爱旅行项目—Redis技术分析及运用

爱旅行项目的认证模块（Auth）中，基于系统的安全性考虑，需要设置Token并通过Redis进行Token有效期的维护管理。下面介绍Redis技术以及在本项目中的具体运用。

## 1.1 Redis简介

目前互联网基本使用两种方式来进行数据的存储：关系型数据库和NoSql数据库。

1. 关系数据库：主要用于存储格式化的数据结构，比如MySQL、Oracle、SqlServer等都属于关系型数据库。
2. NoSql（泛指：非关系型数据库）数据库，其存储的数据都是按照键值对（Key-Value）的结构进行存储。主要服务于特定背景的专用数据库，向外提供特定的API，而不是提供通用的SQL接口，所以效率更加高效，简单理解就是NOSQL可以不写任何SQL语句就能实现数据的存储与查询的数据库。

Redis是一个开源K-V的数据库，属于NoSql数据库，并且与Memcached一样，为了保证效率，数据都是缓存在内存中，并基于内存操作，性能较高。它所支持存储的value类型相对更多，包括string、list、set、zset和hash，在内存中设计了各种数据类型，让业务能够高速原子的访问这些数据结构，并且不需要关心持久存储的问题，从架构上解决了关系型数据库存储需要走一些弯路的问题。所以作为缓存应用，与其类似的[Memcached](http://baike.baidu.com/view/794242.htm)，EHCache,OSCache等缓存器相比，Redis的出现，很大程度补偿了[Memcached](http://baike.baidu.com/view/794242.htm)这类key/value存储的不足，在高并发查询时可以对关系数据库起到很好的性能补充作用。此外与Memcached的最大区别在于：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，故：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。

## 1.2 Redis安装与配置

### 1.2.1安装环境

1. 服务器：CentOS 6.8 64位
2. redis\_version:3.2.8

### 1.2.2安装步骤

1. 创建安装路径：/usr/local/redis
2. 下载地址：<http://redis.googlecode.com/files/redis-3.2.8.tar.gz>
3. 解压安装：
   * tar -zxvf redis-3.2.8.tar.gz
   * make PREFIX=/usr/local/redis install
   * mkdir /usr/local/redis/etc

注：可能会出现的错误提示如下：

>>提示1：  
make[3]: gcc：命令未找到  
>>解决  
yum -y install gcc-c++  
>>提示2：  
在包含自 adlist.c：34 的文件中:  
zmalloc.h:50:31: 错误：jemalloc/jemalloc.h：没有那个文件或目录  
zmalloc.h:55:2: 错误：#error "Newer version of jemalloc required"  
>>解决  
make的时候加上 MALLOC=libc 参数  
make PREFIX=/usr/local/redis MALLOC=libc install

1. 配置：
   * 设置内存分配策略（可选，根据服务器的实际情况进行设置，可选值：0、1、2 ） /proc/sys/vm/overcommit\_memory

echo 1 > /proc/sys/vm/overcommit\_memory

0：表示内核将检查是否有足够的可用内存供应用进程使用；如果有足够的可用内存，内存申请允许；否则，内存申请失败，并把错误返回给应用进程。

1：表示内核允许分配所有的物理内存，而不管当前的内存状态如何。

2：表示内核允许分配超过所有物理内存和交换空间总和的内存

值得注意的一点是，redis在dump数据的时候，会fork出一个子进程，理论上child进程所占用的内存和parent是一样的，比如parent 占用的内存为8G，此时也要同样分配8G的内存给child,如果内存无法负担，往往会造成redis服务器的down机或者IO负载过高，效率下降。所以较为优化的内存分配策略应该设置为 1（表示内核允许分配所有的物理内存，而不管当前的内存状态如何）。

* + 开启Redis端口（6379），修改防火墙配置文件
    - vi /etc/sysconfig/iptables
    - -A INPUT -m state --state NEW -m tcp -p tcp --dport 6379 -j ACCEPT
    - 重新加载规则：service iptables restart
  + 指定Redis的log文件所在目录
    - 设置Redis配置文件redis.conf中参数logfile
  + 指定RedisDB的数据文件所在目录
    - 设置Redis配置文件redis.conf中参数dir
  + 设置Redis密码

Redis默认无密码，可设置密码，设置redis配置文件redis.conf中的参数requirepass的值即可，修改完毕后重启Redis服务。

1. 启动/停止Redis：
   * 启动：redis-server /usr/local/redis/etc/redis.conf
   * 停止：kill -9 id

