

笔记本：数据存储

创建时间：2018\11\1 星期四 14:52

更新时间：2018\11\1 星期四 21:29

作者：804790605@qq.com

标签：redis

当前项目中需要使用Redis处理token，所以使用了Redis。接下来聊聊Redis。

一、概述

Redis 是一个开源（BSD许可）的，**内存中的数据结构存储系统**，它可以用作**数据库、缓存和消息中间件**。它支持多种类型的数据结构，如 字符串（strings），散列（hashes），列表（lists），集合（sets），有序集合（sorted sets）与范围查询，bitmaps，hyperloglogs 和 地理空间（geospatial）索引半径查询。Redis 内置了 复制（replication），LUA脚本（Lua scripting），LRU驱动事件（LRU eviction），事务（transactions）和不同级别的 磁盘持久化（persistence），并通过 Redis哨兵（Sentinel）和自动 分区（Cluster）提供高可用性（high availability）。从2010年3月15日起，Redis的开发工作由VMware主持。从2013年5月开始，Redis的开发由Pivotal赞助。

redis是一个**key-value存储系统**。和Memcached类似，它支持存储的value类型相对更多，包括 **string(字符串)**、**list(链表)**、**set(集合)**、**zset(sorted set --有序集合)**和**hash (哈希类型)**。这些数据类型都支持push/pop、add/remove及取交集并集和差集及更丰富的操作，而且这些操作都是原子性的。在此基础上，redis支持各种不同方式的排序。与memcached一样，为了保证效率，数据都是缓存在内存中。区别的是redis会周期性的把更新的数据写入磁盘或者把修改操作写入追加的记录文件，并且在此基础上实现了master-slave(主从)同步。

Redis 是一个高性能的key-value数据库。redis的出现，很大程度补偿了memcached这类key/value存储的不足，在部分场合可以对关系数据库起到很好的补充作用。它提供了Java，C/C++，C#，PHP，JavaScript，Perl，Object-C，Python，Ruby，Erlang等客户端，使用很方便。[1]

Redis支持主从同步。数据可以从主服务器向任意数量的从服务器上同步，从服务器可以是关联其他从服务器的主服务器。这使得Redis可执行单层树复制。存盘可以有意无意的对数据进行写操作。由于完全实现了发布/订阅机制，使得从数据库在任何地方同步树时，可订阅一个频道并接收主服务器完整的消息发布记录。同步对读取操作的可扩展性和数据冗余很有帮助。

二、安装

由于我使用的是window系统，所以本篇幅不做其他系统的安装详细描述，大家可以网上搜索一下，很多资源。

1、下载安装包

GitHub, Inc. [US] <https://github.com/MicrosoftArchive/redis/releases>

小白一键重装 百度 淘宝 京东 天猫 系统之家 电影大全 intellij git ancun 前端 springboot project tars JPA my 从IE中

Releases Tags

Pre-release

win-3.2.100
def0757

3.2.100

enricogior released this on 1 Jul 2016 · [1208 commits](#) to 3.0 since this release

Assets 4

- Redis-x64-3.2.100.msi
- Redis-x64-3.2.100.zip
- Source code (zip)
- Source code (tar.gz)

This is the first release of Redis on Windows 3.2.

