Redis API使用文档

Author:jannal

# EnableRedisCache注解基本讲解

|  |  |  |
| --- | --- | --- |
| 属性 | 默认值 | 讲解 |
| isOpen | true | 是否开启缓存，默认是true |
| key | 无 | 缓存的key,可以动态传值 |
| value | 无 | 查询后需要转换的Class |
| exceptionStrategy |  | Redis出现异常后的策略,捕获和抛出，默认  ExceptionStrategy.TRY\_CATCH |
| retryTimes | 重试次数 | 当exceptionStrategy为ExceptionStrategy.TRY\_CATCH时才有效。默认情况下进行重试1次(如果事务失败或者redis操作出现其他异常) |
| timeUnit |  | 超时时间与是否配置超时对应，默认单位是秒 |
| timeOut | 0L | 超时时间0L表示不超时，大于0表示超时 |
| cacheType | 缓存类型 | 默认CacheType.REDIS |
| retryTimes |  | 重试次数，默认情况下不进行重试操作(如果事务失败或者redis操作出现其他异常) |
| operationStrategy | KeyStrategy.QUERY | 操作策略 |
| dataType | DataType.STRING | 字符串数据类型 |

## **Key**

缓存的key,可以动态传值(SPEL表达式)

举例

1、

/\*\*

\* 增加或者修改，增加或者修改完毕，必须返回需要缓存的对象

\*/

@EnableRedisCache(key="'member:'+#member.id",value=com.chtwm.base.tools.cache.bean.Member.**class**)

**public** Member saveOrUpdateMember(Member member){

logger.info("开始存入数据库");

Member memberExist = dataMap.get(member.getId()+"");

**if**(memberExist==**null**){

dataMap.put(member.getId()+"",member);

}**else**{

dataMap.put(member.getId()+"", member);

}

**return** member;

}

key="'member:'+#member.id"

参数中指定 member对象,假设member的id是1，则最终的key=member:1

Key最外层需要**双引号**包含，内部字符串常量需要使用**单引号**，与变量拼接使用**加号(+)**,动态传值使用 **#对象.属性**

**对象是传入的参数**

**超时的key，默认会增加前缀TO：**

2、

/\*\*

\* 通过MemberId删除

\*/

@EnableRedisCache(key="'member:'+#memberId",operationStrategy=KeyStrategy.***DELETE***)

@Override

**public** **void** delete(Long memberId) {

dataMap.remove(memberId+"");

}

## **value**

value:主要用于查询后的数据转换为实体对象

/\*\*

\* 通过Id查询Member

\*/

@EnableRedisCache(key="'member:'+#memberId",

value=com.chtwm.base.tools.cache.bean.Member.**class**)

**public** Member findById(Long memberId){

logger.debug("通过memberId开始查询数据库.....");

Member memberExist = dataMap.get(memberId+"");

**return** memberExist;

}

以上代码表示，假如memberId=1，则查询member:1 后的json数据转换为Member类的对象。

## exceptionStrategy与retryTimes

1、redis发生异常后(可能是事务执行失败，也可能是其他异常)，是否捕获还是抛出。需要根据自己的业务进行选择，默认是捕获(ExceptionStrategy.***TRY\_CATCH***)。抛出异常是运行异常，会导致数据库会滚，所以在存入的时候要注意redis失败是否需要回滚数据库

2、retryTimes重试次数，只有exceptionStrategy为***TRY\_CATCH***时才有效果,默认是0表示不重试。

## timeUnit 与timeOut

1、timeOut默认是0L表示key不超时，此时配置TimeOut没有意义

2、如果timeOut大于0,配置TimeUnit才有意义。TimeUnit默认是秒。

## type

/\*\*

\* redis类型

\* **@author** jannal

\*/

**enum** RedisType{

***STRING***,

***HASHSET***,

***LIST***,

***SET***,

***SORTEDSET***;

}

代表redis的五种类型，默认是STRING类型

## operationStrategy

**enum** KeyStrategy {

***QUERY***,

***SAVE\_OR\_UPDATE***,//保存或更新

***DELETE***//删除

}

操作策略，默认是查询