status
JMX enabled by default
Using config: /software/zookeeper-3.4.5/bin/../conf/zoo.cfg
Mode: leader
[root@mydream123 ~]#
```

查看ZK身份



搭建的ZK集群

基本的ZK命令

^

+

🔖

🔗

```
[zk: localhost:2181(CONNECTED) 0] ls /
[zookeeper]
[zk: localhost:2181(CONNECTED) 1] create /node nodeinfo
Created /node
[zk: localhost:2181(CONNECTED) 2] ls /
[node, zookeeper]
[zk: localhost:2181(CONNECTED) 3] get /node
nodeinfo
cZxid = 0x100000008
ctime = Thu May 04 06:05:27 PDT 2017
mZxid = 0x100000008
mtime = Thu May 04 06:05:27 PDT 2017
pZxid = 0x100000008
cversion = 0
dataVersion = 0
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 8
numChildren = 0
[zk: localhost:2181(CONNECTED) 4] set /node nodeinfo2
cZxid = 0x100000008
ctime = Thu May 04 06:05:27 PDT 2017
mZxid = 0x100000009
mtime = Thu May 04 06:05:52 PDT 2017
pZxid = 0x100000008
cversion = 0
dataVersion = 1
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 9
numChildren = 0
[zk: localhost:2181(CONNECTED) 5] []
```

ls/create/get/set

通过zkCli.sh进入客户端进行操作：

查找ls、创建znode节点create（注意每个节点都是有值的）、获取get、设置set。

我们观察下下面几个属性：

ctime和cZxid是一对，表示创建ZNODE的时间和事件编号；

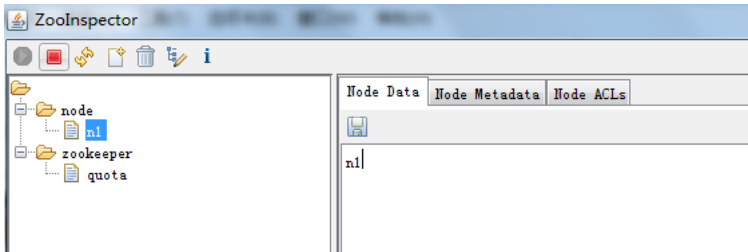
mtime和mZxid是一对，表示修改ZNODE数据内容的时间和事件编号；（通过set指令会改变这2个属性，但是该节点的子节点的变化不会影响该节点）

dataVersion：表示ZNODE数据的版本，注意利用JAVA 原生API进行delete ZNODE时，需要提供version才可以删除ZNODE。（当然我们可以提供-1来跳过版本检查机制）

dataLength、numChildren都好理解。

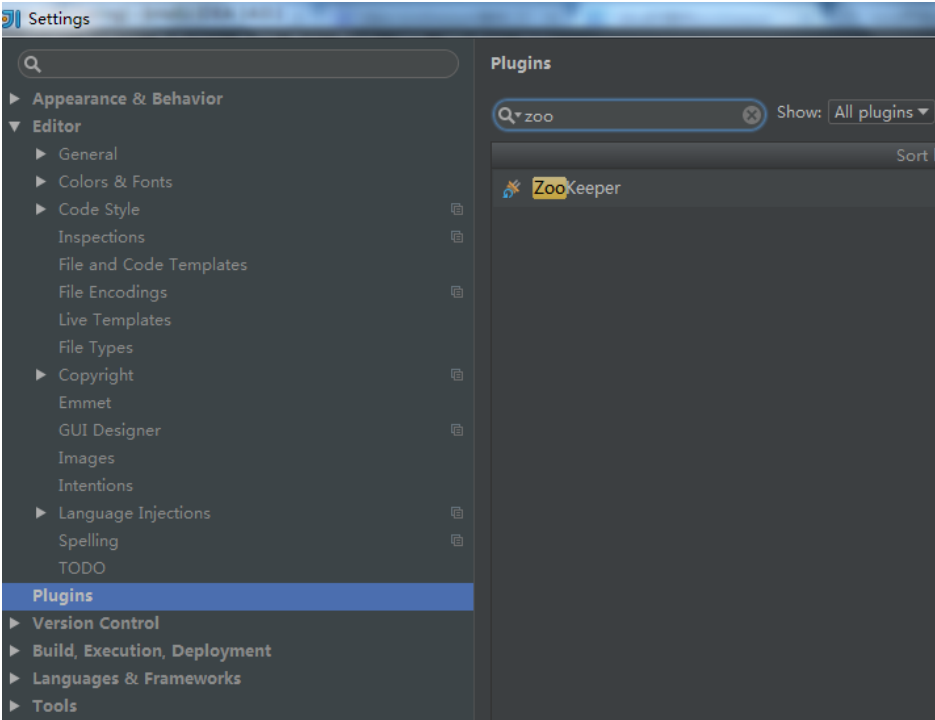
