通过前一篇《[redis的简单使用](http://blog.csdn.net/u013256816/article/details/51120023)》的简单介绍，本篇主要阐述Jedis对redis的五大类型的操作：字符串、列表、散列、集合、有序集合。

JedisUtil

这里的测试用例采用junit4进行运行，准备代码如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | private static final String ipAddr = "10.10.195.112";      private static final int port = 6379;      private static Jedis jedis= null;        @BeforeClass      public static void init()      {          jedis = JedisUtil.getInstance().getJedis(ipAddr, port);      }        @AfterClass      public static void close()      {          JedisUtil.getInstance().closeJedis(jedis,ipAddr, port);      } |

其中JedisUtil是对jedis做的简单封装，代码如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91 | import org.apache.log4j.Logger;    import java.util.HashMap;  import java.util.Map;    import redis.clients.jedis.Jedis;  import redis.clients.jedis.JedisPool;  import redis.clients.jedis.JedisPoolConfig;    public class JedisUtil  {      private Logger logger = Logger.getLogger(this.getClass().getName());        private JedisUtil(){}        private static class RedisUtilHolder{          private static final JedisUtil instance = new JedisUtil();      }        public static JedisUtil getInstance(){          return RedisUtilHolder.instance;      }        private static Map<String,JedisPool> maps = new HashMap<String,JedisPool>();        private static JedisPool getPool(String ip, int port){          String key = ip+":"+port;          JedisPool pool = null;          if(!maps.containsKey(key))          {              JedisPoolConfig config = new JedisPoolConfig();              config.setMaxActive(RedisConfig.MAX\_ACTIVE);              config.setMaxIdle(RedisConfig.MAX\_IDLE);              config.setMaxWait(RedisConfig.MAX\_WAIT);              config.setTestOnBorrow(true);              config.setTestOnReturn(true);                pool = new JedisPool(config,ip,port,RedisConfig.TIMEOUT);              maps.put(key, pool);          }          else          {              pool = maps.get(key);          }          return pool;      }        public Jedis getJedis(String ip, int port)      {          Jedis jedis = null;          int count = 0;          do          {              try              {                  jedis = getPool(ip,port).getResource();              }              catch (Exception e)              {                  logger.error("get redis master1 failed!",e);                  getPool(ip,port).returnBrokenResource(jedis);              }          }          while(jedis == null && count<RedisConfig.RETRY\_NUM);          return jedis;      }        public void closeJedis(Jedis jedis, String ip, int port){          if(jedis != null)          {              getPool(ip,port).returnResource(jedis);          }      }  }    public class RedisConfig  {      //可用连接实例的最大数目，默认值为8；      //如果赋值为-1，则表示不限制；如果pool已经分配了maxActive个jedis实例，则此时pool的状态为exhausted(耗尽)。      public static int MAX\_ACTIVE = 1024;        //控制一个pool最多有多少个状态为idle(空闲的)的jedis实例，默认值也是8。      public static int MAX\_IDLE = 200;        //等待可用连接的最大时间，单位毫秒，默认值为-1，表示永不超时。如果超过等待时间，则直接抛出JedisConnectionException；      public static int MAX\_WAIT = 10000;        public static int TIMEOUT = 10000;        public static int RETRY\_NUM = 5;  } |

键操作

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | @Test public void testKey() throws InterruptedException     {         System.out.println("清空数据："+jedis.flushDB());         System.out.println("判断某个键是否存在："+jedis.exists("username"));         System.out.println("新增<'username','zzh'>的键值对："+jedis.set("username", "zzh"));         System.out.println(jedis.exists("name"));         System.out.println("新增<'password','password'>的键值对："+jedis.set("password", "password"));         System.out.print("系统中所有的键如下：");         Set<String> keys = jedis.keys("\*");         System.out.println(keys);         System.out.println("删除键password:"+jedis.del("password"));         System.out.println("判断键password是否存在："+jedis.exists("password"));         System.out.println("设置键username的过期时间为5s:"+jedis.expire("username", 5));         TimeUnit.SECONDS.sleep(2);         System.out.println("查看键username的剩余生存时间："+jedis.ttl("username"));         System.out.println("移除键username的生存时间："+jedis.persist("username"));         System.out.println("查看键username的剩余生存时间："+jedis.ttl("username"));         System.out.println("查看键username所存储的值的类型："+jedis.type("username"));     } |

