所需**jar**：jedis-2.1.0.jar和commons-pool-1.5.4.jar
Jedis操作步骤如下：
1->获取Jedis实例需要从JedisPool中获取；
2->用完Jedis实例需要返还给JedisPool；
3->如果Jedis在使用过程中出错，则也需要还给JedisPool；
代码如下：

```
package com.ljq.utils;

import redis.clients.jedis.Jedis;
import redis.clients.jedis.JedisPool;
import redis.clients.jedis.JedisPoolConfig;

/**
 * Redis操作接口
 *
 * @author 林计钦
 * @version 1.0 2013-6-14 上午08:54:14
 */
public class RedisAPI {
    private static JedisPool pool = null;

    /**
     * 构建redis连接池
     *
     * @param ip
     * @param port
     * @return JedisPool
     */
    public static JedisPool getPool() {
        if (pool == null) {
            JedisPoolConfig config = new JedisPoolConfig();
            //控制一个pool可分配多少个jedis实例，通过pool.getResource()来获取；
            //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)。
            config.setMaxActive(500);
            //控制一个pool最多有多少个状态为idle(空闲的)的jedis实例。
            config.setMaxIdle(5);
            //表示当borrow(引入)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；
            config.setMaxWait(1000 * 100);
```

```
            //在borrow一个jedis实例时，是否提前进行validate操作；如果为true，则得到的jedis实例均是可用的；
            config.setTestOnBorrow(true);
            pool = new JedisPool(config, "192.168.2.191", 8888);
        }
        return pool;
    }

    /**
     * 返还到连接池
     *
     * @param pool
     * @param redis
     */
    public static void returnResource(JedisPool pool, Jedis redis) {
        if (redis != null) {
            pool.returnResource(redis);
        }
    }

    /**
     * 获取数据
     *
     * @param key
     * @return
     */
    public static String get(String key){
        String value = null;

        JedisPool pool = null;
        Jedis jedis = null;
        try {
            pool = getPool();
            jedis = pool.getResource();
            value = jedis.get(key);
        } catch (Exception e) {
            //释放redis对象
            pool.returnBrokenResource(jedis);
            e.printStackTrace();
        } finally {
            //返还到连接池
            returnResource(pool, jedis);
        }

        return value;
```

```
    }
}
```

代码说明：

a、获取jedis实例时，实际上可能有两类错误。
一类是pool.getReource()，得不到可用的jedis实例；
另一类是jedis.set/get时出错也会抛出异常；
为了实现区分，所以根据instance是否为null来实现，如果为空就证明instance根本就没初始化，也就不用return给pool；如果instance不为null，则证明是需要返还给pool的；
b、在instance出错时，必须调用returnBrokenResource返还给pool，否则下次通过getResource得到的instance的缓冲区可能还存在数据，出现问题！

JedisPool的配置参数很大程度上依赖于实际应用需求、软硬件能力。以前没用过commons-pool，所以这次花了一整天专门看这些参数的含义。。。JedisPool的配置参数大部分是由JedisPoolConfig的对应项来赋值的。
maxActive：控制一个pool可分配多少个jedis实例，通过pool.getResource()来获取；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。
maxIdle：控制一个pool最多有多少个状态为idle(空闲)的jedis实例；
whenExhaustedAction：表示当pool中的jedis实例都被allocated完时，pool要采取的操作；默认有三种。
WHEN_EXHAUSTED_FAIL --> 表示无jedis实例时，直接抛出NoSuchElementException；
WHEN_EXHAUSTED_BLOCK --> 则表示阻塞住，或者达到maxWait时抛出JedisConnectionException；
WHEN_EXHAUSTED_GROW --> 则表示新建一个jedis实例，也就说设置的maxActive无用；
maxWait：表示当borrow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alidate操作；如果为true，则得到的jedis实例均是可用的；
testOnReturn：在return给pool时，是否提前进行validate操作；
testWhileIdle：如果为true，表示有一个idle object evitor线程对idle object进行扫描，如果validate失败，此object会被从pool中drop掉；这一项只有在timeBetweenEvictionRunsMillis大于0时才有意义；
timeBetweenEvictionRunsMillis：表示idle object evitor两次扫描之间要sleep的毫秒数；
numTestsPerEvictionRun：表示idle object evitor每次扫描的最多的对象数；
minEvictableIdleTimeMillis：表示一个对象至少停留在idle状态的最短时间，然后才能被idle object evitor扫描并驱逐；这一项只有在timeBetweenEvictionRunsMillis大于0时才有意义；
softMinEvictableIdleTimeMillis：在minEvictableIdleTimeMillis基础上，加入了至

少minIdle个对象已经在pool里面了。如果为-1，evicted不会根据idle time驱逐任何对象。如果minEvictableIdleTimeMillis>0，则此项设置无意义，且只有在timeBetweenEvictionRunsMillis大于0时才有意义；
lifo：borrowObject返回对象时，是采用DEFAULT_LIFO（last in first out，即类似cache的最频繁使用队列），如果为False，则表示FIFO队列；
其中JedisPoolConfig对一些参数的默认设置如下：
testWhileIdle=true
minEvictableIdleTimeMills=60000
timeBetweenEvictionRunsMillis=30000
numTestsPerEvictionRun=-1