zookeeper

1.zookeeper工作机制

观察者模式设计模式的分布式管理框架，负责存储管理大家都关心的数据；接受观察者的注册，一旦data发生变化，通知所有的观察者

2.zookeeper的特点

1. 一个领导者，多个跟随者；
2. 当zookeeper集群有半数以上的节点启动，zookeeper才可以正常工作
3. zookeeper遵循CAP理论的cp理论，数据一致性
4. 我们安装zookeeper一般安装奇数台

3.zookeeper的数据结构

树装结构，类似于Linux的文件夹　zookeeper的根节点就是zookeeper 每个节点上最多存储1MB的数据

4.zookeeper的安装

官网--->下载---->解压--->修改配置(zoo.cfg)---->bin目录的理解

zoo.cfg:

ticktime:zookeeper服务端和客户端的心跳时间

inittime:zookeeper集群的leader与follow的心跳时间

synclimit:zookeeper在leader节点宕机之后与follow与follow之间的心跳时间

dataDir:数据持久化

clientPort：端口

zookeeper集群搭建

zoo.cfg:配置　A:B:C:D

A：代表主机的myid　B:代表主机名称　C:代表leader和follower的通信端口　D:选举用的通信端口

5.面试重点：

- zookeeper选举机制：第一次启动选举

  假若有五个服务器搭建的集群，myid为1,2,3,5,如果第一台服务器启动上线，他会先给自己投一票，然后进行自己的投票数与服务器半数以上的值进行对比，判断如果1<3，在那等着，第二台服务器上线了，先给自己投一票，然后将票数互投，互投有个原则投票给myid比较大的一方，所以将票投给了myid为2的服务器，2<3 ，同样第三台上线 3=3所以服务器集群启动选举出myid=3的为leader，后面的服务器无论myid有多大，都是follower因为已经有了领导

- zookeeper选举机制：非启动选举

  重新选举一般分为两种方式：

  遇到的情况

1. 其中某一台服务器与其他服务器连接不了，它还不认为自己挂掉了，认为别人挂掉了
2. 真的领导者挂掉了，要重新进行选举

非启动选举：

三个参数

- sid 也就是每台服务器的 myid
- zxid每台服务器被请求的事务id
- EPoch服务器的领导任期代号

选举方式：先通过对EPOCH的值比较，如果相同在比较zxid，如果相同在比较sid。

6.zookeeper客户端命令字操作

- ls
- ls –s　当前节点
- create /sanguo "chaoguo"　创建节点 根节点下面
- create /sanguo/shuguo "liubei"
- ls /sanguo　　查看/sanguo下的节点
- get –s /sanguo　获取节点的值
- delete /sanguo　deleteAll /sanguo