输出结果：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | 清空数据：OK  判断某个键是否存在：false  新增<'username','zzh'>的键值对：OK  false  新增<'password','password'>的键值对：OK  系统中所有的键如下：[username, password]  删除键password:1  判断键password是否存在：false  设置键username的过期时间为5s:1  查看键username的剩余生存时间：3  移除键username的生存时间：1  查看键username的剩余生存时间：-1  查看键username所存储的值的类型：string |

字符串操作

在Redis里面，字符串可以存储三种类型的值：

* 字节串(byte string)
* 整数
* 浮点数

**字节串**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39 | @Test public void testString() throws InterruptedException    {        jedis.flushDB();        System.out.println("===========增加数据===========");        System.out.println(jedis.set("key1","value1"));        System.out.println(jedis.set("key2","value2"));        System.out.println(jedis.set("key3", "value3"));        System.out.println("删除键key2:"+jedis.del("key2"));        System.out.println("获取键key2:"+jedis.get("key2"));        System.out.println("修改key1:"+jedis.set("key1", "value1Changed"));        System.out.println("获取key1的值："+jedis.get("key1"));        System.out.println("在key3后面加入值："+jedis.append("key3", "End"));        System.out.println("key3的值："+jedis.get("key3"));        System.out.println("增加多个键值对："+jedis.mset("key01","value01","key02","value02","key03","value03"));        System.out.println("获取多个键值对："+jedis.mget("key01","key02","key03"));        System.out.println("获取多个键值对："+jedis.mget("key01","key02","key03","key04"));        System.out.println("删除多个键值对："+jedis.del(new String[]{"key01","key02"}));        System.out.println("获取多个键值对："+jedis.mget("key01","key02","key03"));          jedis.flushDB();        System.out.println("===========新增键值对防止覆盖原先值==============");        System.out.println(jedis.setnx("key1", "value1"));        System.out.println(jedis.setnx("key2", "value2"));        System.out.println(jedis.setnx("key2", "value2-new"));        System.out.println(jedis.get("key1"));        System.out.println(jedis.get("key2"));          System.out.println("===========新增键值对并设置有效时间=============");        System.out.println(jedis.setex("key3", 2, "value3"));        System.out.println(jedis.get("key3"));        TimeUnit.SECONDS.sleep(3);        System.out.println(jedis.get("key3"));          System.out.println("===========获取原值，更新为新值==========");//GETSET is an atomic set this value and return the old value command.        System.out.println(jedis.getSet("key2", "key2GetSet"));        System.out.println(jedis.get("key2"));          System.out.println("获得key2的值的字串："+jedis.getrange("key2", 2, 4));    } |

输出结果：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29 | ===========增加数据===========  OK  OK  OK  删除键key2:1  获取键key2:null  修改key1:OK  获取key1的值：value1Changed  在key3后面加入值：9  key3的值：value3End  增加多个键值对：OK  获取多个键值对：[value01, value02, value03]  获取多个键值对：[value01, value02, value03, null]  删除多个键值对：2  获取多个键值对：[null, null, value03]  ===========新增键值对防止覆盖原先值==============  1  1  0  value1  value2  ===========新增键值对并设置有效时间=============  OK  value3  null  ===========获取原值，更新为新值==========  value2  key2GetSet  获得key2的值的字串：y2G |

*memcached和redis同样有append的操作，但是memcached有prepend的操作，redis中并没有。*

**整数和浮点数**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | @Test public void testNumber()     {         jedis.flushDB();         jedis.set("key1", "1");         jedis.set("key2", "2");         jedis.set("key3", "2.3");         System.out.println("key1的值："+jedis.get("key1"));         System.out.println("key2的值："+jedis.get("key2"));         System.out.println("key1的值加1："+jedis.incr("key1"));         System.out.println("获取key1的值："+jedis.get("key1"));         System.out.println("key2的值减1："+jedis.decr("key2"));         System.out.println("获取key2的值："+jedis.get("key2"));         System.out.println("将key1的值加上整数5："+jedis.incrBy("key1", 5));         System.out.println("获取key1的值："+jedis.get("key1"));         System.out.println("将key2的值减去整数5："+jedis.decrBy("key2", 5));         System.out.println("获取key2的值："+jedis.get("key2"));     } |

