01 初识Zookeeper

1.1 Zookeeper发音

1.2 What is Zookeeper?

> Zookeeper官网：https://zookeeper.apache.org/
>
> Wiki：https://en.wikipedia.org/wiki/Apache_ZooKeeper

ZooKeeper is modeled after Google's Chubby lock service[11][12] and was originally developed at Yahoo! for streamlining the processes running on big-data clusters by storing the status in local log files on the ZooKeeper servers. These servers communicate with the client machines to provide them the information. ZooKeeper was developed in order to fix the bugs that occurred while deploying distributed big-data applications. 最早是由雅虎开发的，参考了Google Chubby锁的服务

Apache ZooKeeper

From Wikipedia, the free encyclopedia

高可靠的分布式协调系统

Apache ZooKeeper is an open-source server for highly reliable distributed coordination of cloud applications.[2] It is a project of the Apache Software Foundation.

ZooKeeper is essentially a service for distributed systems offering a hierarchical key-value store, which is used to provide a distributed configuration service, synchronization service, and naming registry for large distributed systems (see *Use cases*).[3] ZooKeeper was a sub-project of Hadoop but is now a top-level Apache project in its own right. 之前是Apache Hadoop的子项目，现在是Apache的顶级项目

1.3 经典应用场景

Use cases [edit] 经典应用场景

Typical use cases for ZooKeeper are:

- Naming service
- Configuration management
- Data Synchronization
- Leader election
- Message queue
- Notification system

1.4 使用Zookeeper的项目

1.4.1 常见的Apache项目

Apache projects using ZooKeeper [edit]

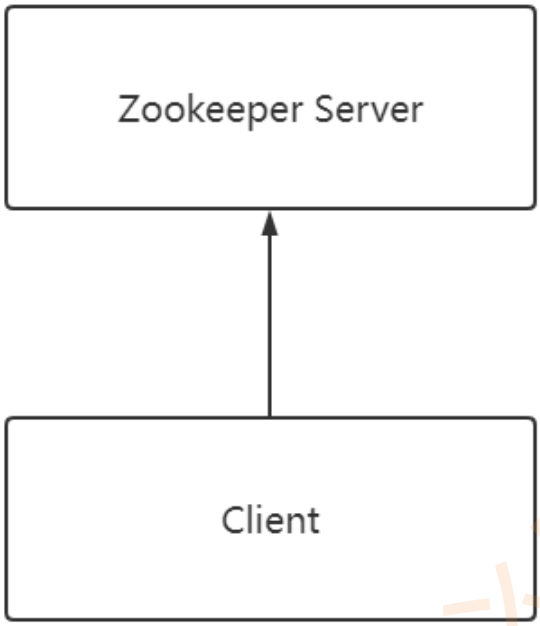
- Apache Hadoop
- Apache Accumulo
- Apache HBase
- Apache Hive
- Apache Kafka
- Apache Solr
- Apache Spark
- Apache NiFi
- Apache Druid
- Apache Helix
- Apache Pinot

1.4.2 详细列表

Zookeeper Use Cases：https://zookeeper.apache.org/doc/current/zookeeperUseCases.html

02 Zookeeper单机架构与环境安装

2.1 单机架构



2.2 单机环境安装

2.2.1 centos7.x

（1）前置环境

```
>=Java8
Maven
```

（2）Zookeeper下载页面

https://zookeeper.apache.org/releases.html

（3）版本3.7.1

```
wget https://www.apache.org/dyn/closer.lua/zookeeper/zookeeper-3.7.1/apache-zookeeper-3.7.1-bin.tar.gz
```

（4）解压

```
tar -xvf apache-zookeeper-3.7.1-bin.tar.gz
```

（5）进入到apache-zookeeper-3.7.1-bin的bin文件夹，查看启动脚本zkServer.sh

```
./zkServer.sh help
```

```
[root@bin]# ./zkServer.sh help
ZooKeeper JMX enabled by default
Using config: /usr/local/zookeeper/apache-zookeeper-3.7.1-bin/bin/../conf/zoo.cfg
Usage: ./zkServer.sh [--config <conf-dir>] {start|start-foreground|stop|version|restart|status|print-cmd}
```

（6）进入conf目录并拷贝zoo_sample.cfg，因为启动的时候会默认使用conf/zoo.cfg文件，当然也可以指定启动文件

```
cp zoo_sample.cfg zoo.cfg
```

（7）修改配置文件conf/zoo.cfg，比如修改dataDir、dataLogDir和clientPort等，这里先修改一下dataDir，也就是快照文件存储的目录。当然如果不额外指定dataLogDir，则事务日志文件也会存储到dataDir中。

```
mkdir -p /usr/local/zookeeper/apache-zookeeper-3.7.1-bin/data
dataDir=/usr/local/zookeeper/apache-zookeeper-3.7.1-bin/data
```

```
tickTime=2000
initLimit=10

syncLimit=5
dataDir=/usr/local/zookeeper/apache-zookeeper-3.7.1-bin/data
clientPort=2181
```