1. –s代表　创建是否带序号的节点
2. –e 代表　创建临时节点（临时还是持久）

节点类型：

持久（临时-e）

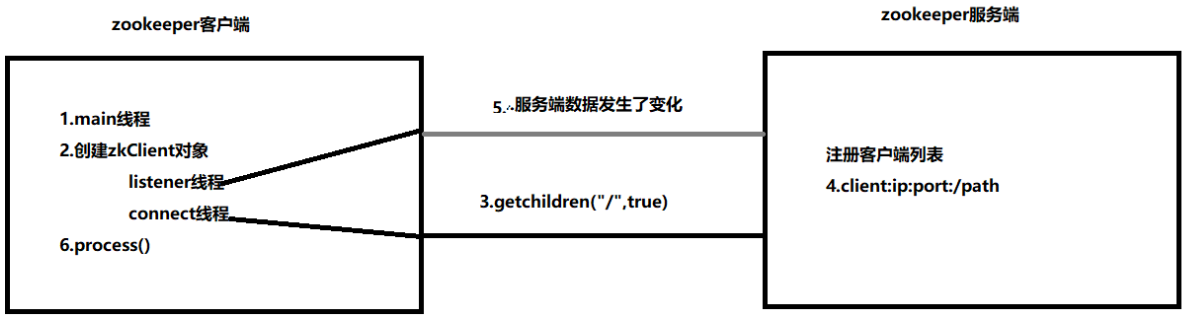
序号（带序号-s）

我们一般创建的都是带序号的临时节点

监听命令：get -w /sanguo 监听节点值的变化；ls -w /sanguo 监听节点个数的变化

注意：每次监听节点只生效一次，若要重复监听，需要写循环

7.zookeeper监听器的原理



8.通过idea的方式操作zookeeper

1. 导入pom依赖：

org.apache.zookeeper 版本号必须一致

2. new Zookeeper("ip:port",超时参数，监听器)
3. 看文档，学习api

9.客户端向服务端写数据流程

- 客户端直接向leader服务器进行写数据

  假如有三个节点：client向领导者发送写write,leader自己先write,在通知其他跟随着进行同步，如果同步的节点数超过半数以上节点数，先让领导者响应客户端，响应完成后，领导者leader在通知其他follower进行同步

- 客户端向follower服务器进行写数据

  假如有三个节点：client向follower发送write,follower将请求转发给leader,leader自己先write,在通知其他跟随着进行同步，如果同步的节点数超过半数以上节点数，领导者leader让follower响应客户端，客户端响应完成后，领导者leader在通知其他follower进行同步。

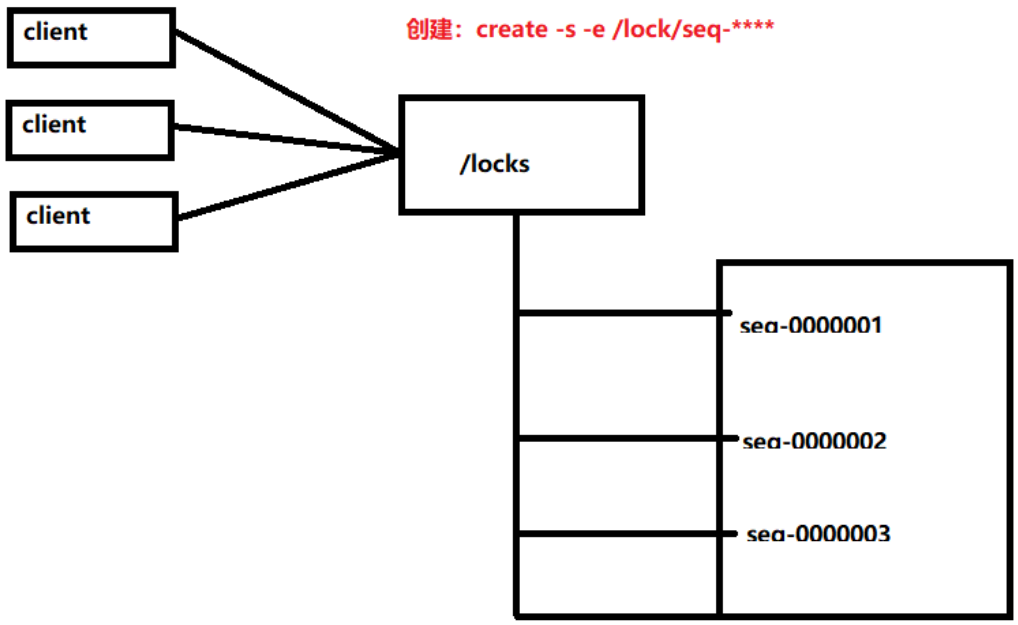
10.zookeeper在注册中心的应用

zookeeper的服务端作为注册中心，这个时候他的原理遵循cp理论，数据一致性协议，这个时候就要注意zoo.cfg的配置如：ticktime,inittime,syncltime

例：有消费者，和生产者两个微服务，生产者为消费者提供服务端。

对于zookeeper而言，生产者和消费者都是zookeeper的客户端，消费者服务启动，就会去zookeeper中注册一个节点，客户端去监听这个节点，的变化ls -w /eurka,这样一旦zookeeper集群的节点发生了变化，就直接通知客户端，客户端也就接收到了消息。

11.zookeeper分布式锁案例



说明：

1. zookeeper接收到集群后，在locks节点下创建一个临时顺序节点
2. 判断自己是不是当前节点中最小的节点，如果是获取到锁，如果不是，对前一个节点进行监听
3. 获取到锁，处理完业务后，delete节点释放锁，然后下面的节点将接收到通知，重复循环
4. 以上三点是分布式锁的一个原理，我们用成熟的分布式锁框架Curator