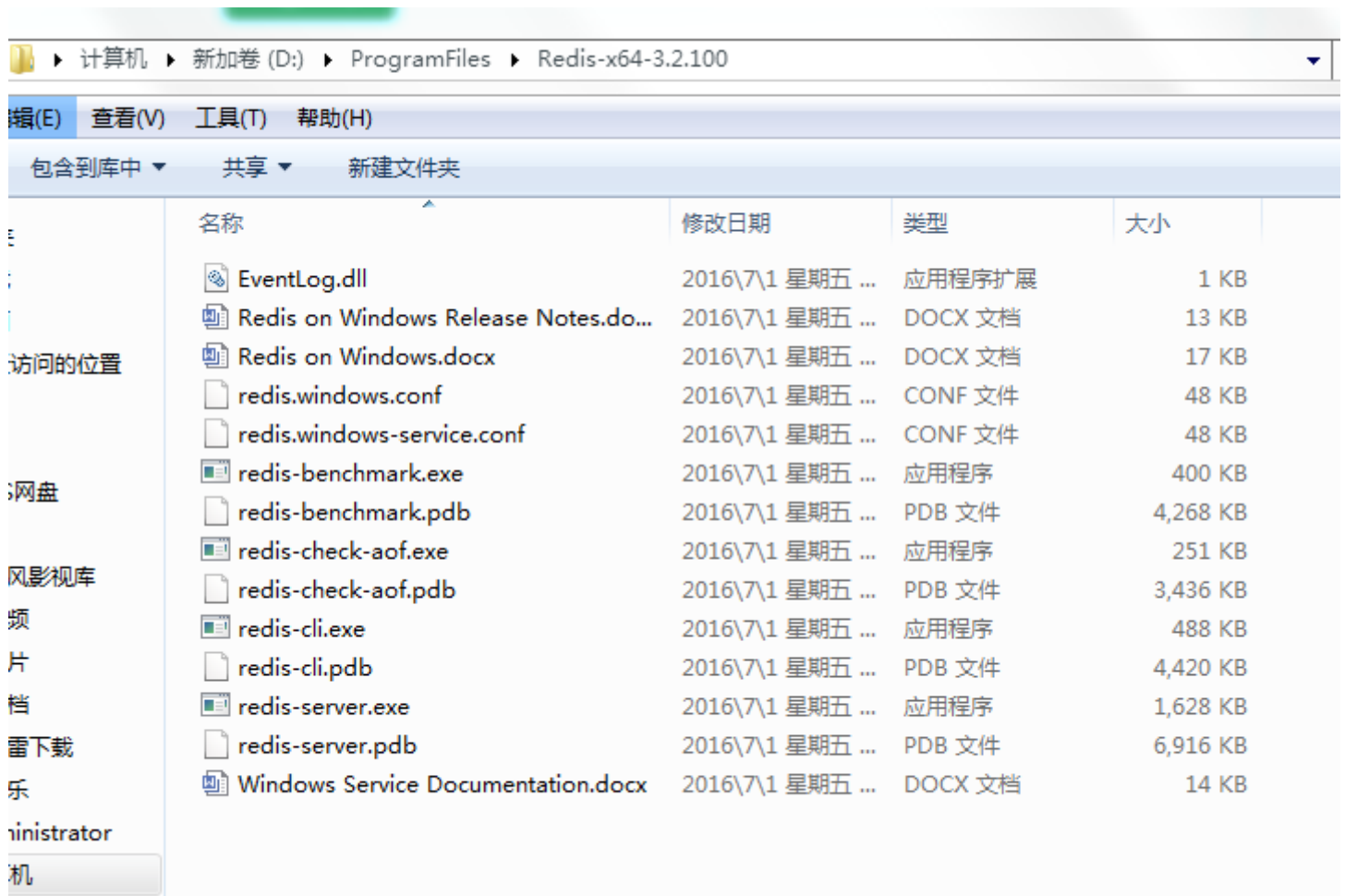
This release is based on antirez/redis/3.2.1 plus some Windows specific fixes. It tests but it hasn't been tested in a production environment.

Therefore, **before considering using this release in production, make sure to test environment.**

See the [release notes](#) for details.

2、解压缩

当前我下载的是zip压缩包，大家也可以下载msi安装版。解压出来如下：



3、运行

打开一个 cmd 窗口 使用 cd 命令切换解压目录到 D:\ProgramFiles\Redis-x64-3.2.100 运行如下命令：

```
redis-server.exe redis.windows.conf
```

