# Zookeeper基础

## 单基部署

先把ZK安装起来，后面的很多操作，都是的前提都是由ZK的操作环境，先来把ZK安装好，

### Zookeeper windows环境安装

环境要求:必须要有jdk环境,本次讲课使用jdk1.8

1.安装jdk

2.安装Zookeeper. 在官网[http://zookeeper.apache.org/](http://zookeeper.apache.org/" \t "_blank)下载zookeeper.我下载的是zookeeper-3.4.12版本。

解压zookeeper-3.4.6至D:\machine\zookeeper-3.4.12.

在D:\machine 新建data及log目录。

3.ZooKeeper的安装模式分为三种，分别为：单机模式（stand-alone）、集群模式和集群伪分布模式。ZooKeeper 单机模式的安装相对比较简单，如果第一次接触ZooKeeper的话，建议安装ZooKeeper单机模式或者集群伪分布模式。

安装单击模式。 至D:\machine\zookeeper-3.4.12\conf 复制 zoo\_sample.cfg 并粘贴到当前目录下，命名zoo.cfg.

### 目录结构

bin 存放系统脚本

conf 存放配置文件

contrib zk附加功能支持

dist-maven maven仓库文件

docs zk文档

lib 依赖的第三方库

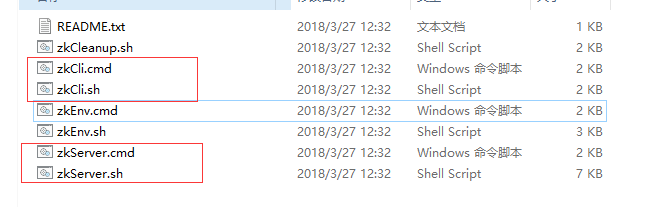
recipes 经典场景样例代码

src zk源码

其中bin和conf是非常重要的两个目录，平时也是经常使用的。

#### bin目录

先看下bin目录



其中

zkServer为服务器，启动后默认端口为2181

zkCli为命令行客户端

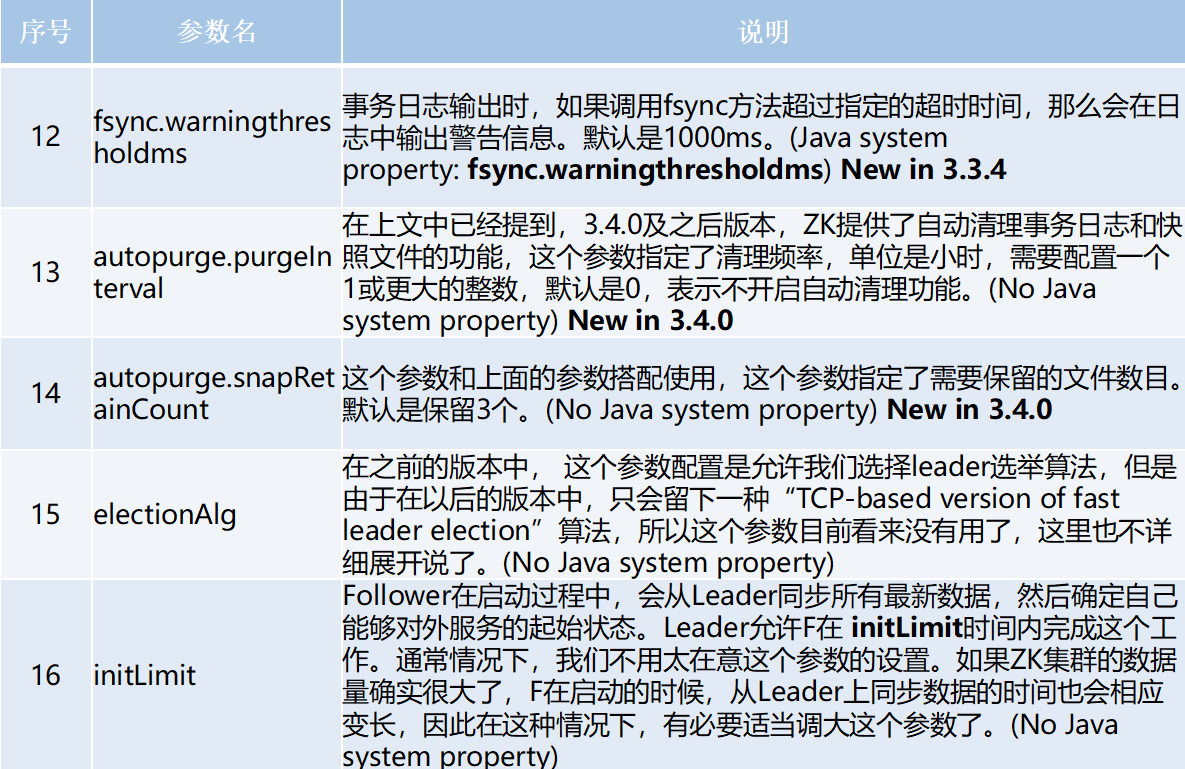
#### conf目录

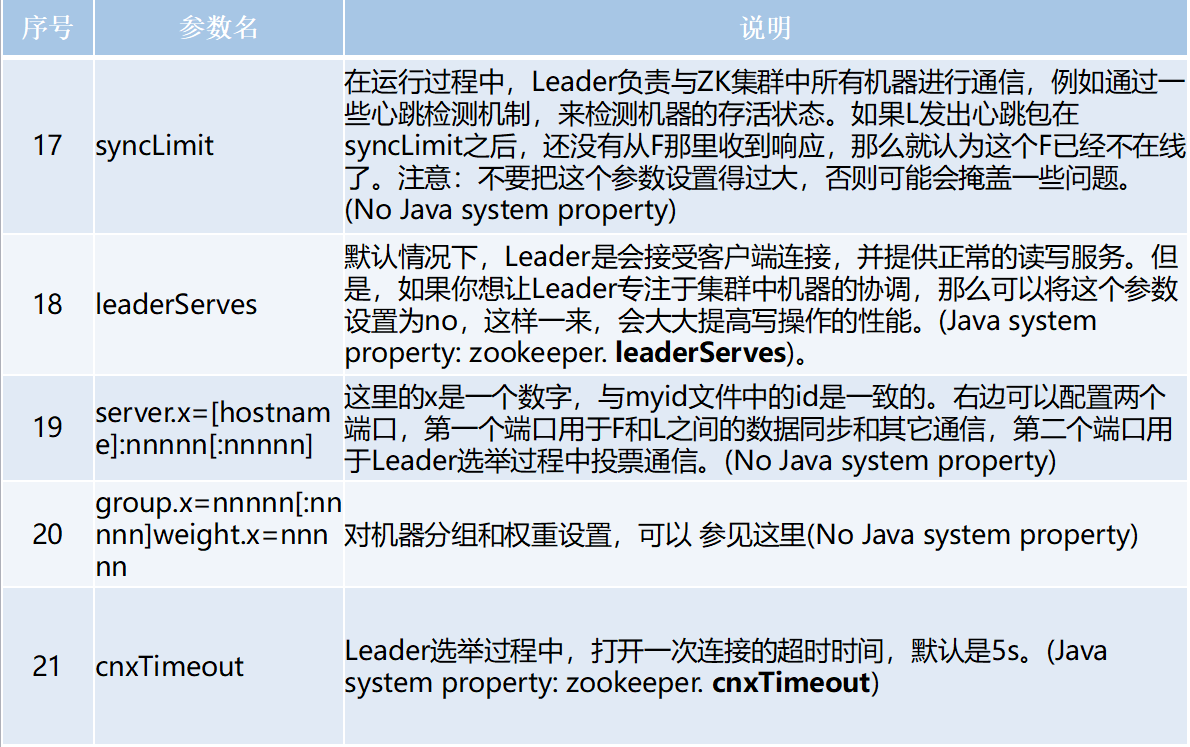
Conf目录为配置文件存放的目录，zoo.cfg为核心的配置文件

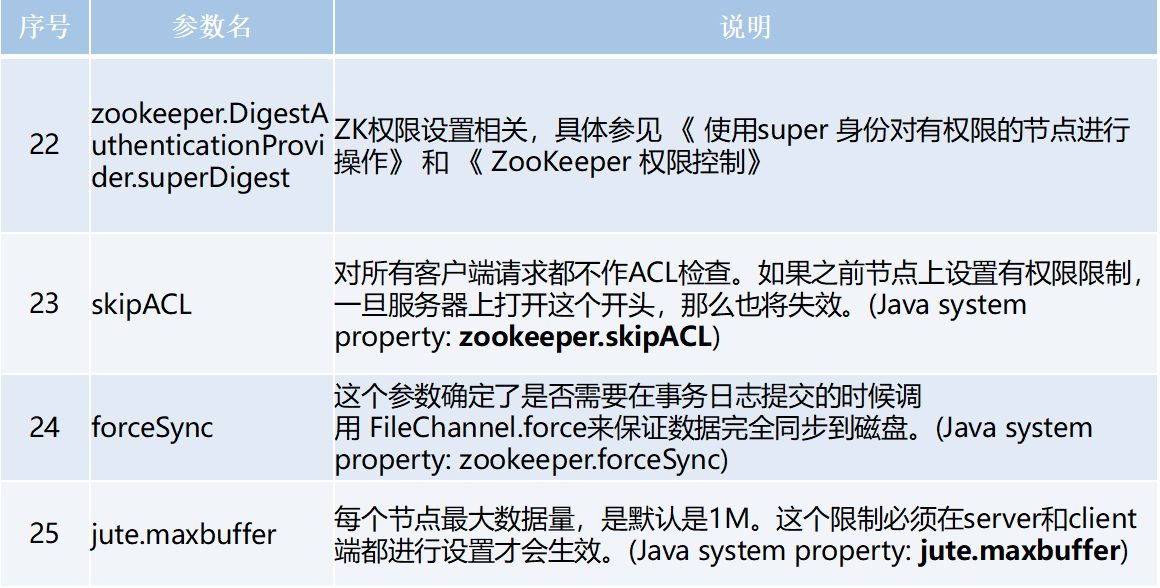
这里面的配置很多，这配置是运维的工作，目前没必要，也没办法全部掌握。