（8）根据自己的需求选择是否配置zk的环境变量，也就是在任意目录之下可以使用bin中的命令

```
vim /etc/profile
# 文件末尾增加如下配置
export ZOOKEEPER_HOME=/usr/local/zookeeper/apache-zookeeper-3.7.1-bin
export PATH=$PATH:$ZOOKEEPER_HOME/bin

# 让profile文件生效
source /etc/profile
```

（9）启动与停止zkServer

```
zkServer.sh start   # 启动
zkServer.sh start-foreground    # 前端启动
zkServer.sh status  # 查看状态
zkServer.sh stop    # 停止
```

2.2.2 docker

（1）打开https://hub.docker.com

（2）搜索zookeeper镜像，版本选择: 3.7.1

（3）拉取镜像

```
docker pull zookeeper:3.7.1
```

（4）运行zookeeper container

```
docker run -d --name jack-zk-server -p 2281:2181 zookeeper:3.7.1
```

（5）查看启动结果

```
docker ps
lsof -i:2281
```

2.2.3 源码

（1）下载源码

https://www.apache.org/dyn/closer.lua/zookeeper/zookeeper-3.7.1/apache-zookeeper-3.7.1.tar.gz

（2）maven构建

> 在源码根目录下执行如下命令

```
mvn clean install -DskipTests
```

```
[INFO]
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ zookeeper-compatibility-tests-curator ---
[INFO] Skipping artifact installation
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary for Apache ZooKeeper 3.7.1:
[INFO]
[INFO] Apache ZooKeeper ................................... SUCCESS [  7.771 s]
[INFO] Apache ZooKeeper - Documentation ................... SUCCESS [  2.196 s]
[INFO] Apache ZooKeeper - Jute ............................ SUCCESS [ 12.201 s]
[INFO] Apache ZooKeeper - Server .......................... SUCCESS [ 27.658 s]
[INFO] Apache ZooKeeper - Metrics Providers ............... SUCCESS [  0.202 s]
[INFO] Apache ZooKeeper - Prometheus.io Metrics Provider .. SUCCESS [  1.676 s]
[INFO] Apache ZooKeeper - Client .......................... SUCCESS [  0.155 s]
[INFO] Apache ZooKeeper - Recipes ......................... SUCCESS [  0.184 s]
[INFO] Apache ZooKeeper - Recipes - Election .............. SUCCESS [  0.491 s]
[INFO] Apache ZooKeeper - Recipes - Lock .................. SUCCESS [  0.474 s]
[INFO] Apache ZooKeeper - Recipes - Queue ................. SUCCESS [  0.421 s]
[INFO] Apache ZooKeeper - Assembly ........................ SUCCESS [  7.102 s]
[INFO] Apache ZooKeeper - Compatibility Tests ............. SUCCESS [  0.170 s]
[INFO] Apache ZooKeeper - Compatibility Tests - Curator ... SUCCESS [  0.456 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  01:01 min
```