如图：

（注意：如果想方便的话，可以把 redis 的路径加到系统的环境变量<path>里，这样就省得再输路径了，后面的那个 redis.windows.conf 可以省略，如果省略，会启用默认的。输入之后，会显示如下界面：）

```
管理员: C:\Windows\System32\cmd.exe - redis-server.exe redis.windows.conf
卷的序列号是 1876-CBE5

D:\Program Files 的目录
2018\10\30 周二 19:42 <DIR> .
2018\10\30 周二 19:42 <DIR> ..
2018\10\30 周二 19:42 <DIR> 360se
2018\09\18 周二 13:38 <DIR> Sublime Text 3
0 个文件 0 字节
4 个目录 98,095,403,008 可用字节

D:\Program Files>cd ..
D:\>cd ProgramFiles
D:\ProgramFiles>cd Redis-x64-3.2.100
D:\ProgramFiles\Redis-x64-3.2.100>redis-server.exe redis.windows.conf

Redis 3.2.100 (00000000/0) 64 bit

Running in standalone mode
Port: 6379
PID: 33408

http://redis.io

[33408] 01 Nov 15:19:43.555 # Server started, Redis version 3.2.100
[33408] 01 Nov 15:19:43.556 * The server is now ready to accept connections on port 6379

半:
```

4、连接使用

这时候另启一个 cmd 窗口，原来的**不要关闭**，不然就无法访问服务端了。
还是切换到 redis 目录D:\ProgramFiles\Redis-x64-3.2.100下，运行命令：

```
redis-cli.exe -h 127.0.0.1 -p 6379
```

//默认使用的端口是6379，当然可以在启动服务的时候使用相对应的参数进行修改启动，默认是无密码的

```
管理员: C:\windows\system32\cmd.exe - redis-cli.exe -h 127.0.0.1 -p 6379
D:\ProgramFiles\Redis-x64-3.2.100>
D:\ProgramFiles\Redis-x64-3.2.100>
D:\ProgramFiles\Redis-x64-3.2.100>redis-cli.exe -h 127.0.0.1 -p 6379
127.0.0.1:6379> set actionkey hello
OK
127.0.0.1:6379> set actionkey hello redis world
(error) ERR syntax error
127.0.0.1:6379> update actionkey helloRedisWorld
(error) ERR unknown command 'update'
127.0.0.1:6379> set actionkey HelloRedisWorld
OK
127.0.0.1:6379> get actionkey
"HelloRedisWorld"
127.0.0.1:6379>
```

至此，Redis安装步骤结束。由于太久没有使用，所以命令有些已经忘记了。望见谅，不过我们可以继续熟悉。

三、使用

1、数据类型使用

Redis支持五种数据类型：string（字符串），hash（哈希），list（列表），set（集合）及zset(sorted set：有序集合）。

String（字符串）

string 是 redis 最基本的类型，你可以理解成与 Memcached 一模一样的类型，一个 key 对应一个 value。

string 类型是二进制安全的。意思是 redis 的 string 可以包含任何数据。比如jpg图片或者序列化的对象。

string 类型是 Redis 最基本的数据类型，string 类型的值最大能存储 512MB。

实例

```
redis 127.0.0.1:6379> SET name "zhengfeng"
OK
redis 127.0.0.1:6379> GET name
"zhengfeng"
```

在以上实例中我们使用了 Redis 的 **SET** 和 **GET** 命令。键为 name，对应的值为**zhengfeng**。

注意：一个键最大能存储512MB。

Hash (哈希)

Redis hash 是一个键值(key=>value)对集合。

Redis hash 是一个 string 类型的 field 和 value 的映射表，hash 特别适合于存储对象。

实例

```
redis> HMSET myhash field1 "Hello" field2 "World""OK"
redis> HGET myhash field1
"Hello"
redis> HGET myhash field2
"World"
```

实例中我们使用了 Redis **HMSET**, **HGET** 命令，**HMSET** 设置了两个 **field=>value** 对, **HGET** 获取对应 **field** 对应的 **value**。

每个 hash 可以存储 $2^{32} - 1$ 键值对 (40多亿)。

List (列表)

