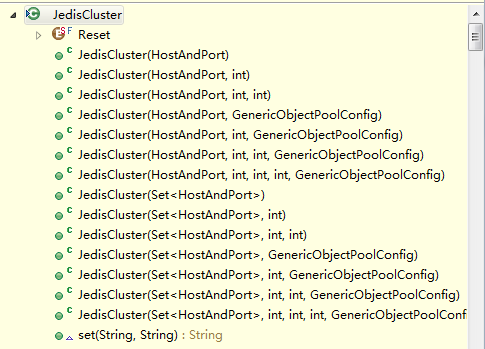
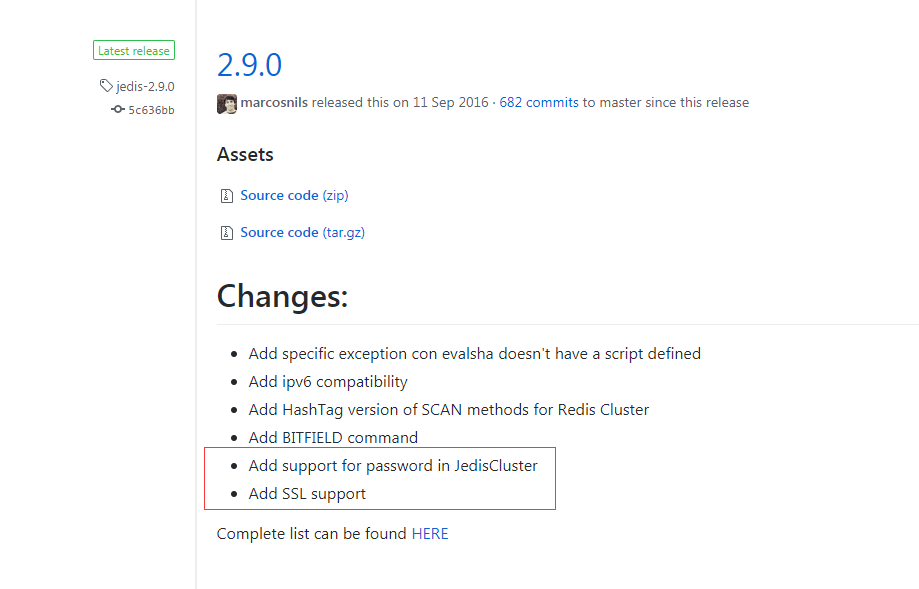
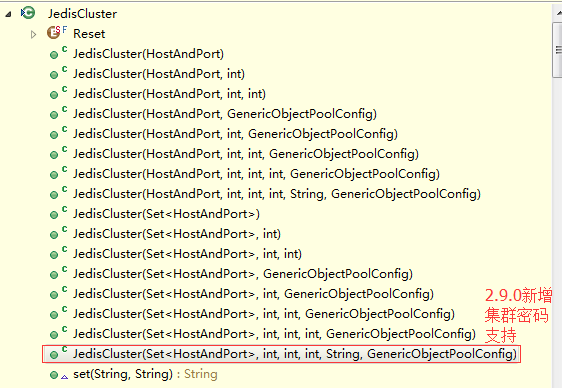
jedis-2.8.1，不支持集群密码设置， 构造方法列表如下：



jedis-2.9.0才支持集群密码





RedisClusterConfiguration

使用连接到Redis群集的配置类，用于通过RedisConnectionFactory设置RedisConnection。 在设置高可用性Redis环境时很有用。

**private** **void** initializeSlotsCache(Set<HostAndPort> startNodes, GenericObjectPoolConfig poolConfig, String password) {

**for** (HostAndPort hostAndPort : startNodes) {

Jedis jedis = **new** Jedis(hostAndPort.getHost(), hostAndPort.getPort());

**if** (password != **null**) {

jedis.auth(password);

}

**try** {

cache.discoverClusterNodesAndSlots(jedis);

**break**;

} **catch** (JedisConnectionException e) {

// try next nodes

} **finally** {

**if** (jedis != **null**) {

jedis.close();

