使用Lettuce操作redis

一 使用Lettuce直接操作redis

1 pom文件

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.qianfeng</groupId>
    <artifactId>reids-lettuce</artifactId>
    <version>1.0-SNAPSHOT</version>
    <dependencies>
        <dependency>
            <groupId>io.lettuce</groupId>
            <artifactId>lettuce-core</artifactId>
            <version>5.0.5.RELEASE</version>
        </dependency>
    </dependencies>

</project>
```

2 测试类

```
package com.qianfeng.redis.lettuce;

//
//                            _ooOoo_
//                           o8888888o
//                           88" . "88
//                           (| -_- |)
//                            O\ = /O
//                        ____/`---'\____
//                      .   ' \\| |// `.
//                       / \\||| : |||// \
//                     / _||||| -:- |||||- \
//                       | | \\\ - /// | |
//                     | \_| ''\---/'' | |
//                      \ .-\__ `-` ___/-. /
//                   ___`. .' /--.--\ `. . __
```

```
//                .."" '< `.___\_<|>_/___.' >'"".
//               | | : `- \`.;`\ _ /`;.`/ - ` : | |
//                 \ \ `-. \_ __\ /__ _/ .-` / /
//         ======`-.____`-.___\_____/___.-`____.-'======
//                            `=---='
//
//         .............................................
//                  佛祖镇楼                  BUG辟易
//           佛曰:
//                  写字楼里写字间，写字间里程序员；
//                  程序人员写程序，又拿程序换酒钱。
//                  酒醒只在网上坐，酒醉还来网下眠；
//                  酒醉酒醒日复日，网上网下年复年。
//                  但愿老死电脑间，不愿鞠躬老板前；
//                  奔驰宝马贵者趣，公交自行程序员。
//                  别人笑我忒疯癫，我笑自己命太贱；
//


import io.lettuce.core.RedisClient;
import io.lettuce.core.RedisFuture;
import io.lettuce.core.RedisURI;
import io.lettuce.core.api.StatefulRedisConnection;
import io.lettuce.core.api.async.RedisAsyncCommands;
import io.lettuce.core.api.sync.RedisCommands;
import io.lettuce.core.cluster.RedisClusterClient;
import io.lettuce.core.cluster.api.StatefulRedisClusterConnection;
import io.lettuce.core.cluster.api.async.RedisAdvancedClusterAsyncCommands;
import io.lettuce.core.cluster.api.sync.RedisAdvancedClusterCommands;

import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.ExecutionException;

/**
 * Created by jackiechan on 2019-04-21/09:14
 *
 * @Author jackiechan
 *
 *
 * uri的语法
 * 单独的redis:
 * redis :// [: password@] host [: port] [/ database][?
[timeout=timeout[d|h|m|s|ms|us|ns]] [&_database=database_]]
 *
 * 单独的redis ssl:
 * redis :// [: password@] host [: port] [/ database][?
[timeout=timeout[d|h|m|s|ms|us|ns]] [&_database=database_]]
 *
```

```
 * 单独的redis unix的socket
 * redis-socket :// path [?[timeout=timeout[d|h|m|s|ms|us|ns]]
[&_database=database_]]
 *
 * redis哨兵模式
 *
 * redis-sentinel :// [: password@] host1[: port1] [, host2[: port2]] [,
hostN[: portN]] [/ database][?[timeout=timeout[d|h|m|s|ms|us|ns]]
[&_sentinelMasterId=sentinelMasterId_] [&_database=database_]]
 */
public class Test {
    public static void main(String[] args) {
        operSingle();
    }

    public static void operSingle(){
       // RedisURI redisURI =
RedisURI.create("redis://redis.qfjava.cn:8400");//指定redis服务器地址
       // redisURI.setPassword("redis001"); //设置redis密码
        RedisURI redisURI =
RedisURI.create("redis://redis001@redis.qfjava.cn:8400");//可以使用这一行代替上面两行,指定redis的服务器地址和密码
        RedisClient client = RedisClient.create(redisURI);
        StatefulRedisConnection<String,String> connect = client.connect();//建立连接