Redis 列表是简单的字符串列表，按照插入顺序排序。你可以添加一个元素到列表的头部 (左边) 或者尾部 (右边)。

实例

```
redis 127.0.0.1:6379> lpush zhengfeng redis
(integer) 1
redis 127.0.0.1:6379> lpush zhengfeng mongodb
(integer) 2
redis 127.0.0.1:6379> lpush zhengfeng rabbitmq
(integer) 3
redis 127.0.0.1:6379> lrange zhengfeng 0 101) "rabbitmq"2) "mongodb"3) "redis"
redis 127.0.0.1:6379>
```

列表最多可存储 $2^{32} - 1$ 元素 (4294967295, 每个列表可存储40多亿)。

Set (集合)

Redis的Set是string类型的无序集合。

集合是通过哈希表实现的，所以添加，删除，查找的复杂度都是O(1)。

sadd 命令

添加一个 string 元素到 key 对应的 set 集合中，成功返回1，如果元素已经在集合中返回 0，如果 key 对应的 set 不存在则返回错误。

```
sadd key member
```

实例

```
redis 127.0.0.1:6379> sadd zhengfeng redis
(integer) 1
redis 127.0.0.1:6379> sadd zhengfeng mongodb
(integer) 1
redis 127.0.0.1:6379> sadd zhengfeng rabbitmq
(integer) 1
redis 127.0.0.1:6379> sadd zhengfeng rabbitmq
(integer) 0
redis 127.0.0.1:6379> smembers zhengfeng

1) "redis"2) "rabbitmq"3) "mongodb"
```

注意：以上实例中 rabbitmq 添加了两次，但根据集合内元素的唯一性，第二次插入的元素将被忽略。

集合中最大的成员数为 $2^{32} - 1$ (4294967295, 每个集合可存储40多亿个成员)。

zset(sorted set : 有序集合)

Redis zset 和 set 一样也是string类型元素的集合,且不允许重复的成员。

不同的是每个元素都会关联一个double类型的分数。redis正是通过分数来为集合中的成员进行从小到大的排序。

zset的成员是唯一的,但分数(score)却可以重复。

zadd 命令

添加元素到集合，元素在集合中存在则更新对应score

```
zadd key score member
```

实例

```
redis 127.0.0.1:6379> zadd zhengfeng0 redis
(integer) 1
redis 127.0.0.1:6379> zadd zhengfeng 0 mongodb
(integer) 1
redis 127.0.0.1:6379> zadd zhengfeng 0 rabbitmq
(integer) 1
redis 127.0.0.1:6379> zadd zhengfeng 0 rabbitmq
(integer) 0
redis 127.0.0.1:6379> > ZRANGEBYSCORE zhengfeng 0 10001) "mongodb"2) "rabbitmq"3) "redis"
```

各个数据类型应用场景：

类型

简介

特性

场景

String(字符串)	二进制安全	可以包含任何数据,比如jpg图片或者序列化的对象,一个键最大能存储512M	---
Hash(字典)	键值对集合,即编程语言中的Map类型	适合存储对象,并且可以像数据库中update一个属性一样只修改某一项属性值(Memcached中需要取出整个字符串反序列化成对象修改完再序列化存回去)	存储、读取、修改用户属性
List(列表)	链表(双向链表)	增删快,提供了操作某一段元素的API	1,最新消息排行等功能(比如朋友圈的时间线) 2,消息队列
Set(集合)	哈希表实现,元素不重复	1、添加、删除,查找的复杂度都是O(1) 2、为集合提供了求交集、并集、差集等操作	1、共同好友 2、利用唯一性,统计访问网站的所有独立ip 3、好友推荐时,根据tag求交集,大于某个阈值就可以推荐
Sorted Set(有序集合)	将Set中的元素增加一个权重参数score,元素按score有序排列	数据插入集合时,已经进行天然排序	1、排行榜 2、带权重的消息队列

当然这只是一部分命令的使用，需要详细的还可以进入帮助文档去查询。

2、服务端命令

```

slect      #选择数据库(数据库编号0-15)
quit       #退出连接
info       #获得服务的信息与统计
monitor    #实时监控
config get  #获得服务配置
flushdb    #删除当前选择的数据库中的key
flushall   #删除所有数据库中的key