（3）导入到idea中

(4) 创建zoo.cfg文件并配置

> conf/zoo.cfg

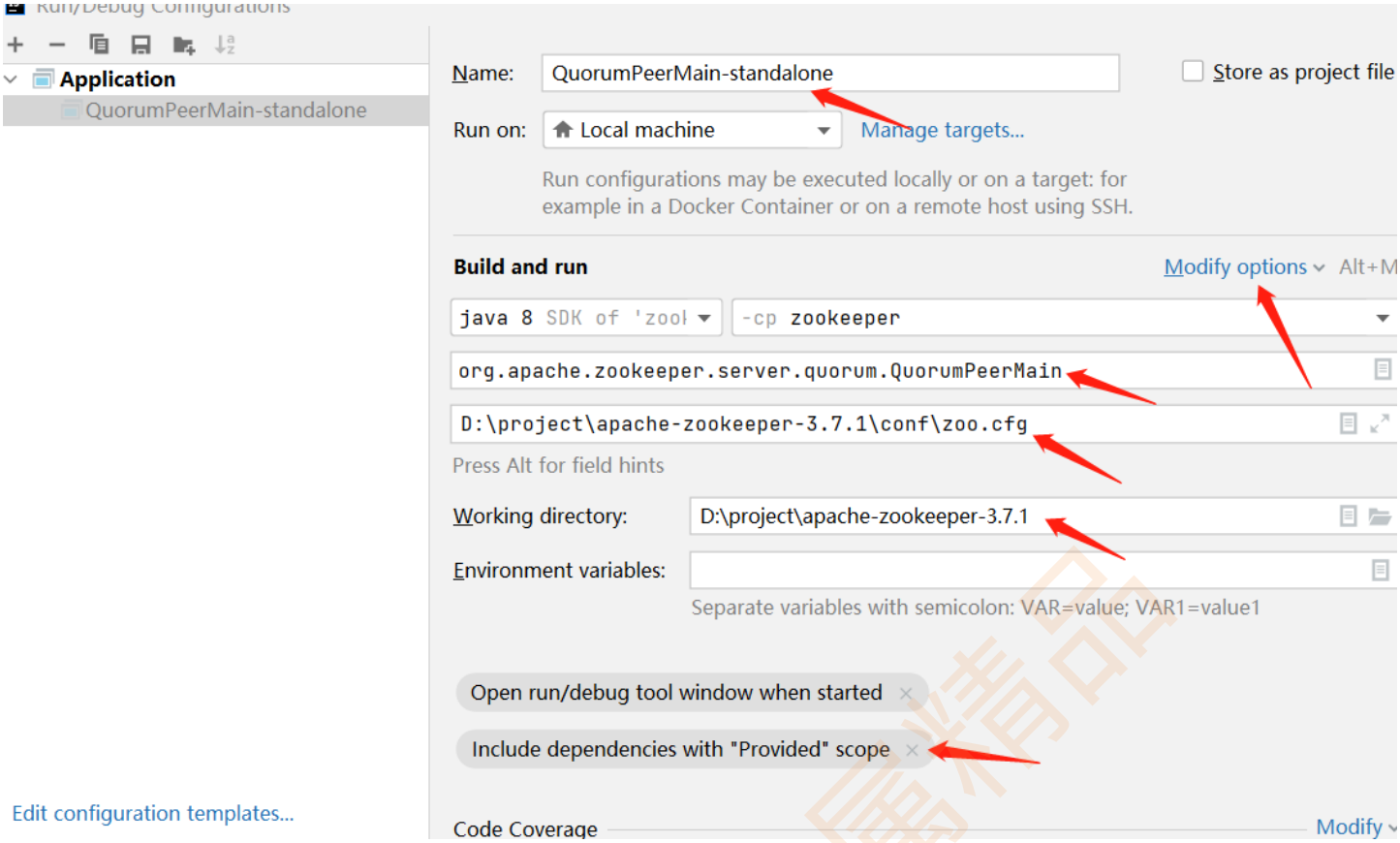
```
tickTime=2000
initLimit=10
syncLimit=5
dataDir=D:\\install\\dev\\zookeeper\\zk-standalone\\data
clientPort=2181
```

(5) 寻找启动类

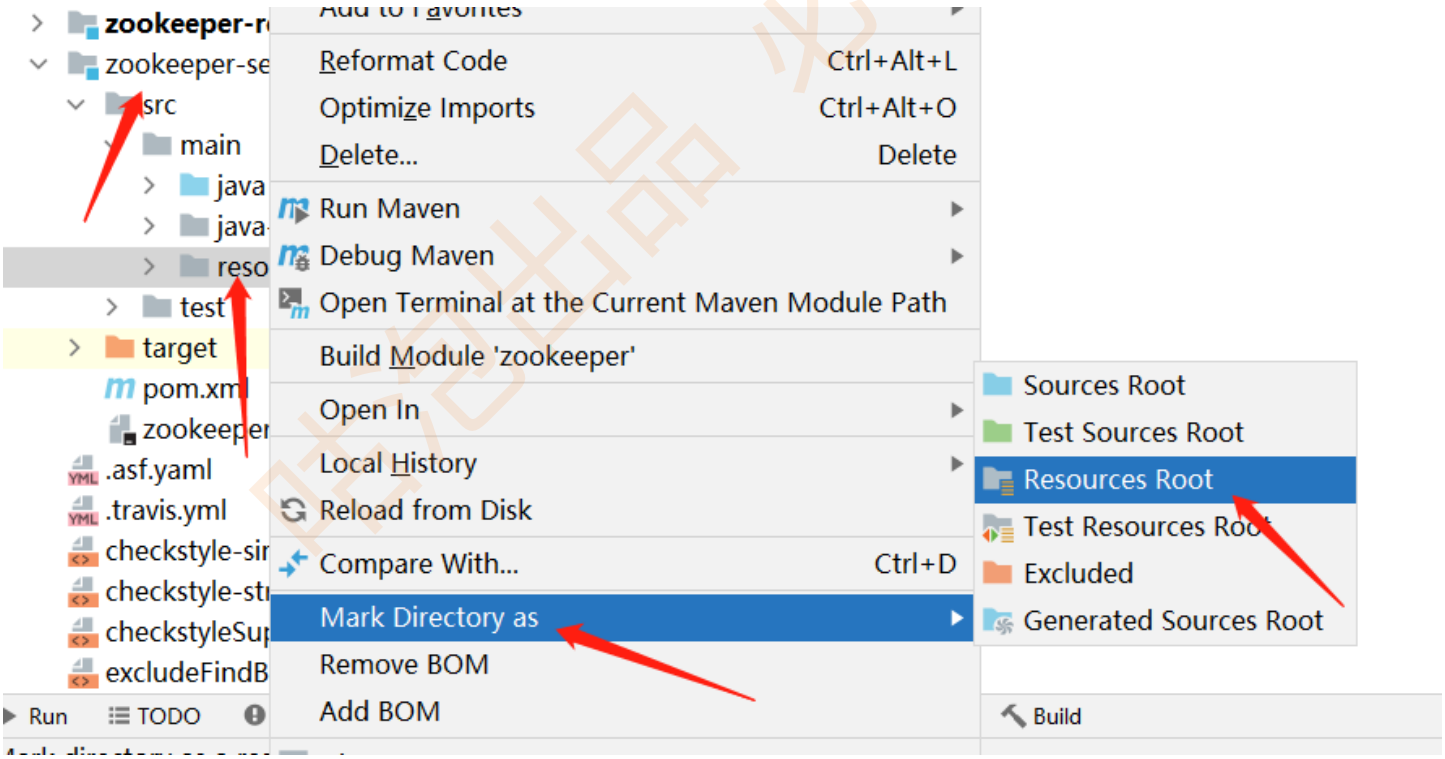
> bin/zkServer.cmd 或 bin/zkServer.sh
>
> 也就是服务端启动的时候，会通过QuorumPeerMain进行启动