### 1.2.3 Redis客户端

除了使用Shell命令行操作Redis，在项目开发期间还可以安装Redis的图形化桌面客户端（RedisDesktopManager），如下图1所示：

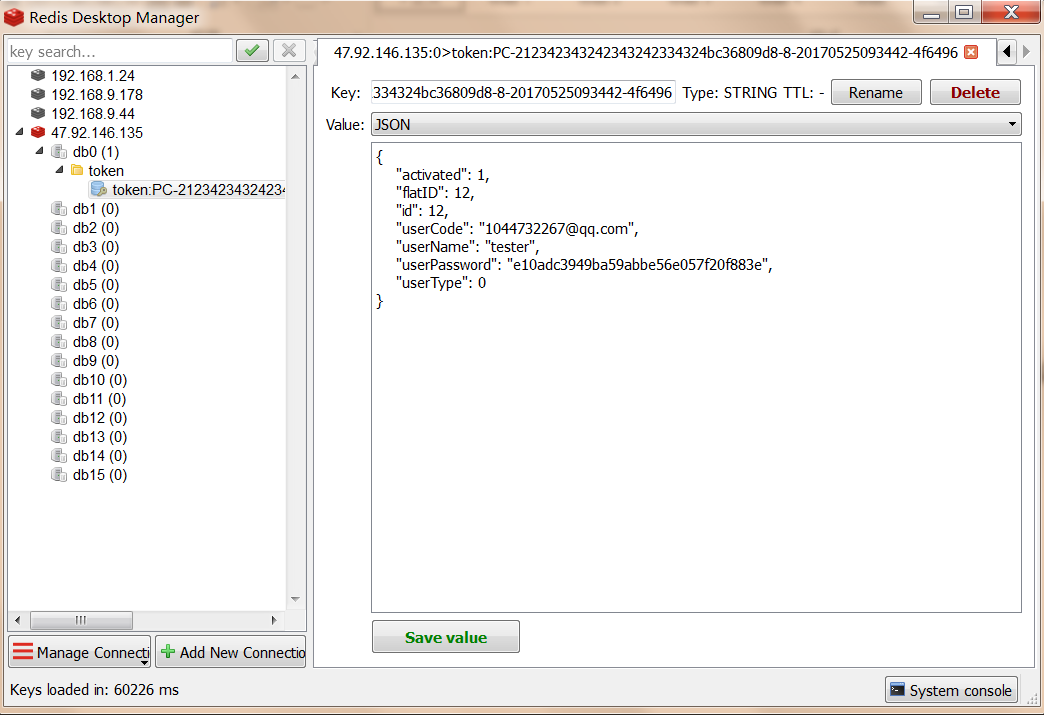


图1：RedisDesktopManager

该客户端操作方便，图形化界面更加清晰明了，非常适用于开发阶段，关于其安装过程较为简单，不再赘述。

## 1.3 Redis常用命令

下面介绍操作Redis的常用命令：

1. redis-cli：Shell命令行下启动Redis客户端工具
2. set（set key value）：

之前介绍过Redis 是以key-value的格式来存储数据的，set命令就是设置key对应的value值（string类型），设置成功，返回1；失败，返回0。

1. get（get key）：通过key值获取其对应的value值，若不存在，则返回nil。
2. exists（exists key）：判断指定的key是否存在，存在返回1，不存在返回0。
3. del(key)：删除一个指定key。
4. quit：关闭连接。
5. info：查看Redis的相关信息。
6. flushdb：清空当前库。
7. flushall：清空所有数据库。

## 1.4 Redis项目中的运用

### 1.4.1 模块实现

Redis在爱旅行项目中主要负责Token的维护管理，针对不同的客户端的Token（PC、移动），维护不同的Token有效期（根据需求设计PC端的Token有效时间为2个小时，移动端的Token永不失效，只有当修改密码后，在Redis中更新该Token中用户修改的新密码即可）。那么用户在系统平台中的所有操作，若需要进行用户身份的验证，只需去Redis中跟key查询相对应的value值即可，便可获取到当前用户的相关信息。

### 1.4.2 Redis的设计思路

Redis的具体使用很简单，只需要在spring配置文件中配置三个对象即可，分别为：jedisPoolConfig、jedisPool、redisAPI，如下图2所示：

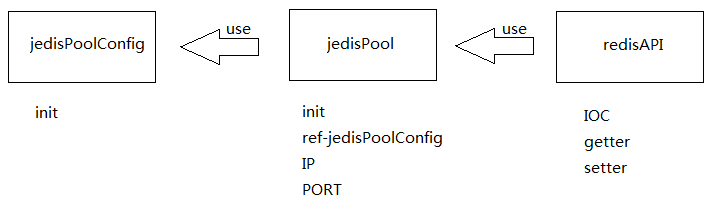
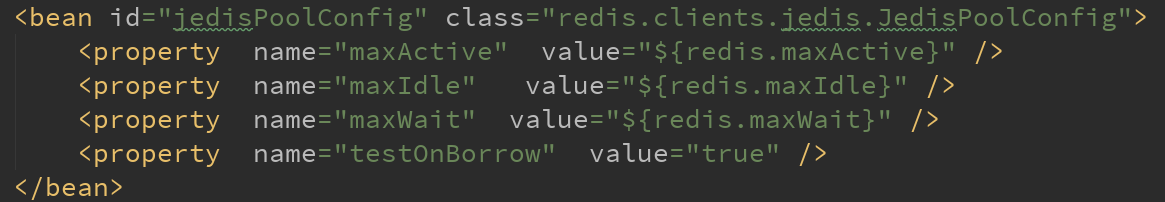