```

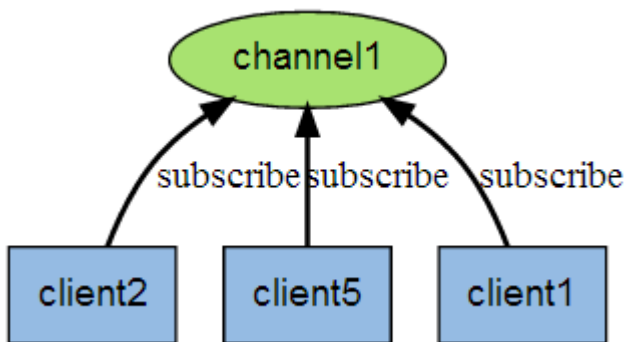


```
127.0.0.1:6379> select 0
OK
127.0.0.1:6379> info
# Server
redis_version:3.0.6
redis_git_sha1:00000000
redis_git_dirty:0
redis_build_id:347e3eeef5029f3
redis_mode:standalone
os:Linux 3.10.0-693.el7.x86_64 x86_64
arch_bits:64
multiplexing_api:epoll
gcc_version:4.8.5
process_id:31197
run_id:8b6ec6ad5035f5df0b94454e199511084ac6fb12
tcp_port:6379
uptime_in_seconds:8514
uptime_in_days:0
hz:10
lru_clock:14015928
config_file:/usr/local/redis/redis.conf
-----省略N行
127.0.0.1:6379> CONFIG GET 0
(empty list or set)
127.0.0.1:6379> CONFIG GET 15
(empty list or set)
```

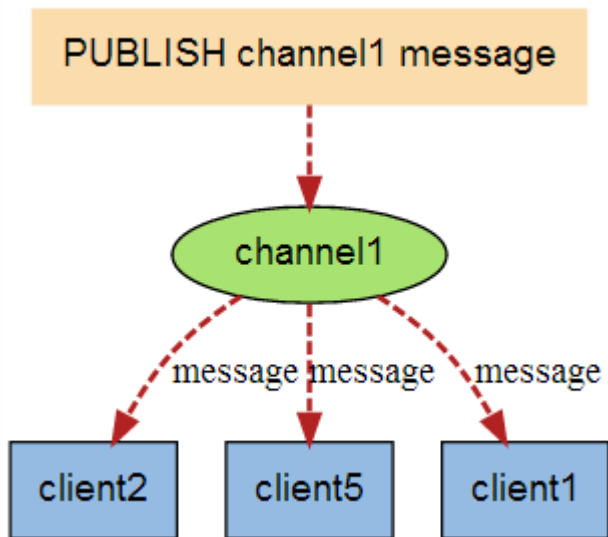
3、发布与预定

redis发布与订阅(pub/sub)是它的一种消息通信模式，一方发送信息，一方接收信息。

下图是三个客户端同时订阅同一个频道



下图是有新信息发送给频道1时，就会将消息发送给订阅它的三个客户端



四、项目使用

说的再多，不如动手一番。看看项目中如何使用。

1、准备

选择项目中支持Redis的插件工具。

a、Jedis，之前使用过这个插件，使用也挺方便的，不过这里不详细说明

b、spring-boot-starter-data-redis，当前项目使用的，去详细了解了解

2、配置

a、pom.xml

配置相对应的java依赖

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-redis</artifactId>
  <version>${spring.boot.version}</version>
</dependency>
```

b、application.yml

配置相对应的Redis服务信息。

```
redis:
  host: 192.168.0.XX #服务器的IP地址
  port: 6379
  password:
  database: 0
  timeout: 10s # 数据库连接超时时间，2.0 中该参数的类型为Duration，这里在配置的时候需要指明单位
  # 连接池配置，2.0中直接使用jedis或者lettuce配置连接池
  lettuce:
    pool:
```