ZOOMAIN=org.apache.zookeeper.server.quorum.QuorumPeerMain

(6) 配置启动类



（6） 标记resouces目录



（7） 日志输出

a- 将conf文件下的log.properties文件复制到zookeeper-server的resources文件夹下

b- 在zookeeper-server的pom.xml中添加slf4j的依赖

```
<dependency>
  <groupId>org.slf4j</groupId>
```

```
    <artifactId>slf4j-simple</artifactId>
    <version>1.7.25</version>
    <scope>compile</scope>
</dependency>
```

（8）启动zk server-standalone

```
[main] INFO org.apache.zookeeper.server.quorum.QuorumPeerConfig - Reading configuration from: D:\project\apache-zookeeper-3.7.1\conf\zoo.cfg
[main] INFO org.apache.zookeeper.server.quorum.QuorumPeerConfig - clientPortAddress is 0.0.0.0:2181
[main] INFO org.apache.zookeeper.server.quorum.QuorumPeerConfig - secureClientPort is not set
[main] INFO org.apache.zookeeper.server.quorum.QuorumPeerConfig - observerMasterPort is not set
[main] INFO org.apache.zookeeper.server.quorum.QuorumPeerConfig - metricsProvider.className is org.apache.zookeeper.metrics.impl.DefaultMetricsProvider
[main] INFO org.apache.zookeeper.server.DatadirCleanupManager - autopurge.snapRetainCount set to 3
[main] INFO org.apache.zookeeper.server.DatadirCleanupManager - autopurge.purgeInterval set to 0
[main] INFO org.apache.zookeeper.server.DatadirCleanupManager - Purge task is not scheduled.
[main] WARN org.apache.zookeeper.server.quorum.QuorumPeerMain - Either no config or no quorum defined in config, running in standalone mode
[main] INFO org.apache.zookeeper.jmx.ManagedUtil - Log4j 1.2 jmx support not found; jmx disabled.
[main] INFO org.apache.zookeeper.server.quorum.QuorumPeerConfig - Reading configuration from: D:\project\apache-zookeeper-3.7.1\conf\zoo.cfg
[main] INFO org.apache.zookeeper.server.quorum.QuorumPeerConfig - clientPortAddress is 0.0.0.0:2181
[main] INFO org.apache.zookeeper.server.quorum.QuorumPeerConfig - secureClientPort is not set
[main] INFO org.apache.zookeeper.server.quorum.QuorumPeerConfig - observerMasterPort is not set
[main] INFO org.apache.zookeeper.server.quorum.QuorumPeerConfig - metricsProvider.className is org.apache.zookeeper.metrics.impl.DefaultMetricsProvider
[main] INFO org.apache.zookeeper.server.ZooKeeperServerMain - Starting server
```

