

小不点啊

结局很美妙的事,开头并非如此!

转载和引用,请注明原文出处!
me on GitHub

Fork

博客园 首页 新随笔 联系 订阅 管理

随笔 - 187 文章 - 0 评论 - 94 阅读 - 100万

昵称： 小不点啊
园龄： 4年6个月
粉丝： 382
关注： 4
+加关注

< 2022年3月 >

日	一	二	三	四	五	六
27	28	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31	1	2
3	4	5	6	7	8	9

搜索

常用链接

我的随笔

我的评论

我的参与

最新评论

我的标签

我的标签

dubbo(6)

jvm(4)

Spring IoC(2)

分布式文件系统(2)

传统服务拆分(2)

原子操作(1)

Spring MVC与Struts2的对比(1)

Spring MVC的运行流程(1)

异常处理(1)

自定义拦截器(1)

Zookeeper系列一：Zookeeper介绍、Zookeeper安装配置、ZK Shell的使用

一、Zookeeper介绍

1. 介绍Zookeeper之前先来介绍一下分布式

1.1 分布式主要是下面两个方面：

1) 任务拆分

任务拆分指的是把传统的单节点服务拆分成多个节点服务部署到不同的机器上对外提供服务。比如一个传统服务有订单+支付+物流等3个模块，拆分成订单系统、支付系统、物流系统3个服务。

2) 节点分工

如上面的服务拆分后，订单系统、支付系统、物流系统各司其职

说明：

分布式解决高可用，高并发的。

集群解决的是高可用。

集群从物理上来定义，分布式一种工作方式。
例如：一个工作任务需要10个小时（单节点）
分布式：10台机器，任务只需要1个小时就能够完成
集群：10台机器，任务还是10个小时。

1.2 分布式协作中的难点：

如果让你设计一个分布式系统，你预见到什么问题？

- 1) 保证节点高可用（节点故障）
- 2) 数据的一致性
- 3) 通讯异常
- 4) 网络分区
-

2. Zookeeper简介

Zookeeper就是用来解决分布式协作中的难点的

zookeeper是google的chubby项目开源实现。最早是hadoop的子项目

Zookeeper的使用场景：

小米米聊、淘宝Taookeeper其实是类zookeeper。

Kafka使用zookeeper。消息发布订阅，其中zk就是用于检测节点崩溃。实现主题的发现，并且保持主题的生产和消费状态。

Hbase三段查找，Root-Region=》Meta Region=》Region（Table）。hbase的元数据信息放在HBase。HMaster挂掉，马上要节点恢复。
Hadoop。NameNode（SecondaryNameNode），HA Hadoop。一般情况下一个简单的hadoop集群，只有一个NameNode，如果NameNode挂掉，hadoop集群不可用。HA Hadoop里面就要用到zk。

3. Zookeeper解决哪些问题

3.1 Master节点管理

解决的问题：Master高可用（挂掉以后，谁来负责工作），保证唯一。

3.2 配置文件管理

https://www.cnblogs.com/leeSmall/p/9563547.html

1/10

更多

随笔分类
BIO、NIO、AIO(2)
Docker(1)
dubbo(2)
Interview(10)
java(6)
java基础(8)
jvm(4)
Linux(1)
mybatis(6)
mysql(9)
nginx(2)
redis(10)
spring(4)
Spring MVC(16)
SpringBoot(12)
更多

随笔档案
2020年11月(10)
2019年3月(11)
2019年1月(1)
2018年10月(2)
2018年9月(11)
2018年8月(11)
2018年7月(11)
2018年6月(13)
2018年5月(11)
2018年4月(22)
2018年3月(9)
2018年2月(4)
2018年1月(11)