输出结果：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | key1的值：1  key2的值：2  key1的值加1：2  获取key1的值：2  key2的值减1：1  获取key2的值：1  将key1的值加上整数5：7  获取key1的值：7  将key2的值减去整数5：-4  获取key2的值：-4 |

*在redis2.6或以上版本中有这个命令：incrbyfloat，即将键存储的值加上浮点数amount，jedis-2.1.0中不支持这一操作。*

列表

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33 | @Test public void testList()  {      jedis.flushDB();      System.out.println("===========添加一个list===========");      jedis.lpush("collections", "ArrayList", "Vector", "Stack", "HashMap", "WeakHashMap", "LinkedHashMap");      jedis.lpush("collections", "HashSet");      jedis.lpush("collections", "TreeSet");      jedis.lpush("collections", "TreeMap");      System.out.println("collections的内容："+jedis.lrange("collections", 0, -1));//-1代表倒数第一个元素，-2代表倒数第二个元素      System.out.println("collections区间0-3的元素："+jedis.lrange("collections",0,3));      System.out.println("===============================");      // 删除列表指定的值 ，第二个参数为删除的个数（有重复时），后add进去的值先被删，类似于出栈      System.out.println("删除指定元素个数："+jedis.lrem("collections", 2, "HashMap"));      System.out.println("collections的内容："+jedis.lrange("collections", 0, -1));      System.out.println("删除下表0-3区间之外的元素："+jedis.ltrim("collections", 0, 3));      System.out.println("collections的内容："+jedis.lrange("collections", 0, -1));      System.out.println("collections列表出栈（左端）："+jedis.lpop("collections"));      System.out.println("collections的内容："+jedis.lrange("collections", 0, -1));      System.out.println("collections添加元素，从列表右端，与lpush相对应："+jedis.rpush("collections", "EnumMap"));      System.out.println("collections的内容："+jedis.lrange("collections", 0, -1));      System.out.println("collections列表出栈（右端）："+jedis.rpop("collections"));      System.out.println("collections的内容："+jedis.lrange("collections", 0, -1));      System.out.println("修改collections指定下标1的内容："+jedis.lset("collections", 1, "LinkedArrayList"));      System.out.println("collections的内容："+jedis.lrange("collections", 0, -1));      System.out.println("===============================");      System.out.println("collections的长度："+jedis.llen("collections"));      System.out.println("获取collections下标为2的元素："+jedis.lindex("collections", 2));      System.out.println("===============================");      jedis.lpush("sortedList", "3","6","2","0","7","4");      System.out.println("sortedList排序前："+jedis.lrange("sortedList", 0, -1));      System.out.println(jedis.sort("sortedList"));      System.out.println("sortedList排序后："+jedis.lrange("sortedList", 0, -1));  } |

输出结果：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | ===========添加一个list===========  collections的内容：[TreeMap, TreeSet, HashSet, LinkedHashMap, WeakHashMap, HashMap, Stack, Vector, ArrayList]  collections区间0-3的元素：[TreeMap, TreeSet, HashSet, LinkedHashMap]  ===============================  删除指定元素个数：1  collections的内容：[TreeMap, TreeSet, HashSet, LinkedHashMap, WeakHashMap, Stack, Vector, ArrayList]  删除下表0-3区间之外的元素：OK  collections的内容：[TreeMap, TreeSet, HashSet, LinkedHashMap]  collections列表出栈（左端）：TreeMap  collections的内容：[TreeSet, HashSet, LinkedHashMap]  collections添加元素，从列表右端，与lpush相对应：4  collections的内容：[TreeSet, HashSet, LinkedHashMap, EnumMap]  collections列表出栈（右端）：EnumMap  collections的内容：[TreeSet, HashSet, LinkedHashMap]  修改collections指定下标1的内容：OK  collections的内容：[TreeSet, LinkedArrayList, LinkedHashMap]  ===============================  collections的长度：3  获取collections下标为2的元素：LinkedHashMap  ===============================  sortedList排序前：[4, 7, 0, 2, 6, 3]  [0, 2, 3, 4, 6, 7]  sortedList排序后：[4, 7, 0, 2, 6, 3] |