2.3 客户端连接

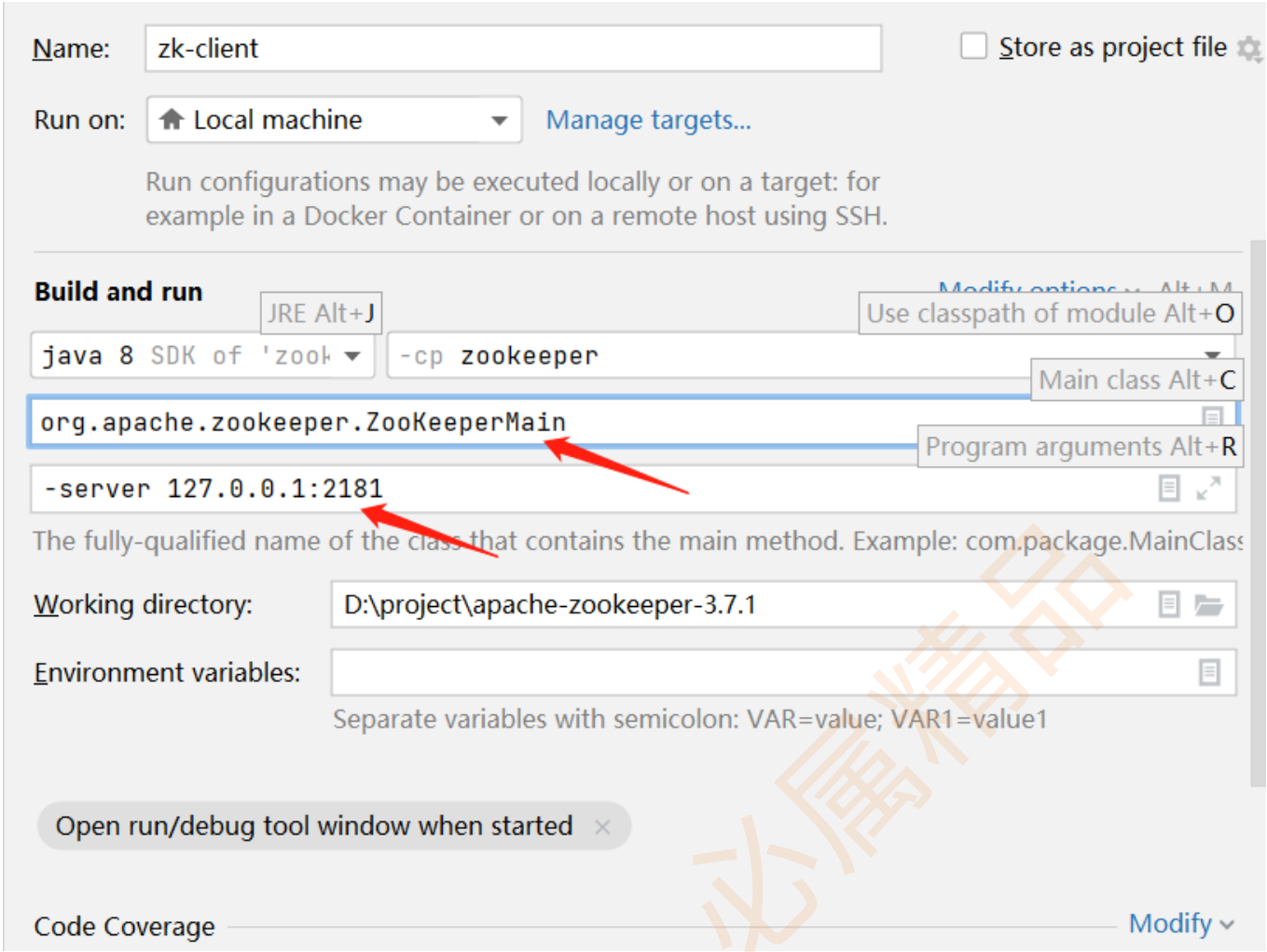
2.3.1 centos7.x

> 通过zkCli.sh连接

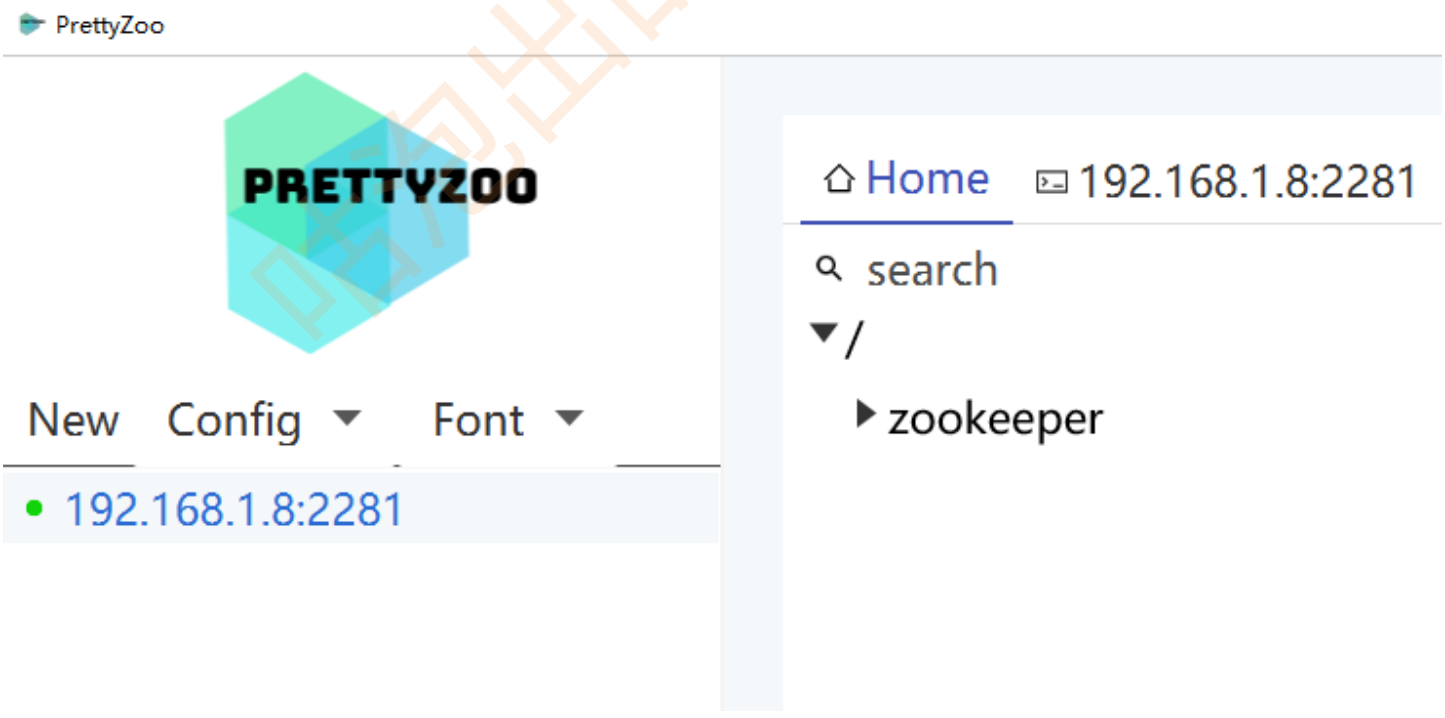
```
zkCli.sh    # 默认连接本机的2181端口
zkCli.sh -server  127.0.0.1:2181    # 指定zk server
```

