

BEIJING JIAOTONG UNIVERSITY

|  |  |
| --- | --- |
| 项目 | **软件学院**    **数据库应用系统综合实践**  **模仿多用户并发访问**  模仿多用户并发访问 |
| 负责人 | 田川 |
| 小组成员 | 皮慧成 孙焱君 田川 |
|  | 马越 孙逸 |
| 任课教师 | 孔令波 |

目录

[1 项目分析 3](#_Toc28528765)

[1.1 项目信息 3](#_Toc28528766)

[1.2 项目概要 3](#_Toc28528771)

[2 项目概述 3](#_Toc28528772)

[2.1 功能介绍 3](#_Toc28528773)

[2.2 原理分析 4](#_Toc28528774)

[2.2.1 CountDownLatch 4](#_Toc28528775)

[2.2.2 ThreadPoolExecutor 5](#_Toc28528778)

# 1 项目分析

## 1.1 项目信息

**项目名称**：模仿多用户并发访问

**小组成员**： 皮慧成 17301076

孙焱君 17301077

马越 17301074

田川 17301080

孙逸 17301044

**项目负责人**： 田川 17301080

**开发环境**：Java

## 1.2 项目概要

本项目要求利用CS架构，实现模拟多个用户并发访问

**2 项目概述**

**2.1 功能介绍**

客户端：每个客户端逻辑相同，随机生成一个字母，将字母传输到服务器。

服务器：接收字母并响应 client x for character

模仿数千个客户端同时发出请求，不分先后顺序。



## 2.2 原理分析

### 2.2.1 CountDownLatch

### CountDownLatch是一个同步工具类，它允许一个或多个线程一直等待，直到其他线程的操作执行完后再执行。它的实现原理通过一个计数器来实现的，计数器的初始值为线程的数量。每当一个线程完成了自己的任务后，计数器的值就会减1。当计数器值到达0时，它表示所有的线程已经完成了任务，然后在闭锁上等待的线程就可以恢复执行任务。比如主任务是一个比较复杂的运算，为了节约时间，我们可以拆分成多个子任务，多线程同时执行，最终统一汇总任务。

### 其实生活也有类似的场景，比如马拉松赛跑，我们不可能按顺序依次跑步，这样得跑到猴年马月。一般来说都是大家听到发号命令一起跑，最终比赛结束，统一汇总成绩。

### 2.2.2 ThreadPoolExecutor

### 在一个应用程序中，我们需要多次使用线程，也就意味着，我们需要多次创建并销毁线程。而创建并销毁线程的过程势必会消耗内存。而在Java中，内存资源是及其宝贵的，所以，我们就提出了线程池的概念。

### 线程池就是首先创建一些线程，它们的集合称为线程池。使用线程池可以很好地提高性能，线程池在系统启动时即创建大量空闲的线程，程序将一个任务传给线程池，线程池就会启动一条线程来执行这个任务，执行结束以后，该线程并不会死亡，而是再次返回线程池中成为空闲状态，等待执行下一个任务。