在这挑选几个讲解：

clientPort：参数无默认值，必须配置，用于配置当前服务器对外的服务端口，客户端必须使用这端口才能进行连接

dataDir：用于存放内存数据库快照的文件夹，同时用于集群的myid文件也存在这个文件夹里（注意：一个配置文件只能包含一个dataDir字样，即使它被注释掉了。）

dataLogDir：用于单独设置transaction log的目录，transaction log分离可以避免和普通log还有快照的竞争

dataDir：新安装zk这文件夹里面是没有文件的，可以通过snapCount参数配置产生快照的时机

以下配置集群中才会使用，后面再讨论

tickTime：心跳时间，为了确保连接存在的，以毫秒为单位，最小超时时间为两个心跳时间

initLimit：多少个心跳时间内，允许其他server连接并初始化数据，如果ZooKeeper管理的数据较大，则应相应增大这个值

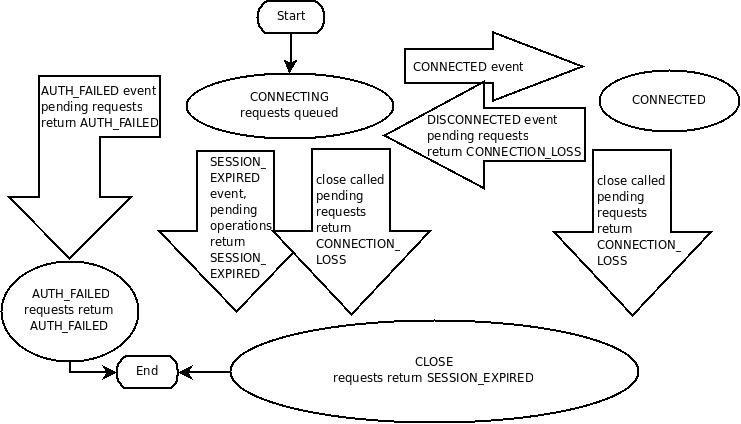
syncLimit：多少个tickTime内，允许follower同步，如果follower落后太多，则会被丢弃。