图2

首先初始化Redis连接池配置对象（jedisPoolConfig），然后再将该对象通过IoC注入到连接池对象（jedisPool）中，当连接池对象初始化时引用连接池配置对象（jedisPoolConfig），同时通过配置的IP、端口等进行连接的实例化。最后由自定义的redisAPI来使用jedisPool对象。redisAPI是基于jedisPool编写的一个数据访问对象，相当于系统中的dao层，比如说对数据进行一些set、get等操作。

### 1.4.3 Redis的相关配置

1. JedisPoolConfig：Redis连接池配置

首先进行spring IoC配置，在spring容器中注册一个ID为jedisPoolConfig的bean，使用的类是redis.clients.jedis.JedisPoolConfig。

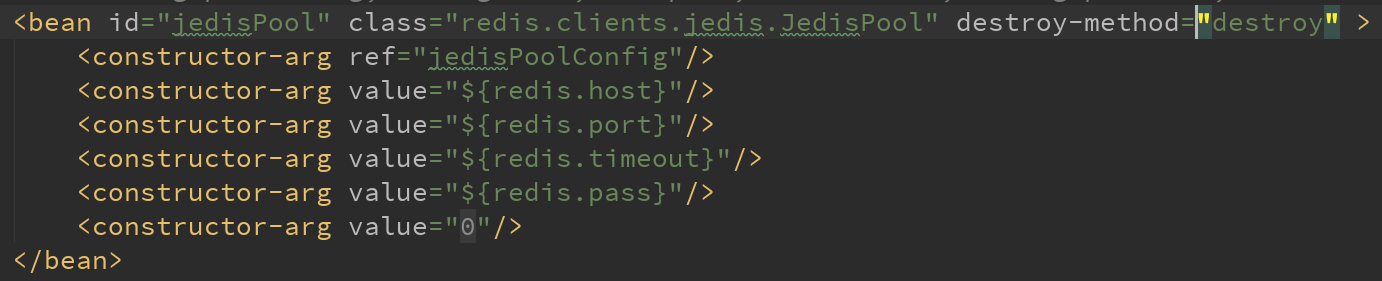
* + maxActive：最大活动数。

若设置value为-1，则表示不限制，若pool已经分配了n（设置的value）个的jedis实例，则此时pool的状态为exhausted(耗尽)。

* + maxIdle：最大空闲数。
  + maxWait：表示当borrow(引入/借调)一个jedis实例时，最大的等待时间，如果超过等待时间，则直接抛出JedisConnectionException。
  + testOnBorrow：在borrow一个jedis实例时，是否提前进行validate操作；若为true，则得到的jedis实例均是可用的。

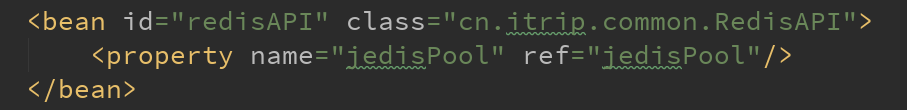
1. jedisPool

在spring容器中注册一个ID为jedisPool的bean，使用的类是redis.clients.jedis.JedisPool，再将配置好的redis的连接池配置对象（jedisPoolConfig）通过IoC注入到连接池对象中，连接池对象初始化时引用上面的连接池配置对象，同时通过配置的IP和端口等进行连接的实例化。



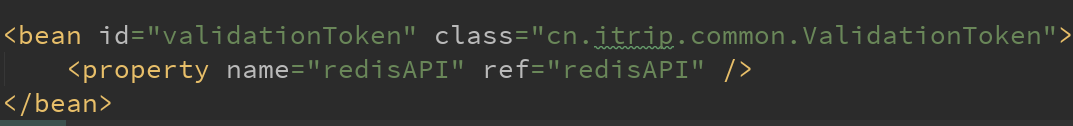
1. redisAPI

在spring容器中注册一个ID为redisAPI的bean，这个bean使用的类是org.slsale.common.RedisAPI，在实例化这个bean的时候要通过jedisPool进行初始化（相当于赋值）。