```
# 最大空闲连接数
max-idle: 8
# 最小空闲连接数
min-idle: 0
# 等待可用连接的最大时间，负数为不限制
max-wait: -1s
# 最大活跃连接数，负数为不限制
max-active: 8
```

3、选择工具类使用

这个主要是根据redis存储的数据类型需求决定，key一般都是String，但是value可能不一样，一般有两种，String和 Object；

如果k-v都是String类型，我们可以直接用 StringRedisTemplate，这个是官方建议的，也是最方便的，直接导入即用，无需多余配置！

如果k-v是Object类型，则需要自定义 RedisTemplate，在这里我们都研究下！

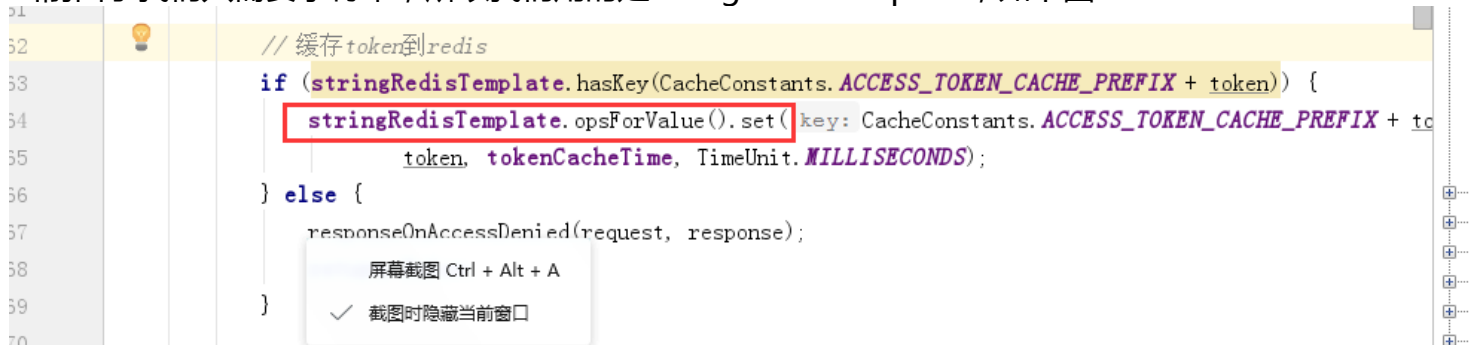
a、StringRedisTemplate

在此存入redis的类型为：**String:String**

常用操作：

```
stringRedisTemplate.opsForValue().set("test", "100", 60*10, TimeUnit.SECONDS); //向redis里存入数据和设置缓存时间
stringRedisTemplate.boundValueOps("test").increment(-1); //val做-1操作
stringRedisTemplate.opsForValue().get("test"); //根据key获取缓存中的val
stringRedisTemplate.boundValueOps("test").increment(1); //val +1
stringRedisTemplate.getExpire("test"); //根据key获取过期时间
stringRedisTemplate.getExpire("test", TimeUnit.SECONDS); //根据key获取过期时间并换算成指定单位
stringRedisTemplate.delete("test"); //根据key删除缓存
stringRedisTemplate.hasKey("546545"); //检查key是否存在，返回boolean值
stringRedisTemplate.opsForSet().add("red_123", "1", "2", "3"); //向指定key中存放set集合
stringRedisTemplate.expire("red_123", 1000, TimeUnit.MILLISECONDS); //设置过期时间
stringRedisTemplate.opsForSet().isMember("red_123", "1"); //根据key查看集合中是否存在指定数据
stringRedisTemplate.opsForSet().members("red_123"); //根据key获取set集合
```

当前由于我们只需要字符串，所以我们用的是StringRedisTemplate，如下图：



```
31
32 // 缓存token到redis
33 if (stringRedisTemplate.hasKey(CacheConstants.ACCESS_TOKEN_CACHE_PREFIX + token)) {
34     stringRedisTemplate.opsForValue().set(key: CacheConstants.ACCESS_TOKEN_CACHE_PREFIX + token, token, tokenCacheTime, TimeUnit.MILLISECONDS);
35 } else {
36     responseOnAccessDenied(request, response);
37 }
38
39
40
```

屏幕截图 Ctrl + Alt + A
✓ 截图时隐藏当前窗口