## ZK的特性

Zk的特性会从会话、数据节点，版本，Watcher，ACL权限控制，集群角色这些部分来了解，其中重点需要掌握的数据节点与Watcher

### 会话

客户端与服务端的一次会话连接，本质是TCP长连接，通过会话可以进行心跳检测和数据传输；



会话（session）是zookepper非常重要的概念，客户端和服务端之间的任何交互操作都与会话有关

**会话状态**

看下这图，Zk客户端和服务端成功连接后，就创建了一次会话，ZK会话在整个运行期间的生命周期中，会在不同的会话状态之间切换，这些状态包括：

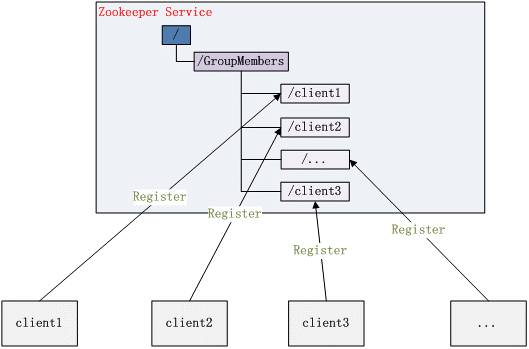
CONNECTING、CONNECTED、RECONNECTING、RECONNECTED、CLOSE

一旦客户端开始创建Zookeeper对象，那么客户端状态就会变成CONNECTING状态，同时客户端开始尝试连接服务端，连接成功后，客户端状态变为CONNECTED，通常情况下，由于断网或其他原因，客户端与服务端之间会出现断开情况，一旦碰到这种情况，Zookeeper客户端会自动进行重连服务，同时客户端状态再次变成CONNCTING，直到重新连上服务端后，状态又变为CONNECTED，在通常情况下，客户端的状态总是介于CONNECTING和CONNECTED之间。但是，如果出现诸如会话超时、权限检查或是客户端主动退出程序等情况，客户端的状态就会直接变更为CLOSE状态

### ZK数据模型

ZooKeeper的视图结构和标准的Unix文件系统类似，其中每个节点称为“数据节点”或ZNode,每个znode可以存储数据，还可以挂载子节点，因此可以称之为“树”

第二点需要注意的是，每一个znode都必须有值，如果没有值，节点是不能创建成功的。



* 在Zookeeper中，znode是一个跟Unix文件系统路径相似的节点，可以往这个节点存储或获取数据
* 通过客户端可对znode进行增删改查的操作，还可以注册watcher监控znode的变化。

### Zookeeper节点类型

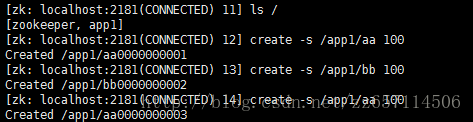
节点类型非常重要，是后面项目实战的基础。

a、Znode有两种类型：

短暂（ephemeral）（create -e /app1/test1 “test1” 客户端断开连接zk删除ephemeral类型节点）   
持久（persistent） （create -s /app1/test2 “test2” 客户端断开连接zk不删除persistent类型节点）

b、Znode有四种形式的目录节点（默认是persistent ）

PERSISTENT   
PERSISTENT\_SEQUENTIAL（持久序列/test0000000019 ）   
EPHEMERAL   
EPHEMERAL\_SEQUENTIAL

c、创建znode时设置顺序标识，znode名称后会附加一个值，顺序号是一个单调递增的计数器，由父节点维护   
       

d、在分布式系统中，顺序号可以被用于为所有的事件进行全局排序，这样客户端可以通过顺序号推断事件的顺序

### Zookeeper节点状态属性



### ACL保障数据的安全

ACL机制，表示为scheme:id:permissions，第一个字段表示采用哪一种机制，第二个id表示用户，permissions表示相关权限（如只读，读写，管理等）。