另外，注意rmr指令可以递归删除ZNODE；而delete只可以删除指定ZNODE。

ZooInspector工具及IntelliJ IDEA与ZK集成

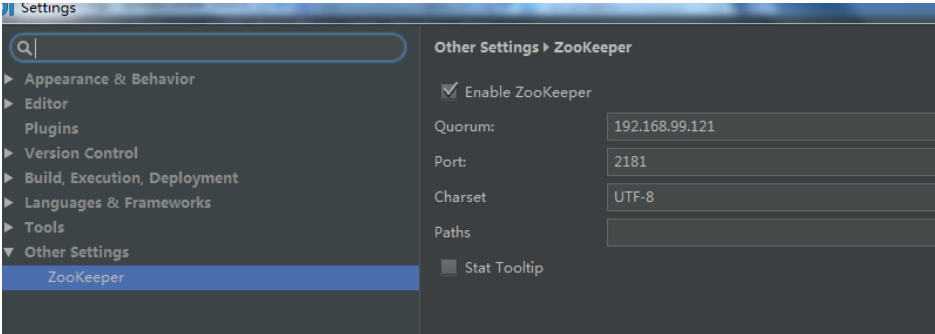


ZooInspector工具

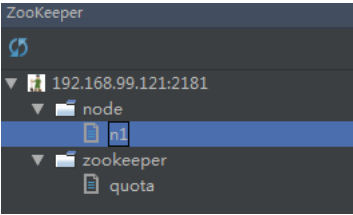
ZooInspector是一个可运行的JAR，运行后直接连接ZK集群中的任何一个SERVER即可。



安装ZK插件

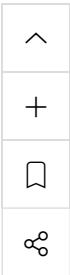


配置ZK插件并启用



ZK视图

OK，到这里，一切准备工作就准备就绪了，晚安吧~



咱们下期来看JAVA操作ZK（基于Zookeeper的原生API）、分布式锁探讨、Watch特性等~

📅 日记本 (/nb/10261827)

举报文章 © 著作权归作者所有



张丰哲 (/u/cb569cce501b) ♂

写了 63893 字，被 2144 人关注，获得了 1674 个喜欢

(/u/cb569cce501b)

✓ 已关注




资深Java工程师 51CTO博客【2014-2016】：<http://zhangfengzhe.blog.51cto.com/>

好好学习，天天赞赏~

赞赏支持



♡ 喜欢 | 33



更多分享

(<http://cwb.assets.jianshu.io/notes/images/1193266>)




写下你的评论...


评论


智慧如你，不想发表一点想法咩~


被以下专题收入，发现更多相似内容


+ 收入我的专题

- 

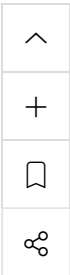
Java学习笔记 (/c/04cb7410c597?utm_source=desktop&utm_medium=notes-included-collection)
- 


程序员 (/c/NEt52a?utm_source=desktop&utm_medium=notes-included-collection)
- 


高性能服务 (/c/8cd53a7673b0?utm_source=desktop&utm_medium=notes-included-collection)
- 

Java · ... (/c/0e8a2277d178?utm_source=desktop&utm_medium=notes-included-collection)
- 

分布式事务 (/c/3046d67088ee?utm_source=desktop&utm_medium=notes-included-collection)



 Zookeeper (/c/057bb7fb1243?utm_source=desktop&utm_medium=notes-included-collection)

 Java技术升华 (/c/d83d00973774?utm_source=desktop&utm_medium=notes-included-collection)