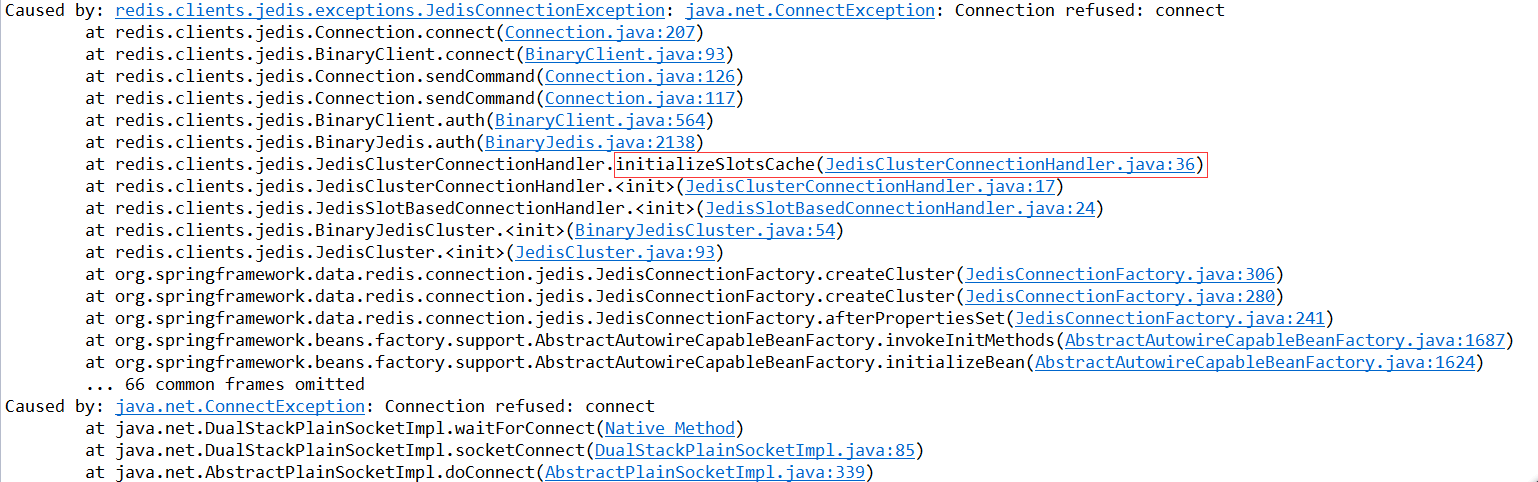
}

}

}

}

注意上面代码段for循环里面的break：   
它让for循环立刻跳出，导致for循环实际上只执行了一次，也就是说只使用了配置文件中所有节点配置的第一个，如此说来，集群配置文件中节点配置一个和配置多个效果一致。   
同时还要注意到：通过获取的第一个节点配置实例化了一个Jedis jedis，如果配置文件中第一个节点指向的服务挂机或无法连接，将导致程序无法使用整个集群，虽然redis集群中其它节点是可用的。第一个节点（并不一定是配置时的第一个节点，set无序存储）宕机，出现如下异常



集群状态为fail也会导致如上异常。

节点顺序：



sentinel哨兵模式监控Master，当Master挂掉时自动选举Slaver变为Master，其它Slaver自动重连新的Master



在sentinel的配置文件中添加以下内容，实现后台运行sentinel：

[root@redis\_01 redis]# vim sentinel.conf

daemonize yes

logfile "/var/log/sentinel\_log.log"

sentinel monitor mymaster 127.0.0.1 6379 2

* 当前Sentinel节点监控 127.0.0.1:6379 这个主节点
* 2代表判断主节点失败至少需要2个Sentinel节点节点同意
* mymaster是主节点的别名

sentinel auth-pass \

* 如果Sentinel监控的主节点配置了密码，可以通过sentinel auth-pass配置通过添加主节点的密码，防止Sentinel节点无法对主节点进行监控。