```java
public static void main( String[] args )
{
    //这是jedis连接池的设置
    JedisPoolConfig config = new JedisPoolConfig();
    config.setMaxTotal(100);//最大连接数
    config.setMaxIdle(10);//最大空闲连接数
    config.setNumTestsPerEvictionRun(30);//每次释放连接的最大数量
    config.setTimeBetweenEvictionRunsMillis(3000);//连接释放的扫描间隔
    config.setMinEvictableIdleTimeMillis(100000);//最小连接空闲时间
    //连接空闲多久后释放空间>该值&&空闲连接>最大空闲连接数时释放
    config.setSoftMinEvictableIdleTimeMillis(10000);
    config.setMaxWaitMillis(10000);//获取连接时的最大等待毫秒数
    config.setTestOnBorrow(true);//获取连接时检查有效性
    config.setTestWhileIdle(true);//连接耗尽时是否阻塞

    String host = "127.0.0.1";
    int port = 6379;

    JedisPool jedisPool = new JedisPool(config,host,port);
    Jedis jedis = jedisPool.getResource();

    int dataNum = 10000;

    //这儿测试普通方式和pipeline（管道）方式的比较
    long start = System.currentTimeMillis();
    for(int i = 0; i < dataNum; i++) {
        jedis.set(String.valueOf(i), String.valueOf(i));
    }
    long end = System.currentTimeMillis();
    System.out.println(end - start);

    Pipeline pipeline = jedis.pipelined();
    long start_pipe = System.currentTimeMillis();
    for(int i = 0; i < dataNum; i++) {
        pipeline.set(String.valueOf(i), String.valueOf(i+1000000));
    }
    pipeline.sync();
    long end_pipe = System.currentTimeMillis();
    System.out.println(end_pipe - start_pipe);
    jedisPool.close();
}
```

Problems @ Javadoc Declaration Console LogCat Servers Analysis Keys

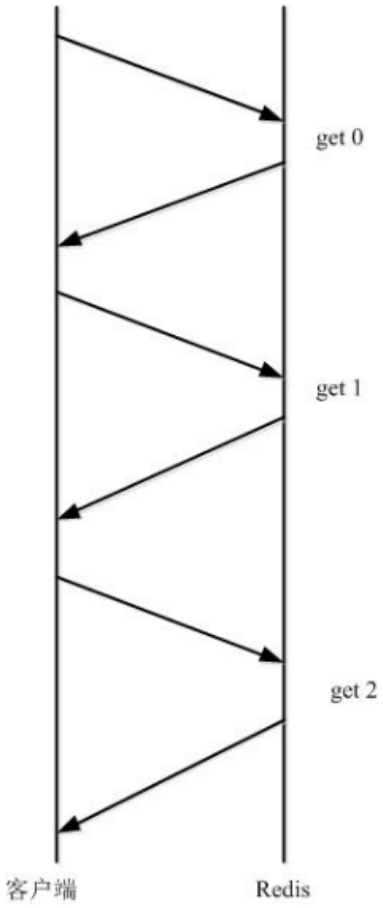
<terminated> App (2) [Java Application] C:\Program Files\Java\jdk1.8.0_112\bin\javaw.exe (2018年7月19日 上午11:48:04)

```
630
34
```

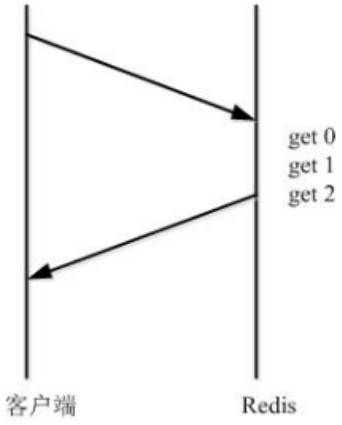
# PipeLine介绍

redis客户端与redis之间使用TCP协议进行连接，一个客户端可以通过一个socket连接发起多个请求。每个请求命令发出后client通常会阻塞并等待redis服务处理，redis处理完后请求命令后将结果通过响应报文返回给client，因此当执行多条命令的时候都需要等待上一条命令执行完毕后才能执行。

其执行过程如下图所示：




而管道（pipeline）可以一次性发送多条命令并在执行完后一次性将结果返回，pipeline通过减少客户端与redis的通信次数来实现降低往返延时时间，其过程如下图所示：




pipeline的sync方法是　相当于把每一次的请求操作加入到一个queue中，

```
 */
public void sync() {
  if (getPipelinedResponseLength() > 0) {
    List<Object> unformatted = client.getMany(getPipelinedResponseLength());
    for (Object o : unformatted) {
      generateResponse(o);
    }
  }
}
```

1）这个getPipelinedResponseLength()就是获取刚才的那个Queue的长度；

2）看看getMany()方法，一上来就fush(),这下前面代码循环写入的那么多set命名全部传到redis server了！

ps:flush刷新此输出流并强制写出所有缓冲的输出字节。flush 的常规协定是：如果此输出流的实现已经缓冲了以前写入的任何字节，则调用此方法指示应将这些字节立即写入它们预期的目标。

```
protected void flush() {
  try {
    outputStream.flush();
  } catch (IOException ex) {
    broken = true;
    throw new JedisConnectionException(ex);
  }
}

protected Object readProtocolWithCheckingBroken() {
  try {
    return Protocol.read(inputStream);
  } catch (JedisConnectionException exc) {
    broken = true;
    throw exc;
  }
}

public List<Object> getMany(final int count) {
  flush();
  final List<Object> responses = new ArrayList<Object>(count);
  for (int i = 0; i < count; i++) {
    try {
      responses.add(readProtocolWithCheckingBroken());
    } catch (JedisDataException e) {
      responses.add(e);
    }
  }
  return responses;
}
```

3）generateResponse(o)就是拿到返回值了，sync是没有返回值的，syncAndReturnAll有返回值