        /**
         * 同步调用
         */
        RedisCommands<String,String> commands = connect.sync();//开启同步操作
        commands.set("hello","hello world");
        String str = commands.get("hello");
        //启动redis命令和jedis一样,都是命令的名字
        System.out.println(str);

        /**
         * 异步调用
         */
        RedisAsyncCommands<String,String> asyncCommands = connect.async();//开启异步操作
        RedisFuture<String> future = asyncCommands.get("hello");
        try {
            String str1 = future.get();
            System.out.println(str1);
        } catch (InterruptedException e) {
            e.printStackTrace();
        } catch (ExecutionException e) {
            e.printStackTrace();
        }
```

```
        connect.close();
        client.shutdown();
    }

    /**
     * 集群模式
     */
    public static void operCluster(){
        List<RedisURI> list = new ArrayList<>();
        list.add(RedisURI.create("redis://127.0.0.1:7001"));
        list.add(RedisURI.create("redis://127.0.0.1:7002"));
        list.add(RedisURI.create("redis://127.0.0.1:7003"));
        list.add(RedisURI.create("redis://127.0.0.1:7004"));
        list.add(RedisURI.create("redis://127.0.0.1:7005"));
        list.add(RedisURI.create("redis://127.0.0.1:7006"));
        RedisClusterClient client = RedisClusterClient.create(list);//创建集群客
户端
        StatefulRedisClusterConnection<String, String> connect =
client.connect();//简历连接

        /**
         * 同步执行命令
         */
        RedisAdvancedClusterCommands<String, String> commands =
connect.sync();
        commands.set("hello","hello world");
        String str = commands.get("hello");
        System.out.println(str);

        /**
         * 异步执行命令
         */
        RedisAdvancedClusterAsyncCommands<String,String> asyncCommands =
connect.async();
        RedisFuture<String> future = asyncCommands.get("hello");

        try {
            String str1 = future.get();//此方法会阻塞到获取到数据为止
            System.out.println(str1);
        } catch (InterruptedException e) {
            e.printStackTrace();
        } catch (ExecutionException e) {
            e.printStackTrace();
        }

        connect.close();
        client.shutdown();
```

```
    }
}
```

二 使用spring-data-redis

> 此内为spring-data-redis+lettuce的方式操作redis,也可以在springboot中使用此配置

1 pom文件

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.qianfeng</groupId>
    <artifactId>reids-lettuce</artifactId>
    <version>1.0-SNAPSHOT</version>
    <dependencies>


        <dependency>
            <groupId>org.springframework.data</groupId>
            <artifactId>spring-data-redis</artifactId>
            <version>2.0.6.RELEASE</version>
        </dependency>
        <dependency>
            <groupId>io.lettuce</groupId>
            <artifactId>lettuce-core</artifactId>
            <version>5.0.5.RELEASE</version>
        </dependency>
    </dependencies>

</project>
```