```

1 private StringRedisTemplate stringRedisTemplate;
2
3 if (stringRedisTemplate.hasKey(CacheConstants.ACCESS_TOKEN_CACHE_PREFIX + token)) {
4     stringRedisTemplate.opsForValue().set(CacheConstants.ACCESS_TOKEN_CACHE_PREFIX + token,
5     private StringRedisTemplate stringRedisTemplate;
6     stringRedisTemplate.opsForValue().set(CacheConstants.ACCESS_TOKEN_CACHE_PREFIX + token,
7     stringRedisTemplate.delete(CacheConstants.ACCESS_TOKEN_CACHE_PREFIX + token);
8     private StringRedisTemplate stringRedisTemplate;
9     Long count = stringRedisTemplate.opsForValue().increment(redisKey, 1);
10    private StringRedisTemplate stringRedisTemplate;
11    stringRedisTemplate.delete(redisKey);

```

b、RedisTemplate

在此存入redis的类型为：**String:Object**

常用操作：

```

redisTemplate.opsForValue();    //操作字符串
redisTemplate.opsForHash();     //操作hash
redisTemplate.opsForList();     //操作list
redisTemplate.opsForSet();      //操作set
redisTemplate.opsForZSet();     //操作有序set

```

StringRedisTemplate与RedisTemplate区别点

- 1、两者的关系是StringRedisTemplate继承RedisTemplate。
 - 2、两者的数据是不共通的；也就是说StringRedisTemplate只能管理StringRedisTemplate里面的数据，RedisTemplate只能管理RedisTemplate中的数据。
 - 3、其实他们两者之间的区别主要在于他们使用的序列化类：
 - a、RedisTemplate使用的是JdkSerializationRedisSerializer 存入数据会将数据先序列化成字节数组然后在存入Redis数据库。
 - b、StringRedisTemplate使用的是StringRedisSerializer
