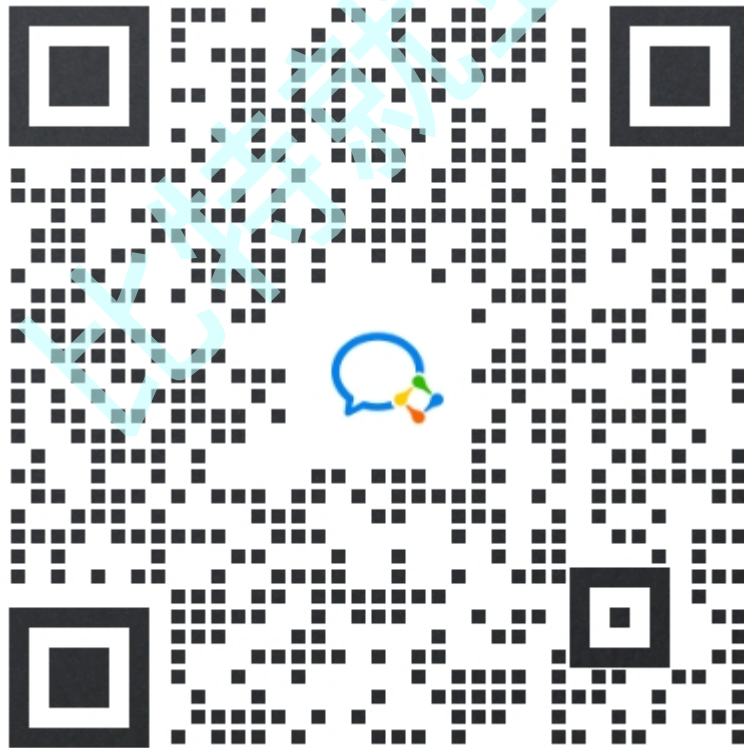


Redis Java 集成到 Spring Boot

版权说明

本“比特就业课”课程（以下简称“本课程”）的所有内容，包括但不限于文字、图片、音频、视频、软件、程序、数据库、设计、布局、界面等，均由本课程的开发者或授权方拥有版权。我们鼓励个人学习者使用本课程进行学习和研究。在遵守相关法律法规的前提下，个人学习者可以下载、浏览、学习本课程的内容，并为了个人学习、研究或教学目的而使用其中的材料。但请注意，未经我们明确授权，个人学习者不得将本课程的内容用于任何商业目的，包括但不限于销售、转让、许可或以其他方式从中获利。此外，个人学习者也不得擅自修改、复制、传播、展示、表演或制作本课程内容的衍生作品。任何未经授权的使用均属侵权行为，我们将依法追究法律责任。如果您希望以其他方式使用本课程的内容，包括但不限于引用、转载、摘录、改编等，请事先与我们联系，获取书面授权。感谢您对“比特就业课”课程的关注与支持，我们将持续努力，为您提供更好的学习体验。特此说明。比特就业课版权所有方