*Redis中还有阻塞式的列表弹出命令以及在列表之间移动元素的命令：blpop, brpop, rpoplpush, brpoplpush等。*

集合（Set）

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33 | @Test public void testSet()    {        jedis.flushDB();        System.out.println("============向集合中添加元素============");        System.out.println(jedis.sadd("eleSet", "e1","e2","e4","e3","e0","e8","e7","e5"));        System.out.println(jedis.sadd("eleSet", "e6"));        System.out.println(jedis.sadd("eleSet", "e6"));        System.out.println("eleSet的所有元素为："+jedis.smembers("eleSet"));        System.out.println("删除一个元素e0："+jedis.srem("eleSet", "e0"));        System.out.println("eleSet的所有元素为："+jedis.smembers("eleSet"));        System.out.println("删除两个元素e7和e6："+jedis.srem("eleSet", "e7","e6"));        System.out.println("eleSet的所有元素为："+jedis.smembers("eleSet"));        System.out.println("随机的移除集合中的一个元素："+jedis.spop("eleSet"));        System.out.println("随机的移除集合中的一个元素："+jedis.spop("eleSet"));        System.out.println("eleSet的所有元素为："+jedis.smembers("eleSet"));        System.out.println("eleSet中包含元素的个数："+jedis.scard("eleSet"));        System.out.println("e3是否在eleSet中："+jedis.sismember("eleSet", "e3"));        System.out.println("e1是否在eleSet中："+jedis.sismember("eleSet", "e1"));        System.out.println("e1是否在eleSet中："+jedis.sismember("eleSet", "e5"));        System.out.println("=================================");        System.out.println(jedis.sadd("eleSet1", "e1","e2","e4","e3","e0","e8","e7","e5"));        System.out.println(jedis.sadd("eleSet2", "e1","e2","e4","e3","e0","e8"));        System.out.println("将eleSet1中删除e1并存入eleSet3中："+jedis.smove("eleSet1", "eleSet3", "e1"));        System.out.println("将eleSet1中删除e2并存入eleSet3中："+jedis.smove("eleSet1", "eleSet3", "e2"));        System.out.println("eleSet1中的元素："+jedis.smembers("eleSet1"));        System.out.println("eleSet3中的元素："+jedis.smembers("eleSet3"));        System.out.println("============集合运算=================");        System.out.println("eleSet1中的元素："+jedis.smembers("eleSet1"));        System.out.println("eleSet2中的元素："+jedis.smembers("eleSet2"));        System.out.println("eleSet1和eleSet2的交集:"+jedis.sinter("eleSet1","eleSet2"));        System.out.println("eleSet1和eleSet2的并集:"+jedis.sunion("eleSet1","eleSet2"));        System.out.println("eleSet1和eleSet2的差集:"+jedis.sdiff("eleSet1","eleSet2"));//eleSet1中有，eleSet2中没有    } |

输出结果：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29 | ============向集合中添加元素============  8  1  0  eleSet的所有元素为：[e3, e4, e1, e2, e0, e8, e7, e6, e5]  删除一个元素e0：1  eleSet的所有元素为：[e3, e4, e1, e2, e8, e7, e6, e5]  删除两个元素e7和e6：2  eleSet的所有元素为：[e3, e4, e1, e2, e8, e5]  随机的移除集合中的一个元素：e5  随机的移除集合中的一个元素：e2  eleSet的所有元素为：[e3, e4, e1, e8]  eleSet中包含元素的个数：4  e3是否在eleSet中：true  e1是否在eleSet中：true  e1是否在eleSet中：false  =================================  8  6  将eleSet1中删除e1并存入eleSet3中：1  将eleSet1中删除e2并存入eleSet3中：1  eleSet1中的元素：[e3, e4, e0, e8, e7, e5]  eleSet3中的元素：[e1, e2]  ============集合运算=================  eleSet1中的元素：[e3, e4, e0, e8, e7, e5]  eleSet2中的元素：[e3, e4, e1, e2, e0, e8]  eleSet1和eleSet2的交集:[e3, e4, e0, e8]  eleSet1和eleSet2的并集:[e3, e4, e1, e2, e0, e8, e7, e5]  eleSet1和eleSet2的差集:[e7, e5] |