解决的问题：统一把配置文件存放zk，由ZK统一分发修改的内容到各台机器。

3.3 发布与订阅

发布者(producer)将数据发布到zk节点上，供订阅者(consumer)动态获取

3.4 分布式锁

分布式环境访问同一个资源，由第三方配锁实现。
解决的问题：由zk统一进行协调，保证数据的一致性。

3.5 集群的管理

Worker集群监控。保证主数据和备份数据的一致

二、Zookeeper安装配置

环境准备：
一台安装有jdk的虚拟机：192.168.152.130

1. 安装

1.1.下载

```
cd /software
wget http://mirror.bit.edu.cn/apache/zookeeper/zookeeper-3.4.6/zookeeper-3.4.6.tar.gz
```

1.2.解压

```
tar -zxvf zookeeper-3.4.6.tar.gz
```

2. 配置

说明：这里配置的是伪分布式的zookeeper
注意：配置之前一定要在/etc/hosts里面配置主机映射,否则会报错提示：

ERROR [master:3890:QuorumCnxManager\$Listener@517] - Exception while listening
java.net.SocketException: Unresolved address

```
vim /etc/hosts

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
127.0.0.1 master
```

2.1 先建立zookeeper的三个数据目录

```
mkdir -p /zookeeper/zk1
mkdir -p /zookeeper/zk2
mkdir -p /zookeeper/zk3
```

```
[root@root zookeeper]# ll
总用量 12
drwxr-xr-x. 2 root root 4096 8月 27 08:35 zk1
drwxr-xr-x. 2 root root 4096 8月 27 08:35 zk2
drwxr-xr-x. 2 root root 4096 8月 27 08:35 zk3
```

2.2 查看zookeeper的配置文件（已删掉多余的配置和注释）

```
vim zoo_sample.cfg

tickTime=2000 #session的会话时间 以ms为单位
initLimit=10 #服务器启动以后，master和slave通讯的时间
syncLimit=5 #master和slave之间的心跳检测时间，检测slave是否存活
dataDir=/tmp/zookeeper #保存zk的快照和数据
clientPort=2181 #客户端访问zk的端口
```

2.3 复制zoo_sample.cfg为zoo1.cfg添加如下配置

```
cp zoo_sample.cfg zoo1.cfg
vim zoo1.cfg
```

2017年12月(1)
2017年11月(19)
更多

阅读排行榜

1. elasticsearch系列七：ES Java客户端-Elasticsearch Java client（ES Client 简介、Java REST Client、Java Client、Spring Data Elasticsearch）(74402)
2. elasticsearch系列八：ES 集群管理（集群规划、集群搭建、集群管理）(66799)
3. elasticsearch系列六：聚合分析（聚合分析简介、指标聚合、桶聚合）(65238)
4. SpringCloud系列七：Hystrix 熔断机制（Hystrix基本配置、服务降级、HystrixDashboard服务监控、Turbine聚合监控）(42183)
5. Redis系列八：redis主从复制和哨兵(35995)

评论排行榜

1. Nginx系列二：（Nginx Rewrite 规则、Nginx 防盗链、Nginx 动静分离、Nginx+keepalived 实现高可用）(8)
2. SpringCloud系列二：Restful 基础架构（搭建项目环境、创建 Dept 微服务、客户端调用微服务）(5)
3. Redis系列十：缓存雪崩、缓存穿透、缓存预热、缓存更新、缓存降级(5)
4. 框架源码系列三：手写Spring AOP（AOP分析、AOP概念学习、切面实现、织入实现）(4)
5. elasticsearch系列七：ES Java客户端-Elasticsearch Java client（ES Client 简介、Java REST Client、Java Client、Spring Data Elasticsearch）(4)

推荐排行榜

1. elasticsearch系列七：ES Java客户端-Elasticsearch Java client（ES Client 简介、Java REST Client、Java Client、Spring Data Elasticsearch）(14)
2. Nginx系列二：（Nginx Rewrite 规则、Nginx 防盗链、Nginx 动静分离、Nginx+keepalived 实现高可用）(8)
3. elasticsearch系列八：ES 集群管理（集群规划、集群搭建、集群管理）(8)

