springCloud笔记

1、天气api

<http://wthrcdn.etouch.cn/weather_mini?citykey=101280601>

<http://wthrcdn.etouch.cn/weather_mini?city>=深圳

http://mobile.weather.com.cn/js/citylist.xml

2、springboot redis配置

springboot使用spring-boot-starter-data-redis来自动配置lettuce和jedis这两种客户端.默认选用的是lettuce.

由于springboot版本不一样,低版本和高版本的配置会有所不同：

低版本的配置：

1. # REDIS (RedisProperties)
2. # Redis数据库索引（默认为0）
3. spring.redis.database=0
4. # Redis服务器地址
5. spring.redis.host=localhost
6. # Redis服务器连接端口
7. spring.redis.port=6379
8. # Redis服务器连接密码（默认为空）
9. spring.redis.password=
10. # 连接池最大连接数（使用负值表示没有限制）
11. spring.redis.pool.max-active=8
12. # 连接池最大阻塞等待时间（使用负值表示没有限制）
13. spring.redis.pool.max-wait=-1
14. # 连接池中的最大空闲连接
15. spring.redis.pool.max-idle=8
16. # 连接池中的最小空闲连接
17. spring.redis.pool.min-idle=0
18. # 连接超时时间（毫秒）
19. spring.redis.timeout=0

高版本的配置：

1. # REDIS (RedisProperties)
2. # Redis数据库索引（默认为0）
3. spring.redis.database=0
4. # Redis服务器地址
5. spring.redis.host=localhost
6. # Redis服务器连接端口
7. spring.redis.port=6379
8. # Redis服务器连接密码（默认为空）
9. spring.redis.password=
10. # 连接池最大连接数（使用负值表示没有限制）
11. spring.redis.jedis.pool.max-active=8
12. # 连接池最大阻塞等待时间（使用负值表示没有限制）
13. spring.redis.jedis.pool.max-wait=-1ms
14. # 连接池中的最大空闲连接
15. spring.redis.jedis.pool.max-idle=8
16. # 连接池中的最小空闲连接
17. spring.redis.jedis.pool.min-idle=0
18. # 连接超时时间（毫秒）
19. spring.redis.timeout=0ms
20. #############redis哨兵设置##############
21. #redis服务器master的名字
22. spring.redis.sentinel.master=master1
23. #redis-sentinel的配置地址和端口
24. spring.redis.sentinel.nodes=127.0.0.1:6381,127.0.0.1:6382

注意的是,timeout和max-wait如果不在后面加上ms就会报错

lettuce专有的配置

1. #连接池最大连接数,负数表示不限制
2. spring.redis.lettuce.pool.max-active=8
3. #连接池最大空闲数,负数表示不限制
4. spring.redis.lettuce.pool.max-idle=8
5. #等待可用连接的最大时间,负数表示不限制
6. spring.redis.lettuce.pool.max-wait=-1ms
7. #连接池最小空闲数
8. spring.redis.lettuce.pool.min-idle=0
9. #连接超时时间
10. spring.redis.lettuce.shutdown-timeout=1000ms

jedis专有配置(和lettuce类似)：

1. spring.redis.jedis.pool.max-active=8
2. spring.redis.jedis.pool.max-idle=8
3. spring.redis.jedis.pool.max-wait=-1ms
4. spring.redis.jedis.pool.min-idle=0

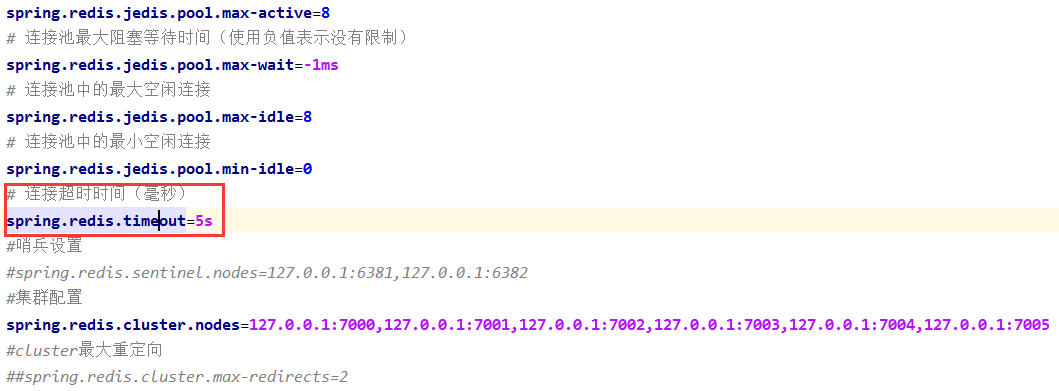
<https://majing.io/posts/10000020931206>

弄了一天的redis，百度springboot redisTemplate都是重写configuration.但是自己重写有错.

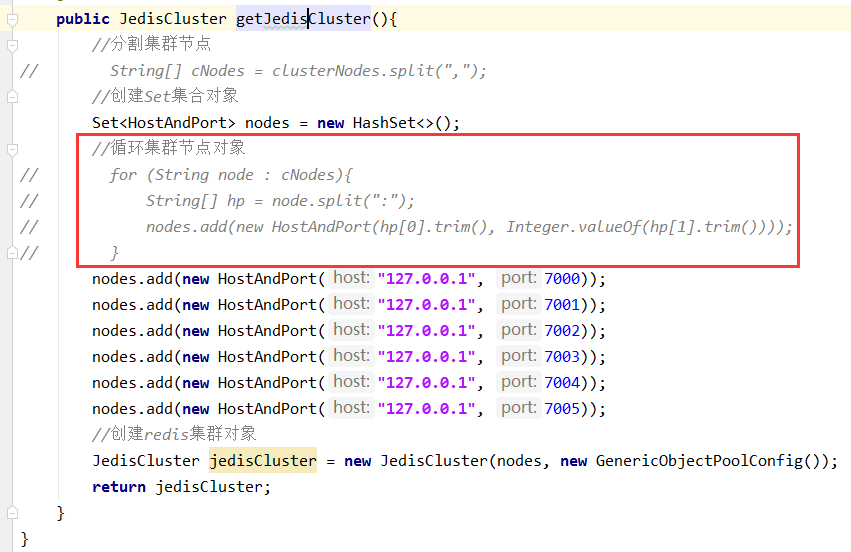
spring-data-redis自2.0后自带lettuce和jedis，默认的是lettuce，像上面那样配置分布式后会报错，但是自己就像用lettuce,因为有优点,但是没成功。

