SpringBoot整合Redis（一）配置redis

1. 要求

|  |  |  |
| --- | --- | --- |
| SpringBoot版本 | 1.5.15.RELEASE | 2.0.4.RELEASE |
| Spring-data-redis版本 | 1.8.14.RELEASE | 2.0.9.RELEASE |
|  |  |  |

1. Maven依赖
2. 配置文件(properties)

# HOST  
spring.redis.host=127.0.0.1  
# 端口  
spring.redis.port=6379  
# 密码（如redis未设置密码则不需要配置）  
spring.redis.password=123456  
# 连接的数据库（默认redis配置的数据库数量为16）  
spring.redis.database=0  
# 最大连接数（-1表示不限制，默认为8）  
spring.redis.jedis.pool.max-idle=5  
# 最大活跃连接数（-1表示不限制，默认为8）  
spring.redis.jedis.pool.max-active=5  
# 最小连接数（-1表示不限制，默认为0）  
spring.redis.jedis.pool.min-idle=1  
#spring.redis.pool.max-idle=5 # springboot 1.x版本配置  
#spring.redis.pool.max-active=5 # springboot 1.x版本配置  
#spring.redis.pool.min-idle=1 # springboot 1.x版本配置

这里SpringBoot版本1.x和2.x之间的配置会有所不同，2.x增加了Jedis属性，并将redis连接池Pool作为Jedis类的属性，因此配置连接池多了一层jedis。

public class RedisProperties {  
 private int database = 0;  
 private String url;  
 private String host = "localhost";  
 private String password;  
 private int port = 6379;  
 private boolean ssl;  
 private Duration timeout;  
 private RedisProperties.Sentinel sentinel;  
 private RedisProperties.Cluster cluster;  
 private final RedisProperties.Jedis jedis = new RedisProperties.Jedis();  
 private final RedisProperties.Lettuce lettuce = new RedisProperties.Lettuce();

public static class Jedis {  
 private RedisProperties.Pool pool;  
  
 public Jedis() {  
 }  
  
 public RedisProperties.Pool getPool() {  
 return this.pool;  
 }  
  
 public void setPool(RedisProperties.Pool pool) {  
 this.pool = pool;  
 }  
}

public static class Pool {  
 private int maxIdle = 8;  
 private int minIdle = 0;  
 private int maxActive = 8;  
 private Duration maxWait = Duration.ofMillis(-1L);  
  
 public Pool() {  
 }  
  
 public int getMaxIdle() {  
 return this.maxIdle;  
 }  
  
 public void setMaxIdle(int maxIdle) {  
 this.maxIdle = maxIdle;  
 }  
  
 public int getMinIdle() {  
 return this.minIdle;  
 }  
  
 public void setMinIdle(int minIdle) {  
 this.minIdle = minIdle;  
 }  
  
 public int getMaxActive() {  
 return this.maxActive;  
 }  
  
 public void setMaxActive(int maxActive) {  
 this.maxActive = maxActive;  
 }  
  
 public Duration getMaxWait() {  
 return this.maxWait;  
 }  
  
 public void setMaxWait(Duration maxWait) {  
 this.maxWait = maxWait;  
 }  
}

1. Bean配置

配置bean的方式一般有两种，xml文件和JavaConfig方式。这里通过注解的方式进行配置。

通过RedisConfig类来初始化与redis相关的所有配置和Bean。需要解释的几个地方：

1. JavaConfig配置需要加@Configuration类注解，
2. 这里配置了两个redisTemplate，一个是进行基本的redis操作，另一个激活事务的redis操作，后续文章会详细讲述。
3. 因为2中配置了两个redisTemplate，在Bean注解时会有一定的影响，因为@AutoWired和@Resource注解有不同的作用，而如果某个Bean初始化有参数依赖（如上述的HashOperations的初始化需要依赖RedisTemplate），此时Spring是采用@AutoWired（因为@Resource是java的注解，而@AutoWired则是spring的注解），而根据@AutoWired的特点是按照类型进行注解的，对于有多个相同类型的Bean时会报异常，为了解决这个问题这里使用了@ConditionalOnMissingBean注解，可以作为缺省时的首要选择（即设为默认值）。
4. 工具类