### 走一遍

systemctl stop firewalld.service #停止firewall

systemctl disable firewalld.service #禁止firewall开机启动

firewall-cmd --state #查看默认防火墙状态（关闭后显示notrunning，开启后显示running）

|  |
| --- |
| **import** org.springframework.cloud.commons.util.IdUtils;  **public** **class** SnowFlake {  **private** **static** **volatile** IdUtils *idUtils* = **null**;  **private** **final** **long** workerId;  **private** **static** **final** **long** ***DEFAULT\_WORKER\_ID*** = 1L;  **private** **long** sequence = 0L;  /\*\*  \* 工作机器id  \*/  **private** **final** **long** workerIdBits = 10L;  **private** **long** lastTimestamp = -1L;  **private** SnowFlake(**long** workerId) {  **super**();  **long** maxWorkerId = -1L ^ -1L << **this**.workerIdBits;  **if** (workerId > maxWorkerId || workerId < 0) {  **throw** **new** IllegalArgumentException(  String.*format*("worker Id can't be greater than %d or less than 0", maxWorkerId));  }  **this**.workerId = workerId;  }  **private** **synchronized** **long** nextId() **throws** Exception {  **long** timestamp = **this**.timeGen();  // 序列号  **long** sequenceBits = 12L;  **if** (**this**.lastTimestamp == timestamp) {  **long** sequenceMask = -1L ^ -1L << sequenceBits;  **this**.sequence = **this**.sequence + 1 & sequenceMask;  **if** (**this**.sequence == 0) {  timestamp = **this**.tilNextMillis(**this**.lastTimestamp);  }  } **else** {  **this**.sequence = 0;  }  **if** (timestamp < **this**.lastTimestamp) {  **throw** **new** Exception(String.*format*("Clock moved backwards.Refusing to generate id for %d milliseconds",  (**this**.lastTimestamp - timestamp)));  }  **this**.lastTimestamp = timestamp;  **long** timestampLeftShift = sequenceBits + **this**.workerIdBits;  // 开始时间截 2018/08/06  **long** snsEpoch = 1533524193324L;  **return** timestamp - snsEpoch << timestampLeftShift | **this**.workerId << sequenceBits | **this**.sequence;  }  /\*\*  \* 保证返回的毫秒数在参数之后  \*/  **private** **long** tilNextMillis(**long** lastTimestamp) {  **long** timestamp = **this**.timeGen();  **while** (timestamp <= lastTimestamp) {  timestamp = **this**.timeGen();  }  **return** timestamp;  }  /\*\*  \* 获得系统当前毫秒数  \*/  **private** **long** timeGen() {  **return** System.*currentTimeMillis*();  }  /\*\*  \* id生成  \*/  **public** **static** String generator() **throws** Exception {  **if** (*idUtils* == **null**) {  **synchronized** (IdUtils.**class**) {  *idUtils* = **new** IdUtils(***DEFAULT\_WORKER\_ID***);  }  }  **return** String.*valueOf*(*idUtils*.nextId());  }  } |