配置：

```
#session的会话时间 以ms为单位
tickTime=2000

#服务器启动以后，master和slave通讯的时间
initLimit=10

#master和slave之间的心跳检测时间，检测slave是否存活
syncLimit=5

#（这个目录可以自行指定）
dataDir=/zookeeper/zk1

#客户端访问zk的端口
clientPort=2181
#master对应于前面在hosts里面配置的主机映射 2888是数据同步和消息传递端口，3888是选举端口
server.1=master:2888:3888

#master对应于前面在hosts里面配置的主机映射 2889是数据同步和消息传递端口，3889是选举端口
server.2=master:2889:3889

#master对应于前面在hosts里面配置的主机映射 2890是数据同步和消息传递端口，3890是选举端口
server.3=master:2890:3890
```

2.4 复制zoo1.cfg为zoo2.cfg添加如下配置

```
cp zoo1.cfg zoo2.cfg
vim zoo2.cfg
```

```
#session的会话时间 以ms为单位
tickTime=2000

#服务器启动以后，master和slave通讯的时间
initLimit=10

#master和slave之间的心跳检测时间，检测slave是否存活
syncLimit=5

#（这个目录可以自行指定）
dataDir=/zookeeper/zk2

#客户端访问zk的端口
clientPort=2182

#master对应于前面在hosts里面配置的主机映射 2888是数据同步和消息传递端口，3888是选举端口
server.1=master:2888:3888

#master对应于前面在hosts里面配置的主机映射 2889是数据同步和消息传递端口，3889是选举端口
server.2=master:2889:3889

#master对应于前面在hosts里面配置的主机映射 2890是数据同步和消息传递端口，3890是选举端口
server.3=master:2890:3890
```

2.5 复制zoo1.cfg为zoo3.cfg添加如下配置

```
cp zoo1.cfg zoo3.cfg
vim zoo3.cfg
```

```
#session的会话时间 以ms为单位
tickTime=2000

#服务器启动以后，master和slave通讯的时间
initLimit=10

#master和slave之间的心跳检测时间，检测slave是否存活
syncLimit=5
```

4. Zookeeper系列二：分布式架构详解、分布式技术详解、分布式事务(6)
5. elasticsearch系列六：聚合分析（聚合分析简介、指标聚合、桶聚合）(6)
最新评论
1. Re:SpringBoot系列十一：SpringBoot整合Restful架构（使用 RestTemplate 模版实现 Rest 服务调用、Swagger 集成、动态修改日志级别）
啊
--规格严格-功夫到家
2. Re:SpringBoot系列四：SpringBoot开发（改变环境属性、读取资源文件、Bean 配置、模版渲染、profile 配置）
6666
--规格严格-功夫到家
3. Re:SpringBoot系列三：SpringBoot基本概念（统一父 pom 管理、SpringBoot 代码测试、启动注解分析、配置访问路径、使用内置对象、项目打包发布）
很棒。
--规格严格-功夫到家
4. Re:SpringBoot系列七：SpringBoot 整合 MyBatis（配置 druid 数据源、配置 MyBatis、事务控制、druid 监控）
h很好。
--规格严格-功夫到家
5. Re:框架源码系列三：手写Spring AOP（AOP分析、AOP概念学习、切面实现、织入实现）
你少复制了一段，简直就是误人子弟，这个比你的全
--MisterXie

```
# (这个目录可以自行指定)
dataDir=/zookeeper/zk3

#客户端访问zk的端口
clientPort=2183
#master对应于前面在hosts里面配置的主机映射 2888是数据同步和消息传递端口，3888是选举端口
server.1=master:2888:3888

#master对应于前面在hosts里面配置的主机映射 2889是数据同步和消息传递端口，3889是选举端口
server.2=master:2889:3889

#master对应于前面在hosts里面配置的主机映射 2890是数据同步和消息传递端口，3890是选举端口
server.3=master:2890:3890
```