2.3.2 源码

> 找到zkCli.cmd，发现客户端是通过ZookeeperMain运行的



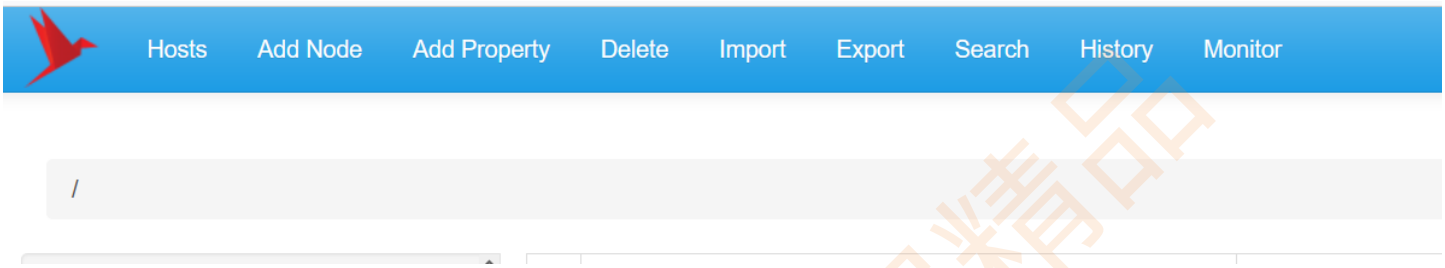
2.3.3 prettyZoo



2.3.4 web图形化界面zkui

github：https://github.com/DeemOpen/zkui

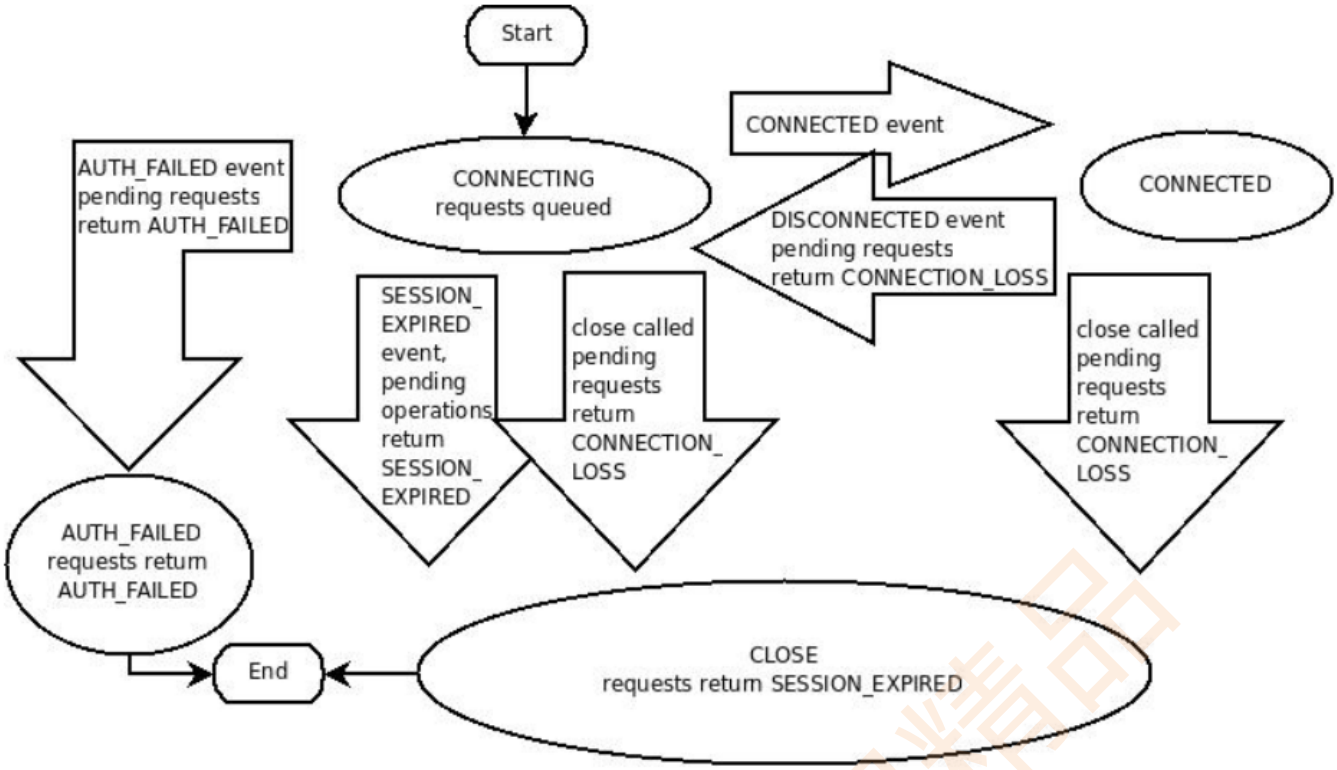
```
（0）来到zk-server所在机器
（1）下载源码：git clone https://github.com/DeemOpen/zkui
（2）mvn clean install
（3）复制config.cfg到jar包所在目录：target
（4）打开配置文件，配置zkServer=zkServer=192.168.0.8:2181
（5）zkui默认端口在9090：serverPort=9090
（6）运行jar包：java -jar zkui-2.0-SNAPSHOT-jar-with-dependencies.jar
（7）访问地址：localhost:9090
（8）默认用户名密码：admin  manager，可以通过userSet修改
```