承上：

之前找资料都是使用redisTemplate或者jedis的,由于之前用过了jedis,所以觉得可以搞定.由于redisTemplate配置集群总是不成功,就先放一边了.然后开始用jedis测试,但是测试的时候出问题,在application.properties配置时,会报错java.lang.NumberFormatException: For input string: "5s",因为部分配置如下：



不知道为什么后面加个”s”就报错,不加又不正确(报红).另外,新建springJedisConfig时,如下所示:

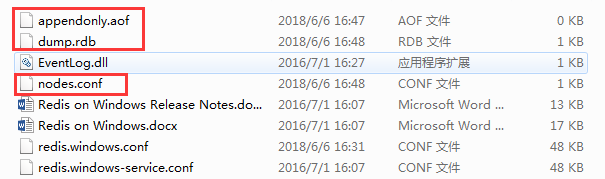


使用红色框里的方法时会报错: java.lang.NumberFormatException: For input string: 具体涉及到add方法里的Integer.parseInt或者Integer.valueof.这里没解决,都说是版本问题,把jedis()于是自己直接配置6个集群,这样虽然在启动时不会报错,但是在访问接口时(<http://localhost:8080/weather/cityName/>武汉),会报错: JedisDataException: ERR This instance has cluster support disabled,百度了下,原来是redis.conf里的配置cluster-enable yes没有打开.然后才明白为什么要jedis.cluster.原来在redis3.0以后配置集群采用的jedis.cluster.之前用的2.x的redis,没有用到cluster,而是采用的shardedredis.正是因为用的3.x的缘故,即使把上一个问题解决了,还是会报错: No reachable node in cluster. 突然想到之前看windows下redis配置时,有个用到了ruby,于是想着是不是3.x的redis安装配置集群和2.x的不一样.然后就百度了windows下怎么配置redis集群：

<https://blog.csdn.net/zsg88/article/details/73715947>

<https://blog.csdn.net/MrXiaGC/article/details/52799081>

第一次搭建集群的时候,不知道为什么第一个和第二个redis总是在写什么东西,强行结束后,再次配置时出现错误: ERR slot 10923 is already busy.百度后才知道是上次配置的时候的配置还保存着,需要删掉redis文件夹下的三个文件



然后重新配置就好了.至此终于把这个redis集群使用jedis的方式弄好了.但是接下来还想用redisTemplate的方式搭建集群.怎么办呢,lets go!

紧接上文:

上文说到使用红色框里的方法时会报一个错误: java.lang.NumberFormatException: For input string: "7005}".当我把每一个端口号打印出来的时候,发现前面的都正常,最后一个7005端口号后面多了一个大括号”}”,百思不得其解,然后换了一种方法:采用三元表达式,若是后面有”}”则截取前面的字符,若没有,保留原来字符:

nodes.add(new HostAndPort(hp[0].trim(), Integer.parseInt(hp[1].endsWith("}")?hp[1].substring(0,hp[1].length()-1):hp[1])));

这样就解决了这个问题.还有上面一个问题就是application.properties里配置的spring.redis.timeout=5s,不加单位就报红,加了单位启动又报错: java.lang.NumberFormatException: For input string 5s,结合刚刚的方法,把后面的单位截取掉后就解决了问题,有毒！

3、quartz scheduler定时

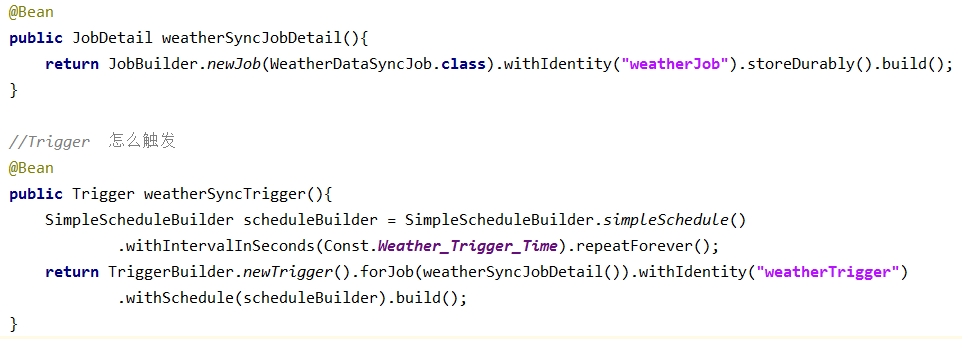
首先引入quartz的依赖：compile(**'org.springframework.boot:spring-boot-starter-quartz'**

quartz中的三个核心概念:

job(任务) scheduler(任务调度器) trigger(触发器)

创建一个任务job,配置相应的job和trigger,通过scheduler来启动任务.

简单示例：



<https://blog.csdn.net/guolong1983811/article/details/51501346>

<https://blog.csdn.net/lkl_csdn/article/details/73613033>