1. validationToken

该工具类主要是为了根据传入的key（Token）去Redis中获取相应用户信息。



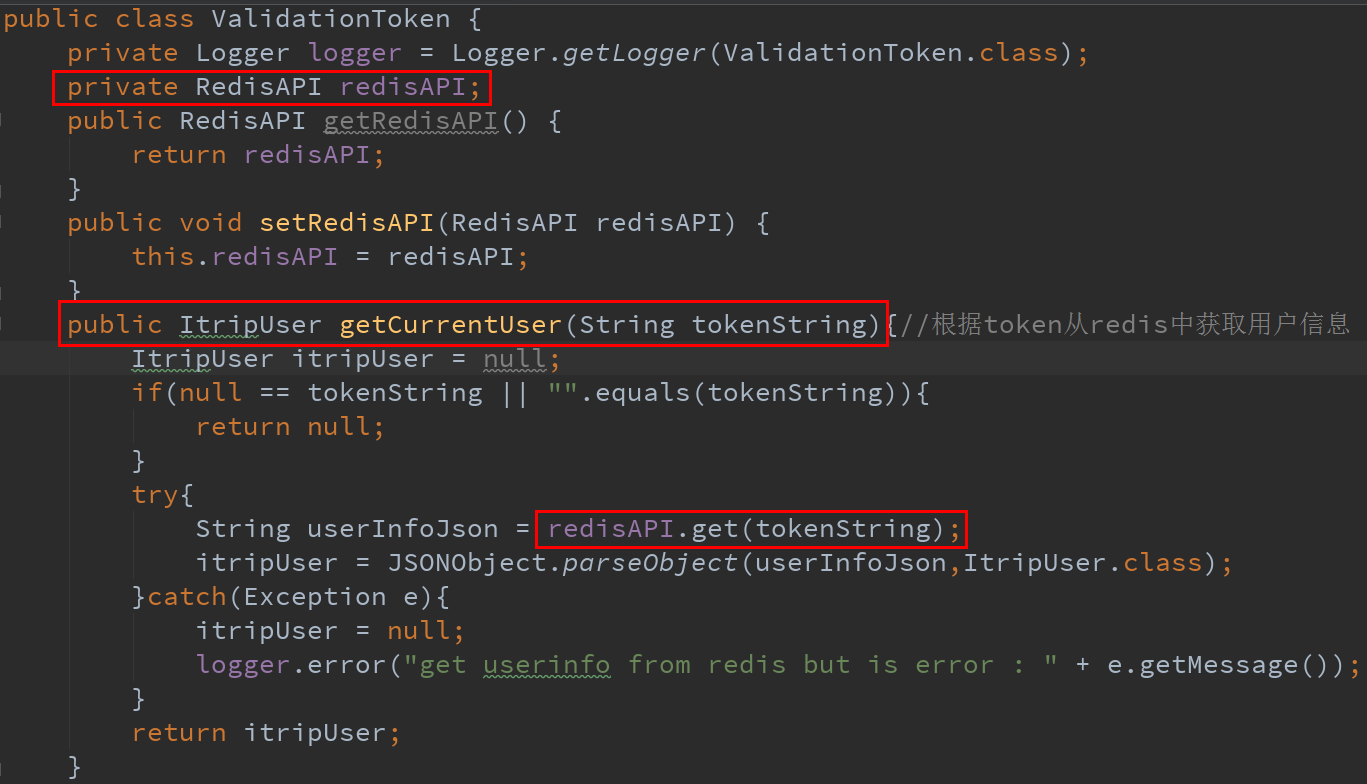
### 1.4.4 代码实现

根据分析，redisAPI需要提供以下六个方法，以后根据项目需求，该 API可以不断扩充。

1. boolean set(String key,String value)：set key value到redis中
2. boolean set(String key,int seconds,String value)：set key、value、有效期到redis中
3. boolean get(String key)：通过key获取value
4. boolean exist(String key)：判断某个key是否存在
5. delete(String key)：删除指定的key
6. Long ttl(String key)：查询key的有效期。当key 不存在时，返回 -2 ；当 key 存在但没有设置剩余有效时间时，返回 -1，否则，以秒为单位，返回 key 的剩余有效时间

作为业务实现，直接调用redisAPI的接口方法进行业务逻辑的处理即可。下面以工具类ValidationToken.java的代码实现为例进行介绍。

该工具类主要作用是调用redisAPI的get方法，通过传入的key（Token）来获取相应value值，即当前用户信息（ItripUser），以验证当前用户是否登录状态。若获取到value，即可获取ItripUser对象，否则返回空，则需要用户进行登录操作。代码片段如下：



注意：所有的工具类中尽量不要使用注解，以降低耦合度，比如：RedisAPI的注入。