对比特课程感兴趣，可以联系这个微信。



代码 & 板书链接

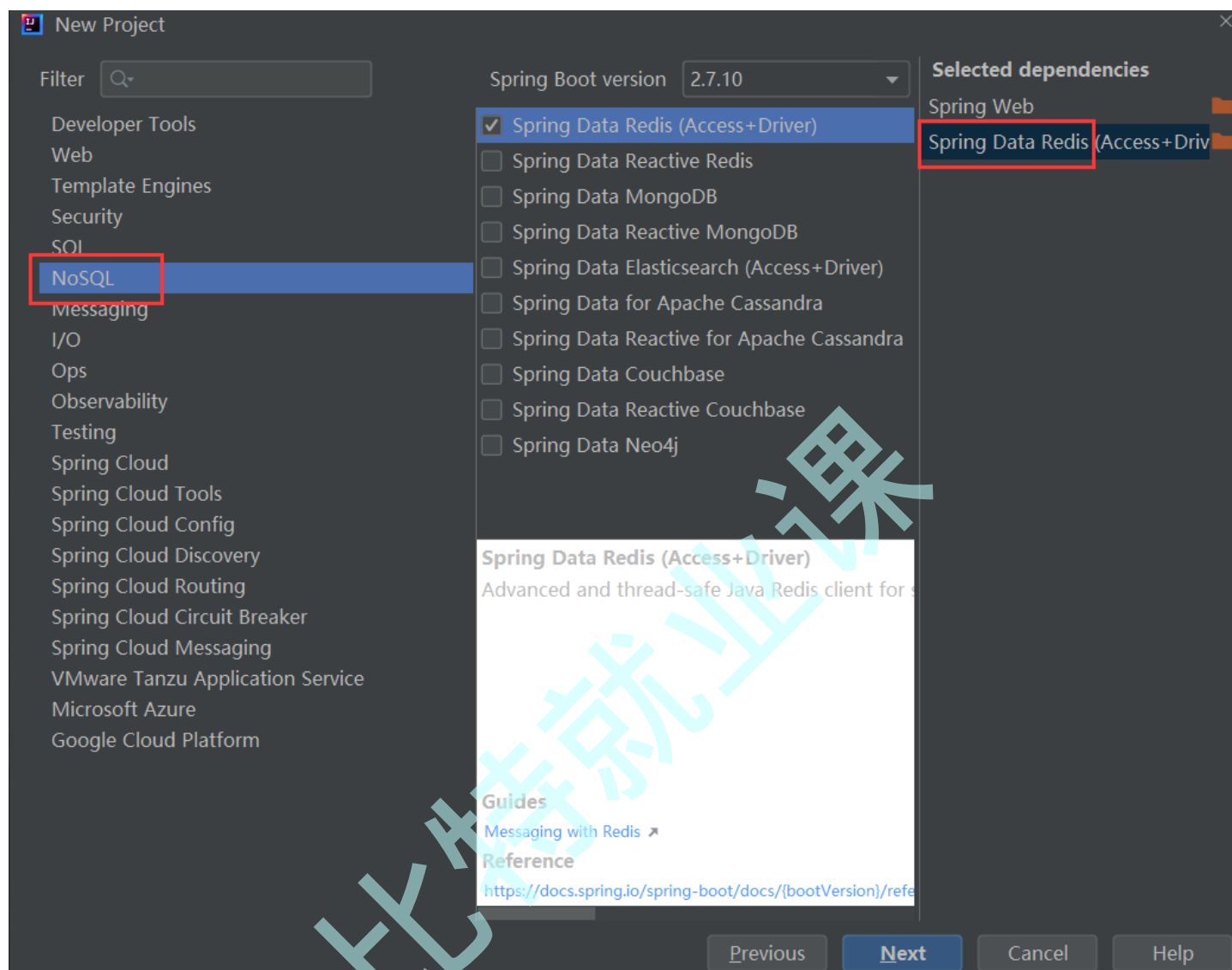
<https://gitee.com/HGtz2222/bitproject/tree/master/redis>

使用 Spring Boot 连接 Redis 单机

创建项目

勾选 NoSQL 中的 `Spring Data Redis`

当然, 把 Web 中的 Spring Web 也勾选一下. 方便写接口进行后续测试.



配置 redis 服务地址

在 application.yml 中配置.

```
1 spring:
2   redis:
3     host: 127.0.0.1
4     port: 8888
```

创建 Controller

创建 `MyController` 类

由于当前只是写简单的测试代码, 我们就不进行分层了. 就只创建个简单的 Controller 即可.

```
1 @RestController
2 public class MyController {
3     @Autowired
4     private StringRedisTemplate redisTemplate;
5
6 }
```

此处需要引入 `StringRedisTemplate` 实例.