zookeeper提供了如下几种机制（scheme）：

**world:** 它下面只有一个id, 叫anyone, world:anyone代表任何人，zookeeper中对所有人有权限的结点就是属于world:anyone的

**auth:** 它不需要id, 只要是通过authentication的user都有权限（zookeeper支持通过kerberos来进行authencation, 也支持username/password形式的authentication)

**digest:** 它对应的id为username:BASE64(SHA1(password))，它需要先通过username:password形式的authentication

**ip:** 它对应的id为客户机的IP地址，设置的时候可以设置一个ip段，比如ip:192.168.1.0/16, 表示匹配前16个bit的IP段

现在看这可能懵懵懂懂，不过没有关系，等会在客户端操作的时候会有详细的操作

## 命令行

### 服务端常用命令

在准备好相应的配置之后，可以直接通过zkServer.sh 这个脚本进行服务的相关操作

启动ZK服务:       sh bin/zkServer.sh start 

查看ZK服务状态: sh bin/zkServer.sh status 

停止ZK服务:       sh bin/zkServer.sh stop 

重启ZK服务:       sh bin/zkServer.sh restart

### 客户端常用命令

使用 zkCli.sh -server 127.0.0.1:2181 连接到 ZooKeeper 服务，连接成功后，系统会输出 ZooKeeper 的相关环境以及配置信息。 命令行工具的一些简单操作如下：

* 显示根目录下、文件： ls / 使用 ls 命令来查看当前 ZooKeeper 中所包含的内容
* 显示根目录下、文件： ls2 / 查看当前节点数据并能看到更新次数等数据
* 创建文件，并设置初始内容： create /zk "test" 创建一个新的 znode节点“ zk ”以及与它关联的字符串  [-e] [-s] 【-e 零时节点】 【-s 顺序节点】
* 获取文件内容： get /zk 确认 znode 是否包含我们所创建的字符串  [watch]【watch 监听】
* 修改文件内容： set /zk "zkbak" 对 zk 所关联的字符串进行设置 
* 删除文件： delete /zk 将刚才创建的 znode 删除，如果存在子节点删除失败
* 递归删除：rmr /zk将刚才创建的 znode 删除，子节点同时删除
* 退出客户端： quit 
* 帮助命令： help

### ACL命令常用命令

再回过头来看下ACL权限

Zookeeper的ACL(Access Control List)，分为三个维度：scheme、id、permission

通常表示为：scheme:id:permission

* schema:代表授权策略
* id:代表用户
* permission:代表权限

#### Scheme

world：

默认方式，相当于全世界都能访问

auth：

代表已经认证通过的用户(可以通过addauth digest user:pwd 来添加授权用户)

digest：

即用户名:密码这种方式认证，这也是业务系统中最常用的

ip：

使用Ip地址认证

#### id

id是验证模式，不同的scheme，id的值也不一样。

scheme为auth时：

username:password

scheme为digest时:  
 username:BASE64(SHA1(password))

scheme为ip时:  
 客户端的ip地址。

scheme为world时  
 anyone。

#### Permission

CREATE、READ、WRITE、DELETE、ADMIN 也就是 增、删、改、查、管理权限，这5种权限简写为crwda(即：每个单词的首字符缩写)

