# 1.关于redis整理

# 2.多线程导出

当时在公司做一个报表,需要使用excel导出,做了好几个版本,进行优化,第一版 就选择了poi导出,通过反射+注解+zip来做,

反应慢,我们加上线程池;

反射是什么

自定义注解

线程池

Zip

1. 配置:我们需要配置上poi的POM依赖,直接可以使用
2. 思路:
3. 首先需要用户传来两个参数,即我们每个sheet的数据量(datasize)以及每个excel中有多少个sheet页即(sheetsize)
4. 查出我们需要导出的(数据量);

sheetAllCount(sheet总数)=总数据量/datasize;

excelCount(excel数量)= (sheet总数)/ (sheetsize);

1. 循环excel总数 每次循环 相当于创造一个excel的数据 , 创建workbox

再循环我们excel中sheet页的数量;得到当前excel中的每个sheet;创建之前 我们需要这个sheet的数据;我们需要通过分页查询按批次查询数据; 那么我们就需要 数据的起始下标与条数 条数我们可以传 每个sheet的数据量(datasize)

起始下标 :当前excel-1 \* sheetAllCount(sheet总数) x2 我们得到每个sheet数据,然后通过集合中的对象得到类对象;

1. 通过反射得到我们javabean的属性,给要添加的属性设置上自定义注解,value为我们的表头信息, getDeclaredFields获取所有字段的集合;循环判断加了注解;将注解的value设为表头;

Createcell.setcellvalue();

1. 循换每个sheet的数据; 创建row 在循环我们反射的bean属性;如果加了注解就创建一个cell;并判断类型;放入对应cell中;
2. 写入我们服务器文件中
3. 为了导出速度;我们加入线程池; 并使用contdownlach;
4. 等所有线程执行完成;我们通过java提供的zip进行压锁并响应; 将响应流加入我们的zip输出流;读写我们excel文件夹; 完成多线程导出;

线程池:

线程是系统非常珍贵的资源;线程池就是一个容器;用来存放线程;并帮我们线程执行任务;

可以减少我们线程创建与销毁的资源消耗;

创建方式 自定义型

Threadpoolexctor t=new Thredddpoolexctor

参数 :

1.核心线程数 即空闲也不会销毁的线程 核心数不足 提交任务会创建

2.最大线程数;核心数满了 就创建非核心 ;想家不能超过最大线程数

3.非核心超时时间: 非核心线程限制多久被回收;

4.线程队列:队列怎么处理线程 是直接创建新线程还是等待还是直接交给核心执行

5,饱和策略:丢弃任务 丢弃其他任务 新建线程 报异常

# 常用线程:

Fixedthredpool=Excutors.newfixedthredpool

固定大小的线程;核心与最大一致 控制并发量 异常 建新的

Cachedthredpool可缓存的线程池; 核心0 最大 max 有任务创建线程; 闲置回收 节约资源

Singlethredexecutor

一个单线程的线程池; 保证顺序

**ScheduledThreadPool**

创建一个可以定时 或 延时的线程池

还有一个单线程的定时延时线程

# 反射

反射是什么 反射就是可以让我们识别一个未知的类,一个未知的对象;

对于类来说就是 我们可以知道他里面的所有属性和所有方法

对于对象来说 我们可以得到所有的属性值和调用它里面的方法

Java反射机制是在运行状态中，对于任意一个类，都能够知道这个类的所有属性和方法；对于任意一个对象，都能够调用它的任意一个方法；这种动态获取的以及动态调用对象的方法功能称为java的反射机制

**反射机制的优缺点**

优点：

运行期类型的判断，动态类的加载，动态代理使用反射

缺点：

性能是一个

**获取类对象的三种方式**

类名点 . class 接收返回值 获取到类对象

Class areaclass= Area.class;

第二种方式

Class areaclass2=Class.forName(“com.jgxi.entity.po.Area”); 填写类的全路径

第三种方式

New对象  通过对象名 get获取class对象

Area a = new Area;

Class areaClass3=a.getClass();

自定义注解 :

一种标识 帮我识别需要的类 属性 方法;

@interf

元注解

@target 声明作用未知

@retention 声明生命周期

String value( ) default “”;

Zip 就是java提供的一个压缩包工具

Zipoutputstream zip输出流

Zipentry 相当于zip 中的一个位置