*关于Set还有一些其他命令：srandmember, sdiffstore, sinterstore, sunionstore等。*

散列

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | @Test public void testHash()      {          jedis.flushDB();          Map<String,String> map = new HashMap<>();          map.put("key1","value1");          map.put("key2","value2");          map.put("key3","value3");          map.put("key4","value4");          jedis.hmset("hash",map);          jedis.hset("hash", "key5", "value5");          System.out.println("散列hash的所有键值对为："+jedis.hgetAll("hash"));//return Map<String,String>          System.out.println("散列hash的所有键为："+jedis.hkeys("hash"));//return Set<String>          System.out.println("散列hash的所有值为："+jedis.hvals("hash"));//return List<String>          System.out.println("将key6保存的值加上一个整数，如果key6不存在则添加key6："+jedis.hincrBy("hash", "key6", 6));          System.out.println("散列hash的所有键值对为："+jedis.hgetAll("hash"));          System.out.println("将key6保存的值加上一个整数，如果key6不存在则添加key6："+jedis.hincrBy("hash", "key6", 3));          System.out.println("散列hash的所有键值对为："+jedis.hgetAll("hash"));          System.out.println("删除一个或者多个键值对："+jedis.hdel("hash", "key2"));          System.out.println("散列hash的所有键值对为："+jedis.hgetAll("hash"));          System.out.println("散列hash中键值对的个数："+jedis.hlen("hash"));          System.out.println("判断hash中是否存在key2："+jedis.hexists("hash","key2"));          System.out.println("判断hash中是否存在key3："+jedis.hexists("hash","key3"));          System.out.println("获取hash中的值："+jedis.hmget("hash","key3"));          System.out.println("获取hash中的值："+jedis.hmget("hash","key3","key4"));      } |

输出结果：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | 散列hash的所有键值对为：{key4=value4, key3=value3, key5=value5, key2=value2, key1=value1}  散列hash的所有键为：[key4, key3, key5, key2, key1]  散列hash的所有值为：[value4, value3, value1, value2, value5]  将key6保存的值加上一个整数，如果key6不存在则添加key6：6  散列hash的所有键值对为：{key4=value4, key3=value3, key6=6, key5=value5, key2=value2, key1=value1}  将key6保存的值加上一个整数，如果key6不存在则添加key6：9  散列hash的所有键值对为：{key4=value4, key3=value3, key6=9, key5=value5, key2=value2, key1=value1}  删除一个或者多个键值对：1  散列hash的所有键值对为：{key4=value4, key3=value3, key6=9, key5=value5, key1=value1}  散列hash中键值对的个数：5  判断hash中是否存在key2：false  判断hash中是否存在key3：true  获取hash中的值：[value3]  获取hash中的值：[value3, value4] |