2 配置类

2.1 单机版配置

```
package com.qianfeng.redis.lettuce;

//
```

```
//                            _ooOoo_
//                           o8888888o
//                           88" . "88
//                           (| -_- |)
//                            O\ = /O
//                        ____/`---'\____
//                      .   ' \\| |// `.
//                       / \\||| : |||// \
//                     / _||||| -:- |||||- \
//                       | | \\\ - /// | |
//                     | \_| ''\---/'' | |
//                      \ .-\__ `-` ___/-. /
//                   ___`. .' /--.--\ `. . __
//                ."" '< `.___\_<|>_/___.' >'"".
//               | | : `- \`.;`\ _ /`;.`/ - ` : | |
//                 \ \ `-. \_ __\ /__ _/ .-` / /
//         ======`-.____`-.___\_____/___.-`____.-'======
//                            `=---='
//
//         .............................................
//                  佛祖镇楼                  BUG辟易
//          佛曰:
//                  写字楼里写字间，写字间里程序员；
//                  程序人员写程序，又拿程序换酒钱。
//                  酒醒只在网上坐，酒醉还来网下眠；
//                  酒醉酒醒日复日，网上网下年复年。
//                  但愿老死电脑间，不愿鞠躬老板前；
//                  奔驰宝马贵者趣，公交自行程序员。
//                  别人笑我忒疯癫，我笑自己命太贱；
//
```

```
import io.lettuce.core.ClientOptions;
import io.lettuce.core.RedisClient;
import io.lettuce.core.RedisURI;
import io.lettuce.core.api.StatefulRedisConnection;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

/**
 * Created by jackiechan on 2019-04-21/09:29
 *
 * @Author jackiechan
 */
@Configuration
public class RedisClientConfig {
    /**
     * redis服务端地址对象
     * @return
```

```
     */
    @Bean
    public RedisURI redisURI() {
        RedisURI redisURI =
RedisURI.create("redis://redis001@redis.qfjava.cn:8400");//可以使用这一行代替上面
两行,指定redis的服务器地址和密码
        return redisURI;
    }

    /**
     * 连接选项
     * @return
     */
    @Bean
    public ClientOptions clientOptions() {
        return ClientOptions.builder().autoReconnect(true).build();//自动重连
    }

    /**
     * 创建redis客户端
     * @return
     */
    @Bean
    public RedisClient redisClient(RedisURI redisURI) {

        return RedisClient.create(redisURI);

    }

    @Bean
    public StatefulRedisConnection<String, String>
stringStringStatefulRedisConnection(RedisClient redisClient) {
        return redisClient.connect();
    }
}
```

2.2 集群版本配置

```
package com.qianfeng.redis.lettuce;

//
//                            _ooOoo_
//                           o8888888o
//                           88" . "88
//                           (| -_- |)
//                            O\ = /O
//                        ____/`---'\____
//                      .   ' \\| |// `.
```

```
//                       / \\||| : |||// \
//                     / _||||| -:- |||||- \
//                       | | \\\ - /// | |
//                     | \_| ''\---/'' | |
//                      \ .-\__ `-` ___/-. /
//                   ___`. .' /--.--\ `. . __
//                ."" '< `.___\_<|>_/___.' >'"".
//               | | : `- \`.;`\ _ /`;.`/ - ` : | |
//                 \ \ `-. \_ __\ /__ _/ .-` / /
//         ======`-.____`-.___\_____/___.-`____.-'======
//                            `=---='
//
//         .............................................
//                  佛祖镇楼                  BUG辟易
//          佛曰:
//                  写字楼里写字间，写字间里程序员；
//                  程序人员写程序，又拿程序换酒钱。
//                  酒醒只在网上坐，酒醉还来网下眠；
//                  酒醉酒醒日复日，网上网下年复年。
//                  但愿老死电脑间，不愿鞠躬老板前；
//                  奔驰宝马贵者趣，公交自行程序员。
//                  别人笑我忒疯癫，我笑自己命太贱；
//
```

```
import io.lettuce.core.RedisURI;
import io.lettuce.core.cluster.ClusterClientOptions;
import io.lettuce.core.cluster.RedisClusterClient;
import io.lettuce.core.cluster.api.StatefulRedisClusterConnection;
import io.lettuce.core.resource.ClientResources;
import org.springframework.context.annotation.Bean;

/**
 * Created by jackiechan on 2019-04-21/09:27
 *
 * @Author jackiechan
 */
//@Configuration
public class RedisClusterClientApplication {

    @Bean(name="clusterRedisUri")
    RedisURI clusterRedisUri(){
        return RedisURI.builder().withHost("127.0.0.1").withPort(7001)
                .withHost("127.0.0.1").withPort(7002)
                .withHost("127.0.0.1").withPort(7003)
                .withHost("127.0.0.1").withPort(7004)
                .withHost("127.0.0.1").withPort(7005)
                .withHost("127.0.0.1").withPort(7006).build();
    }
```

```
    //配置集群选项,自动重连,最多重定型1次
    @Bean
    ClusterClientOptions clusterClientOptions(){
        return
ClusterClientOptions.builder().autoReconnect(true).maxRedirects(1).build();
    }

    //创建集群客户端
    @Bean
    RedisClusterClient redisClusterClient(ClientResources clientResources,
ClusterClientOptions options, RedisURI clusterRedisUri){
        RedisClusterClient redisClusterClient =
RedisClusterClient.create(clientResources,clusterRedisUri);
        redisClusterClient.setOptions(options);
        return redisClusterClient;
    }

    //集群连接
    @Bean(destroyMethod = "close")
    StatefulRedisClusterConnection<String,String>
statefulRedisClusterConnection(RedisClusterClient redisClusterClient){
        return redisClusterClient.connect();
    }
}
```

