

Redis, 一站式高性能存储方案

牛客Java高级工程师 第四章

► 1. Redis入门

- Redis是一款基于键值对的NoSQL数据库，它的值支持多种数据结构：
字符串(strings)、哈希(hashes)、列表(lists)、集合(sets)、有序集合(sorted sets)等。
- Redis将所有数据都存放在内存中，所以它的读写性能十分惊人。
同时，Redis还可以将内存中的数据以快照或日志的形式保存到硬盘上，以保证数据的安全性。
- Redis典型的应用场景包括：缓存、排行榜、计数器、社交网络、消息队列等。

<https://redis.io>

<https://github.com/microsoftarchive/redis>

► 2. Spring整合Redis

- 引入依赖
 - spring-boot-starter-data-redis
- 配置Redis
 - 配置数据库参数
 - 编写配置类，构造RedisTemplate
- 访问Redis
 - `redisTemplate.opsForValue()`
 - `redisTemplate.opsForHash()`
 - `redisTemplate.opsForList()`
 - `redisTemplate.opsForSet()`
 - `redisTemplate.opsForZSet()`

Spring Data Redis 2.1.8

Overview

Learn

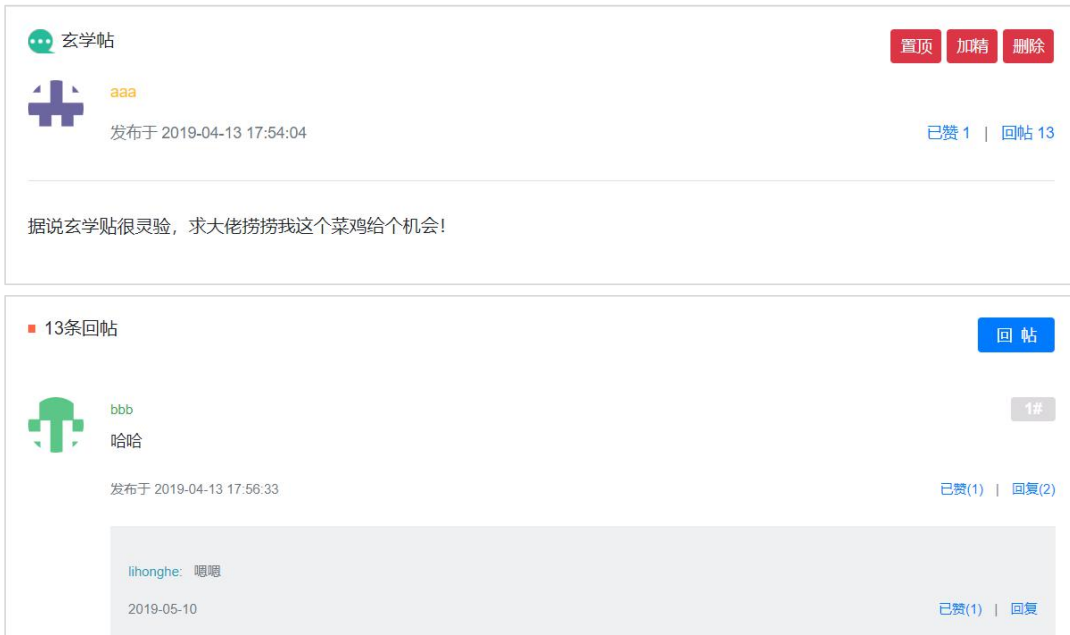
Spring Data Redis, part of the larger Spring Data family, provides easy configuration and access to Redis from Spring applications. It offers both low-level and high-level abstractions for interacting with the store, freeing the user from infrastructural concerns.

Features

- Connection package as low-level abstraction across multiple Redis drivers/connectors ([Jedis](#) and [Lettuce](#). Support for [JRedis](#) and [SRP](#) is deprecated.)
- [Exception](#) translation to Spring's portable Data Access exception hierarchy for Redis driver exceptions
- [RedisTemplate](#) that provides a high level abstraction for performing various Redis operations, exception translation and serialization support
- [Pubsub](#) support (such as a [MessageListenerContainer](#) for message-driven POJOs)
- [Redis Sentinel](#) and [Redis Cluster](#) support
- JDK, String, JSON and Spring Object/XML mapping [serializers](#)
- JDK [Collection](#) implementations on top of Redis
- Atomic [counter](#) support classes
- Sorting and Pipelining functionality
- Dedicated support for SORT, SORT/GET pattern and returned bulk values
- Redis [implementation](#) for Spring 3.1 cache abstraction
- Automatic implementation of [Repository](#) interfaces including support for custom finder methods using [@EnableRedisRepositories](#)
- CDI support for repositories

▶ 3. 点赞

- 点赞
 - 支持对帖子、评论点赞。
 - 第1次点赞，第2次取消点赞。
- 首页点赞数量
 - 统计帖子的点赞数量。
- 详情页点赞数量
 - 统计点赞数量。
 - 显示点赞状态。



► 4. 我收到的赞

- 重构点赞功能
 - 以用户为key，记录点赞数量
 - increment(key), decrement(key)
- 开发个人主页
 - 以用户为key，查询点赞数量



► 5. 关注、取消关注

- 需求
 - 开发关注、取消关注功能。
 - 统计用户的关注数、粉丝数。
- 关键
 - 若A关注了B，则A是B的Follower（粉丝），B是A的Followee（目标）。
 - 关注的目标可以是用户、帖子、题目等，在实现时将这些目标抽象为实体。

► 6. 关注列表、粉丝列表

- 业务层
 - 查询某个用户关注的人，支持分页。
 - 查询某个用户的粉丝，支持分页。
- 表现层
 - 处理 “查询关注的人” 、 “查询粉丝” 请求。
 - 编写 “查询关注的人” 、 “查询粉丝” 模板。

► 7. 优化登录模块

- 使用Redis存储验证码
 - 验证码需要频繁的申请与刷新，对性能要求较高。
 - 验证码不需永久保存，通常在很短的时间后就会失效。
 - 分布式部署时，存在Session共享的问题。
- 使用Redis存储登录凭证
 - 处理每次请求时，都要查询用户的登录凭证，访问的频率非常高。
- 使用Redis缓存用户信息
 - 处理每次请求时，都要根据凭证查询用户信息，访问的频率非常高。

Thanks