2.6 手动建立myid文件且指定在zk数据目录，也就是dataDir指定的路径（不管真分布还是伪分布都需要指定）

```
echo 1 >> /zookeeper/zk1/myid
echo 2 >> /zookeeper/zk2/myid
echo 3 >> /zookeeper/zk3/myid
```

说明：
myid文件内容分别为1,2,3对应于zk配置文件的server.1,server.2,server.3

2.7 测试zookeeper分布式是否搭建成功

分别启动三个zookeeper

```
cd /software/zookeeper-3.4.6/bin/
./zkServer.sh start /software/zookeeper-3.4.6/conf/zoo1.cfg
./zkServer.sh start /software/zookeeper-3.4.6/conf/zoo2.cfg
./zkServer.sh start /software/zookeeper-3.4.6/conf/zoo3.cfg
```

```
[root@root bin]# ./zkServer.sh start /software/zookeeper-3.4.6/conf/zoo1.cfg
JMX enabled by default
Using config: /software/zookeeper-3.4.6/conf/zoo1.cfg
Starting zookeeper ... STARTED
[root@root bin]# ./zkServer.sh start /software/zookeeper-3.4.6/conf/zoo2.cfg
JMX enabled by default
Using config: /software/zookeeper-3.4.6/conf/zoo2.cfg
Starting zookeeper ... STARTED
[root@root bin]# ./zkServer.sh start /software/zookeeper-3.4.6/conf/zoo3.cfg
JMX enabled by default
Using config: /software/zookeeper-3.4.6/conf/zoo3.cfg
Starting zookeeper ... STARTED
```

查看3个zookeeper的状态

```
cd /software/zookeeper-3.4.6/bin/
./zkServer.sh status /software/zookeeper-3.4.6/conf/zoo1.cfg
./zkServer.sh status /software/zookeeper-3.4.6/conf/zoo2.cfg
./zkServer.sh status /software/zookeeper-3.4.6/conf/zoo3.cfg
```

```
[root@root bin]# ./zkServer.sh status /software/zookeeper-3.4.6/conf/zoo1.cfg
JMX enabled by default
Using config: /software/zookeeper-3.4.6/conf/zoo1.cfg
Mode: follower
[root@root bin]# ./zkServer.sh status /software/zookeeper-3.4.6/conf/zoo2.cfg
JMX enabled by default
Using config: /software/zookeeper-3.4.6/conf/zoo2.cfg
Mode: leader
[root@root bin]# ./zkServer.sh status /software/zookeeper-3.4.6/conf/zoo3.cfg
JMX enabled by default
Using config: /software/zookeeper-3.4.6/conf/zoo3.cfg
Mode: follower
[root@root bin]#
```

可以看到第2个zookeeper是leader，第1个和第3个为follower，到此zookeeper分布式搭建完成！！

2.8 Zookeeper常用命令

```
启动：
./zkServer.sh start /software/zookeeper-3.4.6/conf/zoo1.cfg
./zkServer.sh start /software/zookeeper-3.4.6/conf/zoo2.cfg
./zkServer.sh start /software/zookeeper-3.4.6/conf/zoo3.cfg
```

```
停止:
./zkServer.sh stop /software/zookeeper-3.4.6/conf/zoo1.cfg
./zkServer.sh stop /software/zookeeper-3.4.6/conf/zoo2.cfg
./zkServer.sh stop /software/zookeeper-3.4.6/conf/zoo3.cfg

查看状态:
./zkServer.sh status /software/zookeeper-3.4.6/conf/zoo1.cfg
./zkServer.sh status /software/zookeeper-3.4.6/conf/zoo2.cfg
./zkServer.sh status /software/zookeeper-3.4.6/conf/zoo3.cfg
```