3 测试类

```
package com.qianfeng.redis.lettuce;

//
//                            _ooOoo_
//                           o8888888o
//                           88" . "88
//                           (| -_- |)
//                            O\ = /O
//                        ____/`---'\____
//                      .   ' \\| |// `.
//                       / \\||| : |||// \
//                     / _||||| -:- |||||- \
//                       | | \\\ - /// | |
//                     | \_| ''\---/'' | |
//                      \ .-\__ `-` ___/-. /
//                   ___`. .' /--.--\ `. . __
//                ."" '< `.___\_<|>_/___.' >'"".
//               | | : `- \`.;`\ _ /`;.`/ - ` : | |
```

```
//                  \ \ `-. \_ __\ /__ _/ .-` / /
//          ======`-.____`-.___\_____/___.-`____.-'======
//                             `=---='
//
//          .............................................
//                   佛祖镇楼                  BUG辟易
//           佛曰:
//                   写字楼里写字间，写字间里程序员；
//                   程序人员写程序，又拿程序换酒钱。
//                   酒醒只在网上坐，酒醉还来网下眠；
//                   酒醉酒醒日复日，网上网下年复年。
//                   但愿老死电脑间，不愿鞠躬老板前；
//                   奔驰宝马贵者趣，公交自行程序员。
//                   别人笑我忒疯癫，我笑自己命太贱；
//


import io.lettuce.core.api.StatefulRedisConnection;
import io.lettuce.core.api.sync.RedisCommands;
import
org.springframework.context.annotation.AnnotationConfigApplicationContext;

/**
 * Created by jackiechan on 2019-04-21/09:28
 *
 * @Author jackiechan
 */
public class DemoApplication {
    public static void main(String[] args) {

        //测试单机版
        AnnotationConfigApplicationContext context = new
AnnotationConfigApplicationContext(RedisClientConfig.class);
        StatefulRedisConnection connection =
context.getBean(StatefulRedisConnection.class);//获取连接对象
        RedisCommands<String, String> commands = connection.sync();
        String hello = commands.get("hello");
        System.out.println(hello);

        //-------------------------------------
        //测试集群版
        // AnnotationConfigApplicationContext context = new
AnnotationConfigApplicationContext(RedisClusterClientApplication.class);
//        StatefulRedisClusterConnection connection =
(StatefulRedisClusterConnection)
context.getBean("statefulRedisClusterConnection");//获取连接对象
//
//        RedisAdvancedClusterCommands<String,String> clusterCommands =
connection.sync();
```

```
//        String str = clusterCommands.get("hello");
//
//        System.out.println(str);


    }
}
```

三 使用 Springboot-starter

> 此方式主要是使用spring-boot-starter-data-redis来自动配置

1 pom文件

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.qianfeng.commons</groupId>
    <artifactId>commons-redis</artifactId>
    <version>1.0</version>

    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.0.6.RELEASE</version>
    </parent>


    <dependencies>
<!--
spring整合redis 的依赖包,内部使用的是lettuce来操作redis,比jedis效率更高
-->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-redis</artifactId>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
```