03 Zookeeper Sessions

官网：https://zookeeper.apache.org/doc/current/zookeeperProgrammers.html#ch_zkSessions

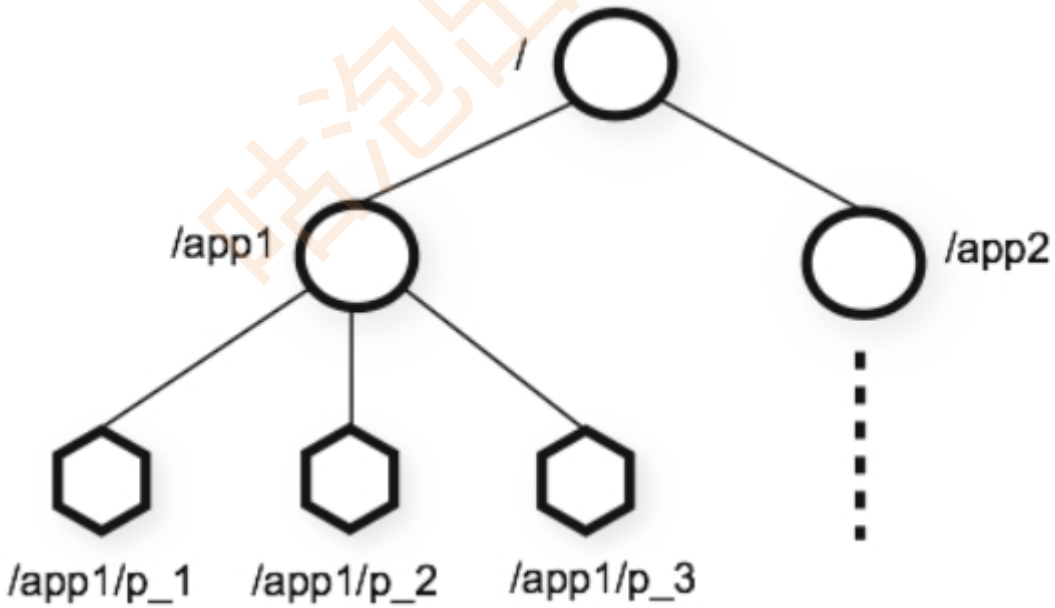
> 可以通过zkCli.sh 查看session相关信息，比如sessionTimeout=30000毫秒、seession id=0x10000314a5a0002
> TCP长连接，超时时间指的是客户端和服务端因为一些原因断开了连接，如果在超时时间内能够重新连接上，则会话依然有效



04 数据模型与基本操作

4.1 数据模型

官网：https://zookeeper.apache.org/doc/current/zookeeperOver.html#sc_dataModelNameSpace



4.2 基本操作

```
# 查看帮助
    help

# 创建节点
    create /gupao
    create /gupao/jack
    create /gupao/james

# 创建节点并往节点上存数据
    create /gupao/jack/age 17
    create /gupao/jack/hobby coding

# 获取ZNode
    ls /
    ls /gupao
    ls -R /gupao

# 查询某个ZNode数据
    get /gupao/jack
    get /gupao/jack/age
    get /gupao/jack/hobby

# 修改ZNode的值
    set /gupao/jack/age 16
    get /gupao/jack/age

# 删除节点，当某个节点下存在子节点时，不能直接删除该节点
    delete /gupao/jack/age
    ls -R /
    deleteall /gupao
    ls -R /
```