2.9 Zookeeper真分布式部署在master、slave1、slave2 三台不同的服务器上
server1....conf/zoo.cfg

```
tickTime=2000
initLimit=10
syncLimit=5
dataDir=/zookeeper/zk
clientPort=2181

server.1=master:2888:3888
server.2=slave1:2888:3888
server.3=slave2:2888:3888
```

server2....conf/zoo.cfg

```
tickTime=2000
initLimit=10
syncLimit=5
dataDir=/zookeeper/zk
clientPort=2181

server.1=master:2888:3888
server.2=slave1:2888:3888
server.3=slave2:2888:3888
```

server3....conf/zoo.cfg

```
tickTime=2000
initLimit=10
syncLimit=5
dataDir=/zookeeper/zk
clientPort=2181

server.1=master:2888:3888
server.2=slave1:2888:3888
server.3=slave2:2888:3888
```

注意:

真分布必须是奇数台，因为必须满足 $n/2 + 1 = m$, m 必须大于部署zk机器数的一半($n/2$)可用才认为集群可用，而且奇数台更节省资源

$2/2 + 1 = 2$ 一台不能坏

$3/2 + 1 = 2$ 可以坏掉一台

$4/2 + 1 = 3$ 只能坏一台

$5/2 + 1 = 3$ 可以坏2台

三、ZK Shell的使用

1. 连接zk

```
zkCli.sh [-server ip:port]
```

示例：

连接到master (127.0.0.1 master) 这台机器上2181的zookeeper

```
./zkCli.sh -server master 2181
```

```
2018-08-29 16:43:30,403 [myid:] - INFO [main-SendThread(local
2018-08-29 16:43:30,568 [myid:] - INFO [main-SendThread(local
WATCHER::
WatchedEvent state:SyncConnected type:None path:null
[zk: master:2181(CONNECTED) 0]
```

2. 列出zk中的节点

```
ls, ls2
```

示例：

```
[zk: master:2181(CONNECTED) 1] ls /
[zk-test, zookeeper]
[zk: master:2181(CONNECTED) 2] ls2 /
[zk-test, zookeeper]
cZxid = 0x0
ctime = Thu Jan 01 08:00:00 CST 1970
mZxid = 0x0
mtime = Thu Jan 01 08:00:00 CST 1970
pZxid = 0x100000005
cversion = 0
dataVersion = 0
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 0
numChildren = 2
[zk: master:2181(CONNECTED) 3]
```

3. 创建节点

```
create [-s] [-e] path data acl
```

说明：

zk的节点分为两种：临时节点（随着zk session消亡而自动删除）、持久节点（一直存在）

-s: 顺序节点

-e: 临时节点

acl: 访问权限控制

示例：

3.1 创建一个zk-test的节点，数据是123 不带-s、-e默认是持久节点

```
create /zk-test "123"
```

```
[zk: master:2181(CONNECTED) 5] create /zk-test "123"
Created /zk-test
[zk: master:2181(CONNECTED) 6] ls /
[zk-test, zookeeper]
[zk: master:2181(CONNECTED) 7]
```

3.2 创建顺序节点

```
create -s /zk-test "test123"
```

```
[zk: master:2181(CONNECTED) 7] create -s /zk-test "test123"
Created /zk-test0000000002
[zk: master:2181(CONNECTED) 8] ls /
[zk-test, zookeeper, zk-test0000000002]
[zk: master:2181(CONNECTED) 9]
```

多次执行创建顺序节点命令，可以看到zookeeper的内部会对顺序节点的名称进行自增控制

```
[zk: master:2181(CONNECTED) 9] create -s /zk-test "test123"
Created /zk-test0000000003
[zk: master:2181(CONNECTED) 10] create -s /zk-test "test123"
Created /zk-test0000000004
[zk: master:2181(CONNECTED) 11] create -s /zk-test "test123"
Created /zk-test0000000005
[zk: master:2181(CONNECTED) 12]
```