```
    </dependencies>
</project>
```

2 application.yml

```
spring:
  redis:
    #redis机器ip
    host: redis.qfjava.cn
    #redis端口
    port: 8400
    #redis密码
    password: redis001
    #redis超时时间（毫秒），如果不设置，取默认值2000
    timeout: 10000
    #最大空闲数
    maxIdle: 300
    #连接池的最大数据库连接数。设为0表示无限制,如果是jedis 2.4以后用redis.maxTotal
    #maxActive=600
    #控制一个pool可分配多少个jedis实例,用来替换上面的redis.maxActive,如果是jedis 2.4以
后用该属性
    maxTotal: 1000
    #最大建立连接等待时间。如果超过此时间将接到异常。设为-1表示无限制。
    maxWaitMillis: 1000
    #连接的最小空闲时间 默认1800000毫秒(30分钟)
    minEvictableIdleTimeMillis: 300000
    #每次释放连接的最大数目,默认3
    numTestsPerEvictionRun: 1024
    #逐出扫描的时间间隔(毫秒) 如果为负数,则不运行逐出线程, 默认-1
    timeBetweenEvictionRunsMillis: 30000
    #是否在从池中取出连接前进行检验,如果检验失败,则从池中去除连接并尝试取出另一个
    testOnBorrow: true
    #在空闲时检查有效性, 默认false
    testWhileIdle: true

  #redis集群配置
#spring.redis.cluster.nodes=192.168.1.1:7001,192.168.1.1:7002,192.168.1.1:7003
,192.168.1.1:7004,192.168.1.1:7005,192.168.1.1:7006
#spring.redis.cluster.max-redirects=3
  #哨兵模式
#spring.redis.sentinel.master
#spring.redis.sentinel.nodes
```

3 配置类

```
package com.qianfeng.commons.redis.config;

//
//                            _ooOoo_
//                           o8888888o
//                           88" . "88
//                           (| -_- |)
//                            O\ = /O
//                        ____/`---'\____
//                      .   ' \\| |// `.
//                       / \\||| : |||// \
//                     / _||||| -:- |||||- \
//                       | | \\\ - /// | |
//                     | \_| ''\---/'' | |
//                      \ .-\__ `-` ___/-. /
//                   ___`. .' /--.--\ `. . __
//                ."" '< `.___\_<|>_/___.' >'"".
//               | | : `- \`.;`\ _ /`;.`/ - ` : | |
//                 \ \ `-. \_ __\ /__ _/ .-` / /
//         ======`-.____`-.___\_____/___.-`____.-'======
//                            `=---='
//
//         .............................................
//                  佛祖镇楼                  BUG辟易
//          佛曰:
//                  写字楼里写字间，写字间里程序员；
//                  程序人员写程序，又拿程序换酒钱。
//                  酒醒只在网上坐，酒醉还来网下眠；
//                  酒醉酒醒日复日，网上网下年复年。
//                  但愿老死电脑间，不愿鞠躬老板前；
//                  奔驰宝马贵者趣，公交自行程序员。
//                  别人笑我忒疯癫，我笑自己命太贱；
//



import org.springframework.boot.autoconfigure.AutoConfigureAfter;
import org.springframework.boot.autoconfigure.data.redis.RedisAutoConfiguration;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import
org.springframework.data.redis.connection.lettuce.LettuceConnectionFactory;
import org.springframework.data.redis.core.RedisTemplate;
import
org.springframework.data.redis.serializer.GenericJackson2JsonRedisSerializer;
import org.springframework.data.redis.serializer.StringRedisSerializer;
```

```
import java.io.Serializable;

/**
 * Created by jackiechan on 2019-04-21/08:55
 *
 * @Author jackiechan
 */

@Configuration
@AutoConfigureAfter(RedisAutoConfiguration.class)
public class RedisConfig {