4.3 节点状态

官网：https://zookeeper.apache.org/doc/current/zookeeperProgrammers.html#sc_zkStatStructure

```
# 查询节点状态信息(详情)     总共11个
```

```
stat /gupao
get -s /gupao
```

4.3.1 时间相关

（1）ctime：当前节点比创建的时间 The time in milliseconds from epoch when this znode was created. 比如ctime = Thu Jul 21 22:00:07 UTC 2022

（2）mtime：当前节点最后一次被修改的时间 The time in milliseconds from epoch when this znode was last modified. 比如mtime = Thu Jul 21 22:08:47 UTC 2022

4.3.2 容易理解

（1）numChildren：当前节点子节点的数量

（2）dataLength：当前节点保存的数据长度

（3）ephemeralOwner：当前节点是否为临时节点，如果为临时节点则该值为sessionid，否则为0 The session id of the owner of this znode if the znode is an ephemeral node. If it is not an ephemeral node, it will be zero.

4.3.3 版本相关

（1）dataVersion：当前节点的数据被修改的次数，也就是数据的版本

```
set -v 2 /gupao 888
set -v 1 /gupao 888
stat /gupao
```

（2）cversion：子节点的版本号

```
create /gupao/test1
stat /gupao
create /gupao/test2
stat /gupao
```

（3）aversion：acl的版本号，这块可以等看完acl的知识后自己测试

4.3.4 zxid

官网：https://zookeeper.apache.org/doc/current/zookeeperProgrammers.html#sc_timeInZk

> 理解zxid

zxid是事务编号，8字节的整型数，即64个比特位，前32位标识epoch，后32位用来计数。

zxid的初始值为0，用二进制标识就是 00000000 00000000 00000000 00000000

每一次事务请求都会把后面32位的值+1，比如进行了10次事务请求，则zxid变为 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00001010

每进行一次leader选举，前32位的值就会+1，并把后面的32位清零，则zxid变为 00000000 00000000 00000000 00000001 00000000 00000000 00000000 00000000

若一直没有进行leader选举，同时一直在发生事务请求，则后面32位会一直增加，极限值是00000000 00000000 00000000 00000000 11111111 11111111 11111111 11111111

此时再发生事务请求，则把前面的32位+1，变成00000000 00000000 00000000 00000001 00000000 00000000 00000000 00000000

（1）czxid：表示当前节点被创建时的事务id

（2）mzxid：表示当前节点被最后一次更新时的事务id

（3）pzxid：表示该节点的子节点最后一次被修改时的事务id。只有子节点变化才会影响pzxid，子节点的数据变化不会影响pzxid

```
create /zk-zxid

stat /zk-zxid

stat /zk-zxid

create /zk-zxid/child01
stat /zk-zxid
stat /zk-zxid/child01

set /zk-zxid/child01 111
stat /zk-zxid

create /zk-zxid/child02
stat /zk-zxid
```


05 节点特性

> 查看create命令

```
[zk: localhost:2181(CONNECTED) 22] create
create [-s] [-e] [-c] [-t ttl] path [data] [acl]
```

5.1 持久节点PERSISTENT

> 不会因为客户端宕机而删除节点

```
create /jack 666
stat /jack   # ephemeralOwner = 0x0
quit
zkCli.sh
ls /
```

5.2 临时节点EPHEMERAL

> 会因为客户端宕机而删除节点

```
create -e /test 111
stat /test   # ephemeralOwner = 0x100000aa4c70002  sessionid
quit
zkCli.sh
ls /
```

5.3 有序节点SEQUENTIAL

> 会给节点名称添加一个自增的序号

5.3.1 持久有序节点

```
create -s /order
create -s /order
create -s /order
ls /
```

5.3.2 临时有序节点

```
create -s -e /product
create -s -e /product
create -s -e /product
ls -R /
```

```
quit
zkCli.sh
ls -R /
```

5.4 容器节点

Added in 3.6.0：https://zookeeper.apache.org/doc/current/zookeeperProgrammers.html#Container+Nodes

ZooKeeper has the notion of container znodes. Container znodes are special purpose znodes useful for recipes such as leader, lock, etc. When the last child of a container is deleted, the container becomes a candidate to be deleted by the server at some point in the future.

```
create -c /jack-container
create /jack-container/sub1
create /jack-container/sub2
ls -R /jack-container

delete /jack-container/sub1
delete /jack-container/sub2

# 等待一段时间之后，查看节点是否存在
```

5.5 TTL节点

Added in 3.6.0：https://zookeeper.apache.org/doc/current/zookeeperProgrammers.html#TTL+Nodes

When creating PERSISTENT or PERSISTENT_SEQUENTIAL znodes, you can optionally set a TTL in milliseconds for the znode. If the znode is not modified within the TTL and has no children it will become a candidate to be deleted by the server at some point in the future.

Note: TTL Nodes must be enabled via System property as they are disabled by default. 在zoo.cfg中配置zookeeper.extendedTypesEnabled=true[记得重启zkServer]

```
create -t 10 /zk-ttl 111
ls /
```