## Redis命令

### String类型

|  |  |  |
| --- | --- | --- |
| 命令 | 说明 | **案例** |
| set | 添加key-value | set username admin |
| get | 根据key获取数据 | get username |
| strlen | 根据key获取值的长度 | strlen key |
| exists | 判断key是否存在 | exists name  返回1存在 0不存在 |
| del | 删除redis中的key | del key |
| Keys | 用于查询符合条件的key | keys \* 查询redis中全部的key  keys n?me 使用占位符获取数据  keys nam\* 获取nam开头的数据 |
| mset | 赋值多个key-value | mset key1 value1 key2 value2 key3 value3 |
| mget | 获取多个key的值 | mget key1 key2 |
| append | 对某个key的值进行追加 | append key value |
| type | 检查某个key的类型 | type key |
| select | 切换redis数据库 | select 0-15 redis中共有16个数据库 |
| flushdb | 清空单个数据库 | flushdb |
| flushall | 清空全部数据库 | flushall |
| incr | 自动加1 | incr key |
| decr | 自动减1 | decr key |
| incrby | 指定数值添加 | incrby 10 |
| decrby | 指定数值减 | decrby 10 |
| expire | 指定key的生效时间 单位秒 | expire key 20  key20秒后失效 |
| pexpire | 指定key的失效时间 单位毫秒 | pexpire key 2000  key 2000毫秒后失效 |
| ttl | 检查key的剩余存活时间 | ttl key -2数据不存在 -1该数据永不超时 |
| persist | 撤销key的失效时间 | persist key |

### Hash类型

说明:可以用散列类型保存对象和属性值

例子:User对象{id:2,name:小明,age:19}

|  |  |  |
| --- | --- | --- |
| 命令 | 说明 | **案例** |
| hset | 为对象添加数据 | hset key field value |
| hget | 获取对象的属性值 | hget key field |
| hexists | 判断对象的属性是否存在 | HEXISTS key field  1表示存在 0表示不存在 |
| hdel | 删除hash中的属性 | hdel user field [field ...] |
| hgetall | 获取hash全部元素和值 | HGETALL key |
| hkyes | 获取hash中的所有字段 | HKEYS key |
| hlen | 获取hash中所有属性的数量 | hlen key |
| hmget | 获取hash里面指定字段的值 | hmget key field [field ...] |
| hmset | 为hash的多个字段设定值 | hmset key field value [field value ...] |
| hsetnx | 设置hash的一个字段,只有当这个字段不存在时有效 | HSETNX key field value |
| hstrlen | 获取hash中指定key的值的长度 | HSTRLEN key field |
| hvals | 获取hash的所有值 | HVALS user |

### List类型

说明:Redis中的List集合是双端循环列表,分别可以从左右两个方向插入数据.

List集合可以当做队列使用,也可以当做栈使用

队列:存入数据的方向和获取数据的方向相反

栈:存入数据的方向和获取数据的方向相同

|  |  |  |
| --- | --- | --- |
| 命令 | 说明 | **案例** |
| lpush | 从队列的左边入队一个或多个元素 | LPUSH key value [value ...] |
| rpush | 从队列的右边入队一个或多个元素 | RPUSH key value [value ...] |
| lpop | 从队列的左端出队一个元素 | LPOP key |
| rpop | 从队列的右端出队一个元素 | RPOP key |
| lpushx | 当队列存在时从队列的左侧入队一个元素 | LPUSHX key value |
| rpushx | 当队列存在时从队列的右侧入队一个元素 | RPUSHx key value |
| lrange | 从列表中获取指定返回的元素 | LRANGE key start stop  Lrange key 0 -1 获取全部队列的数据 |
| lrem | 从存于 key 的列表里移除前 count 次出现的值为 value 的元素。 这个 count 参数通过下面几种方式影响这个操作：   * count > 0: 从头往尾移除值为 value 的元素。 * count < 0: 从尾往头移除值为 value 的元素。 * count = 0: 移除所有值为 value 的元素。 | LREM list -2 “hello” 会从存于 list 的列表里移除最后两个出现的 “hello”。  需要注意的是，如果list里没有存在key就会被当作空list处理，所以当 key 不存在的时候，这个命令会返回 0。 |
| Lset | 设置 index 位置的list元素的值为 value | LSET key index value |

### Redis事务命令

说明:redis中操作可以添加事务的支持.一项任务可以由多个redis命令完成,如果有一个命令失败导致入库失败时.需要实现事务回滚.

|  |  |  |
| --- | --- | --- |
| 命令 | 说明 | **案例** |
| multi | 标记一个事务开始 | 127.0.0.1:6379> MULTI  OK |
| exec | 执行所有multi之后发的命令 | 127.0.0.1:6379> EXEC  OK |
| discard | 丢弃所有multi之后发的命令 |  |

# Redis高级应用

## Redis入门案例

### 添加jar包文件

说明:在JT-PARENT项目中添加jar包文件

<!-- jedis -->

<dependency>

<groupId>redis.clients</groupId>

<artifactId>jedis</artifactId>

<version>${jedis.version}</version>

</dependency>

<!--添加spring-datajar包 -->

<dependency>

<groupId>org.springframework.data</groupId>

<artifactId>spring-data-redis</artifactId>

<version>1.4.1.RELEASE</version>

</dependency>

### 入门案例-String

/\*\*

\* 连接单台redis

\* 参数介绍:

\* redisIP地址.

\* redis:6379

\*/

@Test

**public** **void** test01(){

Jedis jedis = **new** Jedis("192.168.126.166",6379);

jedis.set("redis", "redis入门案例");

System.***out***.println

("获取redis中的数据:"+jedis.get("redis"));

//为数据设定超时时间 单位秒

jedis.setex("1804", 100, "1804班");

}

### 入门案例-hash

@Test

**public** **void** test01(){

Jedis jedis = **new** Jedis("192.168.126.148", 6379);

jedis.hset("user", "id", "1");

jedis.hset("user", "name", "tomcat");

jedis.hset("user", "age", "18");

System.***out***.println("操作完成!!!"+jedis.hget("user", "id"));

Map<String,String> map = jedis.hgetAll("user");

System.***out***.println(map);

}

结果展现:

操作完成!!!1

{name=tomcat, age=18, id=1}

### 入门案例-List

@Test

**public** **void** test02(){

Jedis jedis = **new** Jedis("192.168.126.148", 6379);

Long number = jedis.lpush("list", "a","b","c","d","e");

System.***out***.println("获取数据"+number);

List<String> list= jedis.lrange("list", 0, -1);

System.***out***.println("获取参数:"+list);

}

结果展现:

获取数据5

获取参数:[e, d, c, b, a]