有序集合

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | @Test public void testSortedSet()      {          jedis.flushDB();          Map<Double,String> map = new HashMap<>();          map.put(1.2,"key2");          map.put(4.0, "key3");          map.put(5.0,"key4");          map.put(0.2,"key5");          System.out.println(jedis.zadd("zset", 3,"key1"));          System.out.println(jedis.zadd("zset",map));          System.out.println("zset中的所有元素："+jedis.zrange("zset", 0, -1));          System.out.println("zset中的所有元素："+jedis.zrangeWithScores("zset", 0, -1));          System.out.println("zset中的所有元素："+jedis.zrangeByScore("zset", 0,100));          System.out.println("zset中的所有元素："+jedis.zrangeByScoreWithScores("zset", 0,100));          System.out.println("zset中key2的分值："+jedis.zscore("zset", "key2"));          System.out.println("zset中key2的排名："+jedis.zrank("zset", "key2"));          System.out.println("删除zset中的元素key3："+jedis.zrem("zset", "key3"));          System.out.println("zset中的所有元素："+jedis.zrange("zset", 0, -1));          System.out.println("zset中元素的个数："+jedis.zcard("zset"));          System.out.println("zset中分值在1-4之间的元素的个数："+jedis.zcount("zset", 1, 4));          System.out.println("key2的分值加上5："+jedis.zincrby("zset", 5, "key2"));          System.out.println("key3的分值加上4："+jedis.zincrby("zset", 4, "key3"));          System.out.println("zset中的所有元素："+jedis.zrange("zset", 0, -1));      } |

输出结果：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | 1  4  zset中的所有元素：[key5, key2, key1, key3, key4]  zset中的所有元素：[[[107, 101, 121, 53],0.2], [[107, 101, 121, 50],1.2], [[107, 101, 121, 49],3.0], [[107, 101, 121, 51],4.0], [[107, 101, 121, 52],5.0]]  zset中的所有元素：[key5, key2, key1, key3, key4]  zset中的所有元素：[[[107, 101, 121, 53],0.2], [[107, 101, 121, 50],1.2], [[107, 101, 121, 49],3.0], [[107, 101, 121, 51],4.0], [[107, 101, 121, 52],5.0]]  zset中key2的分值：1.2  zset中key2的排名：1  删除zset中的元素key3：1  zset中的所有元素：[key5, key2, key1, key4]  zset中元素的个数：4  zset中分值在1-4之间的元素的个数：2  key2的分值加上5：6.2  key3的分值加上4：4.0  zset中的所有元素：[key5, key1, key3, key4, key2] |

*有序集合还有诸如zinterstore, zunionstore, zremrangebyscore, zremrangebyrank, zrevrank, zrevrange, zrangebyscore等命令。*

排序sort

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | @Test public void testSort()    {        jedis.flushDB();        jedis.lpush("collections", "ArrayList", "Vector", "Stack", "HashMap", "WeakHashMap", "LinkedHashMap");        System.out.println("collections的内容："+jedis.lrange("collections", 0, -1));        SortingParams sortingParameters = new SortingParams();        System.out.println(jedis.sort("collections",sortingParameters.alpha()));        System.out.println("===============================");        jedis.lpush("sortedList", "3","6","2","0","7","4");        System.out.println("sortedList排序前："+jedis.lrange("sortedList", 0, -1));        System.out.println("升序："+jedis.sort("sortedList", sortingParameters.asc()));        System.out.println("升序："+jedis.sort("sortedList", sortingParameters.desc()));        System.out.println("===============================");        jedis.lpush("userlist", "33");        jedis.lpush("userlist", "22");        jedis.lpush("userlist", "55");        jedis.lpush("userlist", "11");        jedis.hset("user:66", "name", "66");        jedis.hset("user:55", "name", "55");        jedis.hset("user:33", "name", "33");        jedis.hset("user:22", "name", "79");        jedis.hset("user:11", "name", "24");        jedis.hset("user:11", "add", "beijing");        jedis.hset("user:22", "add", "shanghai");        jedis.hset("user:33", "add", "guangzhou");        jedis.hset("user:55", "add", "chongqing");        jedis.hset("user:66", "add", "xi'an");        sortingParameters = new SortingParams();        sortingParameters.get("user:\*->name");        sortingParameters.get("user:\*->add");        System.out.println(jedis.sort("userlist",sortingParameters));    } |

输出结果：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | collections的内容：[LinkedHashMap, WeakHashMap, HashMap, Stack, Vector, ArrayList]  [ArrayList, HashMap, LinkedHashMap, Stack, Vector, WeakHashMap]  ===============================  sortedList排序前：[4, 7, 0, 2, 6, 3]  升序：[0, 2, 3, 4, 6, 7]  升序：[7, 6, 4, 3, 2, 0]  ===============================  [24, beijing, 79, shanghai, 33, guangzhou, 55, chongqing] |