    /**
     * 创建模板
     * @param redisConnectionFactory 这个对象springboot会自动创建
     * @return
     */
    @Bean
    public RedisTemplate<String, Serializable>
redisCacheTemplate(LettuceConnectionFactory redisConnectionFactory) {
        RedisTemplate<String, Serializable> template = new RedisTemplate<>();
        template.setKeySerializer(new StringRedisSerializer());//配置key的序列化方
式是字符串
        template.setValueSerializer(new GenericJackson2JsonRedisSerializer());//
配置value的序列化方式为转换为json,可以自己写序列化类来实现序列化
        template.setConnectionFactory(redisConnectionFactory);
        return template;
    }
}
```

4 controller

> 此controller主要目的是测试用

```
package com.qianfeng.commons.redis.config;

//
//                            _ooOoo_
//                           o8888888o
//                           88" . "88
//                           (| -_- |)
//                            O\ = /O
//                        ____/`---'\____
//                      .   ' \\| |// `.
//                       / \\||| : |||// \
//                     / _||||| -:- |||||- \
```

```
//                       | | \\\ - /// | |
//                     | \_| ''\---/'' | |
//                      \ .-\__ `-` ___/-. /
//                   ___`. .' /--.--\ `. . __
//                ."" '< `.___\_<|>_/___.' >'"".
//               | | : `- \`.;`\ _ /`;.`/ - ` : | |
//                 \ \ `-. \_ __\ /__ _/ .-` / /
//         ======`-.____`-.___\_____/___.-`____.-'======
//                            `=---='
//
//         .............................................
//                  佛祖镇楼                  BUG辟易
//          佛曰:
//                  写字楼里写字间，写字间里程序员；
//                  程序人员写程序，又拿程序换酒钱。
//                  酒醒只在网上坐，酒醉还来网下眠；
//                  酒醉酒醒日复日，网上网下年复年。
//                  但愿老死电脑间，不愿鞠躬老板前；
//                  奔驰宝马贵者趣，公交自行程序员。
//                  别人笑我忒疯癫，我笑自己命太贱；
//


import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.redis.core.StringRedisTemplate;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

/**
 * Created by jackiechan on 2019-04-21/09:01
 *
 * @Author jackiechan
 */
@RestController
public class RedisC {
    @Autowired
    private StringRedisTemplate stringRedisTemplate;

    //添加
    @GetMapping(value="/redisAdd")
    public void saveRedis(){
        stringRedisTemplate.opsForValue().set("a","test");
    }

    //获取
    @GetMapping(value="/redisGet")
    public String getRedis() {
        return stringRedisTemplate.opsForValue().get("a");
    }
```

```
}
```

5 启动类

```
package com.qianfeng.commons.redis.config;

//
//                            _ooOoo_
//                           o8888888o
//                           88" . "88
//                           (| -_- |)
//                            O\ = /O
//                        ____/`---'\____
//                      .   ' \\| |// `.
//                       / \\||| : |||// \
//                     / _||||| -:- |||||- \
//                       | | \\\ - /// | |
//                     | \_| ''\---/'' | |
//                      \ .-\__ `-` ___/-. /
//                   ___`. .' /--.--\ `. . __
//                ."" '< `.___\_<|>_/___.' >'"".
//               | | : `- \`.;`\ _ /`;.`/ - ` : | |
//                 \ \ `-. \_ __\ /__ _/ .-` / /
//         ======`-.____`-.___\_____/___.-`____.-'======
//                            `=---='
//
//         .............................................
//                  佛祖镇楼                  BUG辟易
//          佛曰:
//                  写字楼里写字间，写字间里程序员；
//                  程序人员写程序，又拿程序换酒钱。
//                  酒醒只在网上坐，酒醉还来网下眠；
//                  酒醉酒醒日复日，网上网下年复年。
//                  但愿老死电脑间，不愿鞠躬老板前；
//                  奔驰宝马贵者趣，公交自行程序员。
//                  别人笑我忒疯癫，我笑自己命太贱；
//



import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

/**
 * Created by jackiechan on 2019-04-21/09:02
```

```
 *
 * @Author jackiechan
 */
@SpringBootApplication
public class StartApp {
    public static void main (String[] args){
        SpringApplication.run(StartApp.class,args);
    }
}
```

6 测试

> 启动程序后访问接口,测试即可