使用 String

```
1 @GetMapping("/testString")
2 @ResponseBody
3 public String testString() {
4     redisTemplate.opsForValue().set("key", "value");
5     String value = redisTemplate.opsForValue().get("key");
6     System.out.println(value);
7
8     redisTemplate.delete("key");
9     return "OK"
10 }
```

使用 List

```
1 @GetMapping("/testList")
2 @ResponseBody
3 public String testList() {
4     redisTemplate.opsForList().leftPush("key", "a");
5     redisTemplate.opsForList().leftPushAll("key", "b", "c", "d");
6     List<String> values = redisTemplate.opsForList().range("key", 1, 2);
7     System.out.println(values);
8
9     redisTemplate.delete("key");
10     return "OK";
11 }
```

使用 Hash

```
1 @GetMapping("/testHashMap")
2 @ResponseBody
3 public String testHashMap() {
4     redisTemplate.opsForHash().put("key", "name", "zhangsan");
5     String value = (String) redisTemplate.opsForHash().get("key", "name");
6     System.out.println(value);
7     redisTemplate.opsForHash().delete("key", "name");
8     boolean ok = redisTemplate.opsForHash().hasKey("key", "name");
9     System.out.println(ok);
10
11     redisTemplate.delete("key");
12     return "OK";
13 }
```

使用 Set

```
1 @GetMapping("/testSet")
2 @ResponseBody
3 public String testSet() {
4     redisTemplate.opsForSet().add("key", "aaa", "bbb", "ccc");
5     boolean ok = redisTemplate.opsForSet().isMember("key", "aaa");
6     System.out.println(ok);
7     redisTemplate.opsForSet().remove("key", "aaa");
8     long n = redisTemplate.opsForSet().size("key");
9     System.out.println(n);
10
11     redisTemplate.delete("key");
12     return "OK";
13 }
```

使用 ZSet

```
1 @GetMapping("/testZSet")
2 @ResponseBody
3 public String testZSet() {
4     redisTemplate.opsForZSet().add("key", "吕布", 100);
5     redisTemplate.opsForZSet().add("key", "赵云", 98);
6     redisTemplate.opsForZSet().add("key", "典韦", 95);
7
8     Set<String> values = redisTemplate.opsForZSet().range("key", 0, 2);
9     System.out.println(values);
10
11     long n = redisTemplate.opsForZSet().count("key", 95, 100);
12     System.out.println(n);
13
14     redisTemplate.delete("key");
15     return "OK";
16 }
```

执行结果

运行程序, 构造上述请求, 即可在服务器的控制台中看到执行结果.

使用 Spring Boot 连接 Redis 集群

配置 redis 集群地址

```
1 spring:
2   redis:
3     cluster:
4       nodes:
5         - 172.30.0.101:6379
6         - 172.30.0.102:6379
7         - 172.30.0.103:6379
8         - 172.30.0.104:6379
9         - 172.30.0.105:6379
10        - 172.30.0.106:6379
11        - 172.30.0.107:6379
12        - 172.30.0.108:6379
13        - 172.30.0.109:6379
14     lettuce:
15       cluster:
16         refresh:
```

```
17     adaptive: true
18     period: 2000
```



下方的 lettuce 系列配置, 目的是为了自动刷新集群的拓扑结构. 当集群中有节点宕机/加入新节点之后, 我们的代码能够自动感知到集群的变化.

改完配置之后, 其他代码无需做出任何调整, 直接就可以正常运行了.



注意!

由于上述 ip 都是 docker 容器的 ip, 在 windows 主机上不能直接访问.

因此需要把程序打成 jar 包, 部署到 linux 上, 再通过 `java -jar [jar包名]` 的方式执行.

小结

Spring Boot 2 系列内置的 Redis 是 Lettuce, 和 Jedis 的使用还是存在一定的差异.

对于 Jedis 来说, 各个方法和 Redis 的命令基本是一致的.

而集成到 Spring Boot 之后, 接口上和原始 Redis 命令存在部分差别, 但是使用起来也并不困难, 只要大家熟悉 Redis 的基本操作, 还是很容易可以通过方法名字理解用法的.