3.3 创建临时节点

```
create -e /zk-test123 "test1234"
```

```
zk: master:2181(CONNECTED) 14] create -e /zk-test123 "test1234"
Created /zk-test123
zk: master:2181(CONNECTED) 15] ls /
zk-test, zookeeper, zk-test123, zk-test0000000005, zk-test0000000004, zk-test0000000003, zk-test0000000002]
zk: master:2181(CONNECTED) 16]
```

说明:

临时节点依赖于顺序节点，临时节点下不能再创建临时节点，顺序节点下才可以创建临时节点

3.4 创建子节点test

```
create /zk-test ""
create /zk-test/test "1"
ls /zk-test/test
```

```
[zk: master:2181(CONNECTED) 6] create /zk-test ""
Created /zk-test
[zk: master:2181(CONNECTED) 7] create /zk-test/test "1"
Created /zk-test/test
[zk: master:2181(CONNECTED) 8] ls /zk-test/test
[]
[zk: master:2181(CONNECTED) 9]
```

4. 删除节点命令

```
delete
```

示例:

删除节点zk-test，如果zk-test下面还有子节点得先删除子节点，才能删除zk-test

```
delete /zk-test/test
delete /zk-test
```

递归删除一个znode

```
rmr path
```

示例:

删除zk-test及其下面的子节点

rmr /zk-test

5. 获取节点信息

```
get
```

示例:

```
create /zk-123 "abc"
get /zk-123
```

```
[zk: master:2181(CONNECTED) 13] create /zk-123 "abc"
Created /zk-123
[zk: master:2181(CONNECTED) 14] get /zk-123
"abc"
cZxid = 0x10000001b
ctime = Wed Aug 29 17:21:45 CST 2018
mZxid = 0x10000001b
mtime = Wed Aug 29 17:21:45 CST 2018
pZxid = 0x10000001b
cversion = 0
dataVersion = 0
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 9
numChildren = 0
[zk: master:2181(CONNECTED) 15]
```

结果说明:



结果

cZxid = #创建节点时zk内部自己分配的id

Ctime = #创建节点的时间

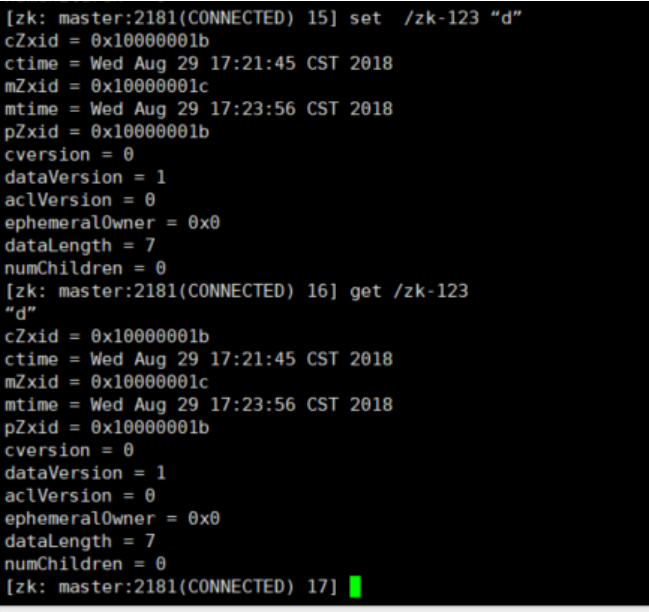
mZxid = #修改的id

mtime = 修改的时间
pZxid = 子节点最后一次被修改的id
cVersion = 0 #拥有的子节点被改的话，该值随着改变
dataVersion = 0 #数据版本
aclVersion = 0 # 访问控制权限的版本
ephemeralOwner = 0x0 #临时节点还是持久节点 临时节点值不为0（值为当前会话id），持久节点值永远为0
dataLength = 3 #数据长度
numChildren = 0 #子节点个数