 - 4、SDR默认采用的序列化策略有两种，一种是String的序列化策略，一种是JDK的序列化策略。
 - a、StringRedisTemplate默认采用的是String的序列化策略，保存的key和value都是采用此策略序列化保存的。
 - b、RedisTemplate默认采用的是JDK的序列化策略，保存的key和value都是采用此策略序列化保存的。
- （当然有时间还可以去看看源码进行分析）

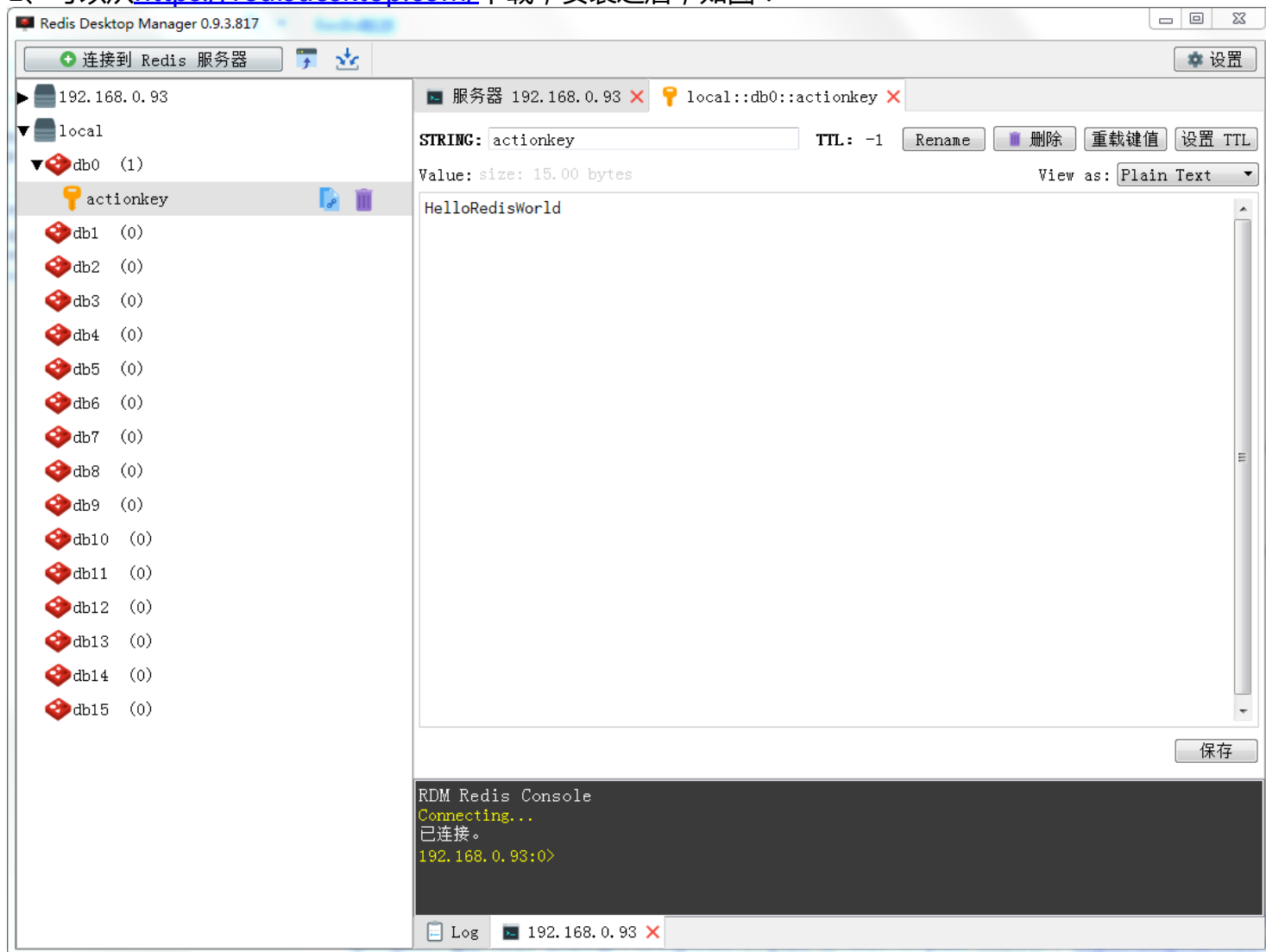
使用时注意事项：

- 1、当你的redis数据库里面本来存的是字符串数据或者你要存取的数据就是字符串类型数据的时候，那么你就使用StringRedisTemplate即可。

- 2、如果你的数据是复杂的对象类型，而取出的时候又不想做任何的数据转换，直接从Redis里面取出一个对象，那么使用RedisTemplate是更好的选择。
- 3、RedisTemplate使用时常见问题：
 - a、redisTemplate 中存取数据都是字节数组。
 - b、当redis中存入的数据是可读形式而非字节数组时，使用redisTemplate取值的时候会无法获取导出数据，获得的值为null。可以使用 StringRedisTemplate 试试。

Redis可视化工具：

- 1、可以从<https://redisdesktop.com/>下载，安装之后，如图：



当然各种工具有很多，个人觉得这个比较好用。推荐大家也可以下载使用。

参考资料：

Redis官网 <https://redis.io/>，中文官网 <http://www.redis.cn/>，资源官网 <https://github.com/MicrosoftArchive/redis/releases>

SpringBoot 整合 Redis 使用详解 (StringRedisTemplate 和 RedisTemplate 对比分析) <https://blog.csdn.net/Abysscarry/article/details/80557347>

如何使用RedisTemplate访问Redis数据结构 <https://www.jianshu.com/p/7bf5dc61ca06>

StringRedisTemplate操作redis数据 <https://www.cnblogs.com/slowcity/p/9002660.html>

Redis网络教程 <http://www.runoob.com/redis/redis-tutorial.html>

百度百科 <https://baike.baidu.com/item/Redis/6549233?fr=aladdin>

Redis 总结精讲 <https://blog.csdn.net/hjm4702192/article/details/80518856>

RedisDesktopManager工具官网 <https://redisdesktop.com/>