CREATE(c)：创建子节点的权限

DELETE(d)：删除节点的权限

READ(r)：读取节点数据的权限

WRITE(w)：修改节点数据的权限

ADMIN(a)：设置子节点权限的权限

#### ACL命令

getAcl

获取指定节点的ACL信息

create /testDir/testAcl deer # 创建一个子节点

getAcl /testDir/testAcl # 获取该节点的acl权限信息

setAcl

设置指定节点的ACL信息

setAcl /testDir/testAcl world:anyone:crwa # 设置该节点的acl权限

getAcl /testDir/testAcl # 获取该节点的acl权限信息，成功后，该节点就少了d权限

create /testDir/testAcl/xyz xyz-data # 创建子节点

delete /testDir/testAcl/xyz # 由于没有d权限，所以提示无法删除

addauth

注册会话授权信息

Auth

addauth digest user1:123456 # 需要先添加一个用户

setAcl /testDir/testAcl auth:user1:123456:crwa # 然后才可以拿着这个用户去设置权限

getAcl /testDir/testAcl # 密码是以密文的形式存储的

create /testDir/testAcl/testa aaa

delete /testDir/testAcl/testa # 由于没有d权限，所以提示无法删除

退出客户端后：

ls /testDir/testAcl #没有权限无法访问

create /testDir/testAcl/testb bbb #没有权限无法访问

addauth digest user1:123456 # 重新新增权限后可以访问了

Digest

auth与digest的区别就是，前者使用明文密码进行登录，后者使用密文密码进行登录

create /testDir/testDigest data

addauth digest user1:123456

setAcl /testDir/testDigest digest:user1:HYGa7IZRm2PUBFiFFu8xY2pPP/s=:crwa # 使用digest来设置权限

注意：这里如果使用明文，会导致该znode不可访问

**通过明文获得密文**

shell>

java -Djava.ext.dirs=/soft/zookeeper-3.4.12/lib -cp /soft/zookeeper-3.4.12/zookeeper-3.4.12.jar org.apache.zookeeper.server.auth.DigestAuthenticationProvider deer:123456

deer:123456->deer:ACFm5rWnnKn9K9RN/Oc8qEYGYDs=

acl命令行ip

create /testDir/testIp data

setAcl /testDir/testIp ip:192.168.30.10:cdrwa

getAcl /testDir/testIp

### 常用四字命令

ZooKeeper 支持某些特定的四字命令字母与其的交互。用来获取 ZooKeeper 服务的当前状态及相关信息。可通过 telnet 或 nc 向 ZooKeeper 提交相应的命令 ：

当然，前提是安装好了nc

echo stat|nc 127.0.0.1 2181 来查看哪个节点被选择作为follower或者leader 

使用echo ruok|nc 127.0.0.1 2181 测试是否启动了该Server，若回复imok表示已经启动。 

echo dump| nc 127.0.0.1 2181 ,列出未经处理的会话和临时节点。 

echo kill | nc 127.0.0.1 2181 ,关掉server 

echo conf | nc 127.0.0.1 2181 ,输出相关服务配置的详细信息。 

echo cons | nc 127.0.0.1 2181 ,列出所有连接到服务器的客户端的完全的连接 / 会话的详细信息 

echo envi |nc 127.0.0.1 2181 ,输出关于服务环境的详细信息（区别于 conf 命令）。 

echo reqs | nc 127.0.0.1 2181 ,列出未经处理的请求。 

echo wchs | nc 127.0.0.1 2181 ,列出服务器 watch 的详细信息。 

echo wchc | nc 127.0.0.1 2181 ,通过 session 列出服务器 watch 的详细信息，它的输出是一个与 watch 相关的会话的列表。 

echo wchp | nc 127.0.0.1 2181 ,通过路径列出服务器 watch 的详细信息。它输出一个与 session 相关的路径。

### ZooKeeper 日志可视化

前面以及讲了两个非常重要的配置一个是dataDir，存放的快照数据，一个是dataLogDir，存放的是事务日志文件

java -cp /soft/zookeeper-3.4.12/zookeeper-3.4.12.jar:/soft/zookeeper-3.4.12/lib/slf4j-api-1.7.25.jar org.apache.zookeeper.server.LogFormatter log.1

java -cp /soft/zookeeper-3.4.12/zookeeper-3.4.12.jar:/soft/zookeeper-3.4.12/lib/slf4j-api-1.7.25.jar org.apache.zookeeper.server.SnapshotFormatter log.1

## Java客户端框架（\*重要）

### Zookeeper原生客户端

### ZkClient

### Curator