6. 更新节点数据

set /zk-123 "d"



7. 连接到指定节点

connect host:port

示例：



连接到2181
connect master:2181

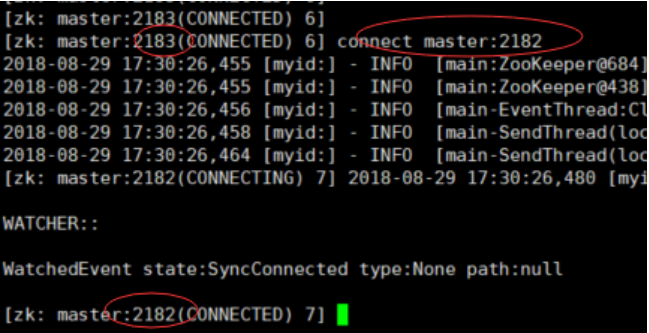
连接到2182
connect master:2182

连接到2183
connect master:2183



说明：

这个命令是在已经连接到zookeeper之后，在里面切换到其他zookeeper时使用



8. 设置配额

配额：给某个目录指定多少存储空间或者允许创建多少个节点

```
setquota -n|-b val path
```

参数说明：

- n 指定可以设置多少个子节点
- b 指定可以设置多大空间 (byte)

示例：

```
setquota -n 5 /zk-123
create /zk-123/1 1
create /zk-123/2 2
create /zk-123/3 3
create /zk-123/4 4
create /zk-123/5 5
create /zk-123/6 6
```

说明：

对于配额不是硬性的提示，超过配额还是可以继续创建，只不过在日志里面有提示

配额的用途：

限制子节点的创建个数和分配空间的大小，如指定某个session有多少空间可以用

9. 查看配额

```
listquota path
```

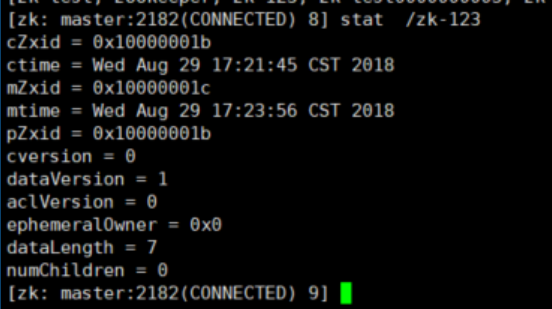
10. 查看节点的状态

```
stat path
```

示例：

查看节点zk-123的状态


```
stat /zk-123
```



分类: [Zookeeper](#)

好文置顶 关注我 收藏该文



 小不点啊

关注 - 4

粉丝 - 382

+加关注

4 0

« 上一篇: [Mysql系列九：使用zookeeper管理远程Mycat配置文件、Mycat监控、Mycat数据迁移（扩容）](#)
» 下一篇: [Zookeeper系列二：分布式架构详解、分布式技术详解、分布式事务](#)

登录后才能查看或发表评论，立即 [登录](#) 或者 [逛逛](#) 博客园首页

编辑推荐：

- [C# 异步编程由浅入深（三）细说 Awaiter](#)
- [突破限制，CSS font-variation 可变字体的魅力](#)
- [浅谈 C# 字符串构建利器 StringBuilder](#)
- [老项目的倔强 —— 性能优化篇](#)
- [2021 .NET Conf China 主题分享之-轻松玩转.NET大规模版本升级](#)

最新新闻：

- [龚宇与陈睿殊途同归](#)
 - [睡前刷8分钟手机，身体兴奋1小时](#)
 - [Google 员工 4 月返回办公室，居家办公「没戏」了？](#)
 - [关闭68家门店，亚马逊的零售「野心」转向了](#)
 - [内卷失败：敲了10000小时代码，我也没能成为一名高级程序员](#)
- » [更多新闻...](#)

Copyright © 2022 小不点啊
Powered by .NET 6 on Kubernetes